Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)

Michael Mandel, Justin Salamon, and Daniel P. W. Ellis (eds.)

New York University, NY, USA

October 2019

Citation:

Michael Mandel, Justin Salamon, and Daniel P. W. Ellis (eds.), Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019), New York University, NY, USA, Oct. 2019.

DOI: `https://doi.org/10.33682/1syg-dy60`

ISBN: 978-0-578-59596-2

Contents

Urban Noise Monitoring in the Stadtlärm Project - A Field Report
J. Abeßer, M. Götze, T. Clauß, D. Zapf, C. Kühn, H. Lukashevich, S. Kühnlenz & S. Mimilakis 1

Urban Sound Tagging using Convolutional Neural Networks
S. Adapa 5

A Multi-room Reverberant Dataset for Sound Event Localization and Detection
S. Adavanne, A. Politis & T. Virtanen 10

Sound Event Classification and Detection with Weakly Labeled Data
S. Adavanne, H. Fayek & V. Tourbabin 15

Localization, Detection and Tracking of Multiple Moving Sound Sources with a Convolutional Recurrent Neural Network
S. Adavanne, A. Politis & T. Virtanen 20

DCASE 2019 Task 2: Multitask Learning, Semi-supervised Learning and Model Ensemble with Noisy Data for Audio Tagging
O. Akiyama & J. Sato 25

Polyphonic Sound Event Detection and Localization using a Two-Stage Strategy
Y. Cao, Q. Kong, T. Iqbal, F. An, W. Wang & M. Plumbley 30

SONYC Urban Sound Tagging (SONYC-UST): A Multilabel Dataset from an Urban Acoustic Sensor Network
M. Cartwright, A.E. Mendez Mendez, J. Cramer, V. Lostanlen, G. Dove, H. Wu, J. Salamon, O. Nov & J.P. Bello 35

Non-Negative Matrix Factorization-Convolutional Neural Network (NMF-CNN) for Sound Event Detection
T.K. Chan, C.S. Chin & Y. Li 40

Acoustic Scene Classification Based on a Large-margin Factorized CNN
J. Cho, S. Yun, H. Park, J. Eum & K. Hwang 45

Hierarchical Detection of Sound Events and their Localization Using Convolutional Neural Networks with Adaptive Thresholds
S. Chytas & G. Potamianos 50

GCC-PHAT Cross-Correlation Audio Features for Simultaneous Sound Event Localization and Detection (SELD) on Multiple Rooms
H. Cordourier Maruri, P. López Meyer, J. Huang, J. del Hoyo Ontiveros & H. Lu 55

Language Modelling for Sound Event Detection with Teacher Forcing and Scheduled Sampling
K. Drossos, S. Gharib, P. Magron & T. Virtanen 59

Convolutional Recurrent Neural Network and Data Augmentation for Audio Tagging with Noisy Labels and Minimal Supervision
J. Ebbers & R. Häb-Umbach 64

Audio Tagging with Noisy Labels and Minimal Supervision
E. Fonseca, M. Plakal, F. Font, D.P.W. Ellis & X. Serra 69

Robust Non-negative Block Sparse Coding for Acoustic Novelty Detection
R. Giri, A. Krishnaswamy & K. Helwani 74

Sound Source Localisation in Ambisonic Audio Using Peak Clustering
M. Green & D. Murphy 79

Sound Event Localization and Detection Using CRNN on Pairs of Microphones
F. Grondin, I. Sobieraj, M. Plumbley & J. Glass 84

Multiple Neural Networks with Ensemble Method for Audio Tagging with Noisy Labels and Minimal Supervision
K. He, Y. Shen & W. Zhang 89

Acoustic Scene Classification Using Deep Learning-based Ensemble Averaging
J. Huang, H. Lu, P. López Meyer, H. Cordourier Maruri & J. del Hoyo Ontiveros 94

Neural Audio Captioning Based on Conditional Sequence-to-Sequence Model
S. Ikawa & K. Kashino 99

RU Multichannel Domestic Acoustic Scenes 2019: A Multichannel Dataset Recorded by Distributed Microphones with Various Properties
K. Imoto & N. Ono 104

Shuffling and Mixing Data Augmentation for Environmental Sound Classification
T. Inoue, P. Vinayavekhin, S. Wang, D. Wood, A. Munawar, B. Ko, N. Greco & R. Tachibana 109

Distilling the Knowledge of Specialist Deep Neural Networks in Acoustic Scene Classification
J. Jung, H. Heo, H. Shim & H. Yu 114

Sound Source Detection, Localization and Classification using Consecutive Ensemble of CRNN Models
S. Kapka & M. Lewandowski 119

Receptive-Field-Regularized CNN Variants for Acoustic Scene Classification
K. Koutini, H. Eghbal-zadeh & G. Widmer 124

SpecAugment for Sound Event Detection in Domestic Environments using Ensemble of Convolutional Recurrent Neural Networks
W. Lim 129

Guided Learning Convolution System for DCASE 2019 Task 4
L. Lin, X. Wang, H. Liu & Y. Qian 134

Crowdsourcing a Dataset of Audio Captions
S. Lipping, K. Drossos & T. Virtanen 139

Long-distance Detection of Bioacoustic Events with Per-channel Energy Normalization
V. Lostanlen, K. Palmer, E. Knight, C. Clark, H. Klinck, A. Farnsworth, T. Wong, J. Cramer & J.P. Bello 144

Acoustic Scene Classification from Binaural Signals using Convolutional Neural Networks
R. Mars, P. Pratik, S. Nagisetty & C. Lim 149

First Order Ambisonics Domain Spatial Augmentation for DNN-based Direction of Arrival Estimation
L. Mazzon, Y. Koizumi, M. Yasuda & N. Harada 154

The Impact of Missing Labels and Overlapping Sound Events on Multi-label Multi-instance Learning for Sound Event Classification
M. Meire, P. Karsmakers & L. Vuegen 159

Acoustic Scene Classification in DCASE 2019 Challenge: Closed and Open Set Classification and Data Mismatch Setups
A. Mesaros, T. Heittola & T. Virtanen 164

OtoMechanic: Auditory Automobile Diagnostics via Query-by-Example
M. Morrison & B. Pardo 169

Onsets, Activity, and Events: A Multi-task Approach for Polyphonic Sound Event Modelling
A. Pankajakshan, H. Bear & E. Benetos 174

TrellisNet-Based Architecture for Sound Event Localization and Detection with Reassembly Learning
S. Park 179

Weakly Labeled Sound Event Detection using Tri-training and Adversarial Learning
H. Park, S. Yun, J. Eum, J. Cho & K. Hwang 184

A Hybrid Parametric-Deep Learning Approach for Sound Event Localization and Detection
A. Perez-Lopez, E. Fonseca & X. Serra 189

Classifying Non-speech Vocals: Deep vs Signal Processing Representations
F. Pishdadian, P. Seetharaman, B. Kim & B. Pardo 194

Sound Event Localization and Detection using CRNN Architecture with Mixup for Model Generalization
P. Pratik, W. Jee, S. Nagisetty, R. Mars & C. Lim 199

Exploiting Parallel Audio Recordings to Enforce Device Invariance in CNN-based Acoustic Scene Classification
P. Primus, H. Eghbal-zadeh, D. Eitelsebner, K. Koutini, A. Arzt & G. Widmer 204

MIMII Dataset: Sound Dataset for Malfunctioning Industrial Machine Investigation and Inspection
H. Purohit, R. Tanabe, T. Ichige, T. Endo, Y. Nikaido, K. Suefusa & Y. Kawaguchi . . . 209

Sound Event Detection and Direction of Arrival Estimation using Residual Net and Recurrent Neural Networks
R. Ranjan, S. Jayabalan, T. Nguyen & W. Gan 214

Open-set Evolving Acoustic Scene Classification System
F. Saki, Y. Guo, C. Hung, L. Kim, M. Deshpande, S. Moon, E. Koh & E. Visser 219

HODGEPODGE: Sound Event Detection Based on Ensemble of Semi-Supervised Learning Methods
Z. Shi, L. Liu, H. Lin, R. Liu & A. Shi 224

Deep Multi-view Features from Raw Audio for Acoustic Scene Classification
A. Singh, P. Rajan & A. Bhavsar 229

Audio Tagging using Linear Noise Modelling Layer
S. Singh, A. Pankajakshan & E. Benetos 234

Robustness of Adversarial Attacks in Sound Event Classification
V. Subramanian, E. Benetos & M. Sandler 239

Improvement of DOA Estimation by using Quaternion Output in Sound Event Localization and Detection
Y. Sudo, K. Itoyama, K. Nishida & K. Nakadai 244

Hierarchical Sound Event Classification
E. Nichols, D. Tompkins & J. Fan 248

Sound Event Detection in Domestic Environments with Weakly Labeled Data and Soundscape Synthesis
N. Turpault, R. Serizel, J. Salamon & A. Shah 253

Open-Set Acoustic Scene Classification with Deep Convolutional Autoencoders
K. Wilkinghoff & F. Kurth 258

MAVD: A Dataset for Sound Event Detection in Urban Environments
P. Zinemanas, P. Cancela & M. Rocamora 263

Preface

This volume gathers the papers presented at the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019), New York University, NY, USA, during 25–26 October 2019.

The DCASE 2019 Workshop was the fourth workshop on Detection and Classification of Acoustic Scenes and Events, organized again in conjunction with the DCASE Challenge. The aim of the workshop was to bring together researchers from many different universities and companies with interest in the topic, and provide the opportunity for scientific exchange of ideas and opinions.

The DCASE 2019 Workshop was jointly organized by researchers at New York University, Brooklyn College, Cornell University, Adobe, and Google, Inc. The associated DCASE 2019 Challenge tasks were organized by researchers at Tampere University (Task 1: Acoustic scene classification, Task 3: Sound Event Localization and Detection); Universitat Pompeu Fabra and Google, Inc. (Task 2: Audio tagging with noisy labels and minimal supervision); University of Lorraine, Inria Nancy Grand-Est, Carnegie Mellon University, Johannes Kepler University, and Adobe (Task 4: Sound event detection in domestic environments); and New York University, Cornell University, and Adobe (Task 5: Urban Sound Tagging).

For this edition of the DCASE 2019 Workshop, 82 full papers were submitted, each reviewed by at least three members of our Technical Program Committee. From these, 54 papers were accepted, 14 for oral presentation and 40 for poster presentation.

The Organizing Committee was also pleased to invite leading experts for keynote addresses:

- Catherine Gustavino (McGill University, School of Information Studies)
- Jessie Barry (Cornell University, The Cornell Lab of Ornithology)

The success of the DCASE 2019 Workshop was the result of the hard work of many people whom we wish to warmly thank here, including all the authors and keynote speakers, as well as all the members of the Technical Program Committee, without whom this edition of the DCASE 2019 Workshop would not exist. We also wish to thank the organizers and participants of the DCASE Challenge tasks.

This edition of the workshop was supported by sponsorship from Adobe, Amazon Alexa, Audio Analytic, Bose, Cochlear.ai, Facebook, Google, IBM Research, Microsoft, Mitsubishi Electric, and Sound Intelligence. We wish to thank them warmly for their valuable support to this workshop and the expanding topic area.

Juan P. Bello
Mark Cartwright
Michael Mandel
Justin Salamon
Daniel P. W. Ellis
Vincent Lostanlen

URBAN NOISE MONITORING IN THE STADTLÄRM PROJECT - A FIELD REPORT

Jakob Abeßer[1], *Marco Götze*[2], *Tobias Clauß*[1], *Dominik Zapf*[1]
Christian Kühn[1], *Hanna Lukashevich*[1], *Stephanie Kühnlenz*[3], *Stylianos Mimilakis*[1]

[1] Fraunhofer Institute for Digital Media Technology (IDMT), Ilmenau, Germany
[2] IMMS Institut für Mikroelektronik- und Mechatronik-Systeme
gemeinnützige GmbH, Ilmenau, Germany
[3] Software-Service John GmbH, Ilmenau, Germany
jakob.abesser@idmt.fraunhofer.de

ABSTRACT

As noise pollution in urban environments is constantly rising, novel smart city applications are required for acoustic monitoring and municipal decision making. This paper summarizes the experiences made during the field test of the Stadtlärm system for distributed noise measurement in summer/fall of 2018 in Jena, Germany.

Index Terms— internet of things (IoT), smart city, acoustic scene classification, sensor network, noise level measurement

1. INTRODUCTION

Urban dwellers are often exposed to high levels of noise from a variety of sources such as road traffic, construction sites and public sport and music events. Ideally, the city administration needs to systematically investigate any complaint. However, this task is hardly feasible due to the high personnel and cost involved.

As previously introduced in [1], the main goal of the Stadtlärm research project was to develop a distributed noise monitoring system, which supports city management by continuously measuring noise levels and sources. The developed system of 12 distributed sensors was subjected to a field test lasting several months after one and a half years of development. This paper presents a field report and discusses various experiences that have been made. We hope that this experience report will also be useful for other research projects in the field of IoT and Smart Cities. Related research projects propose similar solutions for noise monitoring in urban environments (see for instance [2, 3, 4]). Among others, the special focus during the Stadtlärm project was on checking legally prescribed noise limits.

The paper will be structured as follows. Section 2 briefly describes the individual components of the Stadtlärm noise monitoring system. Then, Section 3 discusses practical hardware considerations such as the microphone selection, weather resistance, moisture protection, as well as the long-term operating status of the sensor devices. Different measurements towards on-site sound propagation will be described in Section 4. Finally we will illustrate how the set of acoustic classes was refined and the algorithm for acoustic scene classification was improved during the project duration in Section 5.

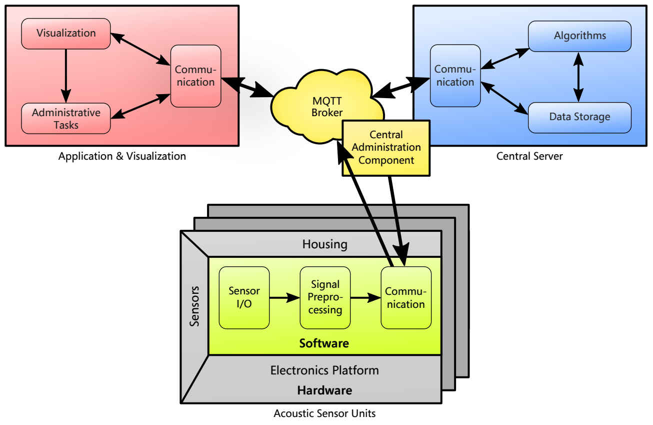


Figure 1: Stadtlärm system overview. The block diagram illustrates the data measurement and processing on the acoustic sensor units, communication via an MQTT broker, as well as data storage and visualization [1].

2. SYSTEM OVERVIEW

The Stadtlärm noise monitoring system consists of multiple distributed sensors, i. e., embedded systems that record and process audio data, and server-side services for further audio processing, data storage and access, as well as user applications. The overall system's backbone is a broker-based communications architecture. All communications among components and services utilize MQTT via a central broker, which also handles authentication and authorization. We used MQTT as the underlying communication protocol. Therefore, we defined a topics hierarchy and extended the publish/ subscribe paradigm by a convention enabling a request/response mechanism. The communications architecture is discussed in more detail in [5].

As shown in Figure 1, the system's components can be partitioned into three main functional groups. The acoustic sensor units are embedded field devices for acoustic data acquisition (see Section 3). On the software side, the sensors run embedded Linux and a custom application (implemented in Go), which handles communication (data, metadata, and management aspects). This application is also responsible for integration with the actual audio retrieval and processing software (implemented in Python). This software performs level measurements according to requirements of the German

https://doi.org/10.33682/s9w3-5341

TA Lärm regulations (see Section 4) as well as acoustic scene classification based on deep neural networks (see Section 5). By only transmitting the noise level and classification analysis results, privacy by design is implemented (as no speech recognition / surveillance is possible) and the amount of data for mobile communications is reduced.

The Stadtlärm audio service runs on a central server. After receiving measurement data from the acoustic sensor units, the service complements their data processing by computing long-term noise level parameters as defined in the German TA Lärm regulations. Data is stored and made available by request through an MQTT API. The Stadtlärm web application runs on a separate server and offers a web frontend to the data accumulated by the system tailored for administrative employees in departments concerned with urban noise. As such, it represents sensors on a city map, indicating status and current noise levels per location. On a per-sensor basis, historical level and classification data can be viewed, and reports can be created based on this.

Central to the system, an MQTT broker (Mosquitto) runs on another server and acts as a communications hub with central authentication/authorization. The broker is complemented by a central administration component serving as a registry of acoustic sensor units, a monitor of the overall system's status and load, and providing facilities to manage the sensors.

3. SENSOR HARDWARE

The acoustic sensor units are custom embedded devices developed in the course of the project. Each unit consists of a computation platform (Raspberry Pi 3 Compute Module Lite) and a microphone integrated on a custom PCB that also addresses aspects of robustness (robust power supply, hardware watchdog, etc.) and communications (M.2 slot for a wireless modem). The required computing power was estimated based on an initial implementation of the audio processing software (implemented in Python and utilizing the *keras* deep learning framework). Thanks to further optimizations made during the algorithmic development in the course of the project, the final CPU load is about 50% (across its 4 cores, i.e., equivalent to 200% load on a single core) and thus more than adequate, with ample reserves for communications and management as well as potential future adaptations.

For communications, both mobile wireless communications and utilizing public WiFi networks were considered. Each sensor unit is outfitted with a wireless modem and a SIM card. The total data volume used per month and unit is less than 1 GB (even including a remote firmware update or two), with the net amount of audio data being communicated by a single sensor being about 9 MB per day. As the sensors were meant to be mounted on lighting posts, assuming a permanent mains supply was an unrealistic assumption. The devices were therefore fitted with an additional power management PCB and a rechargeable battery (150 Wh), which enable off-grid operation with a periodic recharge (of at least 5 hours per day, typically overnight). Both computations and always-on mobile communications (with data being sent in near-real-time) result in a continuous power consumption of about 4 W. For this, the battery's capacity is ample; the situation is further relaxed by the fact that, in the field trial, lighting posts are powered for significantly more than 5 hours in the annual average, topping 16 hours in winter.

The sensor systems are fitted in weatherproof housings (for cost reasons, off-the-shelf) meant to be mounted on lighting posts of varying heights and diameters, at heights of 3 m or more. For this, the housings were fitted with steel strap clamps, which are flexible with respect to a post's diameter. The housing's external dimensions are 25 x 35 x 15 cm^3, so it is large enough to fit all the components and even leaves some room for optional equipment. It features a lockable hinged door, enabling easy but restricted access to its interior, necessary for wiring the device up to mains power during installation. For the pilot trial, the devices were otherwise assembled and configured manually prior to their being rolled out.

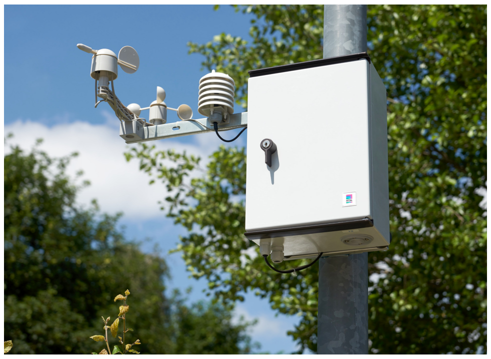


Figure 2: A Stadtlärm sensor equipped with the optional weather station components.

Another challenge was the integration of the microphone. Industry-grade microphones in outdoor-capable housings cost easily as much as the rest of the sensor system, which is why the goal with respect to the microphone was to use low-cost hardware in conjunction with state-of-the-art processing to nevertheless deliver high-quality audio recognition results. To this end, a MEMS microphone (ICS 43434, with an I^{2}S interface) was selected based on an systematic measurement and comparison of various microphone models. The easiest way of integrating that with the housing was to place the microphone directly behind a 4 mm drill hole at the bottom. Frequency response measurements and field recordings confirmed this to be adequate for the purpose of noise measurement and acoustic scene classification. However, in the course of the pilot trial, wind noise manifested itself as an issue of underestimated impact, requiring to make adaptions to the underlying acoustic scene classification model (see Section 5).

As weather may have profound effects on audio propagation and reception, a subset of sensor units in the pilot trial were extended by an inexpensive weather station measuring wind speed and direction as well as temperature and ambient humidity (Fig. 2). Internally, these additional components are connected via USB, proving the extensibility of the hardware platform. The weather data (measured by 3 devices placed in representative locations during the field test) correlate closely with weather data available from public providers, with plausible local deviations nevertheless being evident.

4. NOISE LEVEL MEASUREMENT

4.1. Sound Propagation in the Target Area

We performed an experiment to investigate the sound propagation in the target area of the field test. Therefore, given different configurations of source-to-sensor distances, we measured the drop in sound level. As a rule of thumb, we expect the sound level to drop by 6dB for a doubling of the distance. The test setup included an om-

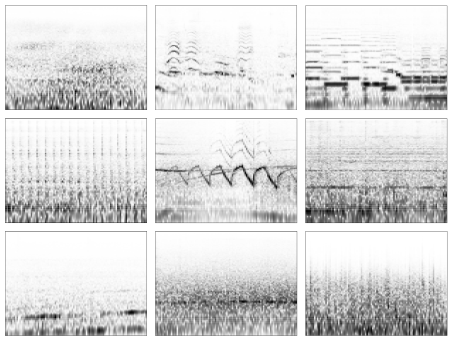


Figure 3: Examples of two-second long spectrogram patches for the classes car, conversation, music, roadworks, siren, train, tram, truck, and wind (from top left to right bottom), which are processed by the neural network.

nidirectional loudspeaker (GlobeSource Radiator by Outline) with a mobile power supply unit, which was powered by a truck battery. The measurement device was a NTi Audio XL2 with a calibrated Earthworks M30 as microphone. The Earthworks microphone was phantom-powered by the internal batteries of the NTi XL2 audio. We initially set up the SPL at a distance of 1 meter to 90 dB and for the second play-through to 100 dB.

As test signals, we used a simple sinusoidal signal with frequency of 1 kHz as well as a pink noise signal. A third test signal was a compilation of audio recordings covering different acoustic scenes. We repeated the test procedure at six different locations within the target area. From the results, we found that the measured SPL levels were slightly higher than expected. Possible reasons could be the rustling of the leaves or the wind, sound reflections by trees and buildings, as well as noises caused by pedestrians and cyclists nearby.

4.2. Case Study: Noise Pollution caused by Soccer Games

Residents from the nearby residential areas around the target area often complain about noise from soccer games, which happen typically on the weekends. As a show case, we analyzed the loudness curves, which have a temporal resolution of 0.125 s, recorded at the sound emission location i. e., the soccer stadium as well as a higher residential area around 600 m away. By computing the cross-correlation between overlapping 30 second long segments of the loudness curve, we derive the local maximum cross-correlation. Based on an additional test-recording nearby the second sensor, we were able to associate peaks in the cross-correlation curve with typical acoustic events during a soccer game such as drumming, fan cheering, and announcements from the stadium speaker. This provides clear evidence that noise from the soccer game is audible even in surrounding residential areas.

5. ACOUSTIC SCENE AND EVENT CLASSIFICATION

5.1. Acoustic Classes

In the final project stage, the initial number of acoustic scene classes (as presented in [1]) was reduced from 18 to 9 by focusing on the most relevant noise sources in the target area. In order to improve the applicability of the Stadtlärm system for traffic monitoring applications, the "truck" class was added to distinguish between the four most relevant vehicle types in the target site—cars, trains, trams, and trucks. During test recordings, we found that wind noises often overshadow other present noise sources, hence we added "wind" as an additional sound class. The intuition was to get a better sense of the classification confidence of other recognized sound sources. The initial sound classes "busking", "music event", and "open-air" were merged to a unified class "music", which includes all music-related events. For the class "siren", we improved the training data by adding recordings of local police cars, fire trucks, and ambulances. At the same time, we discarded the sound event classes "applause, "chants", "horn", and "shouting" as well as the (more ambiguous) sound scenes "public place" and "sports events" for now.

5.2. Acoustic Scene Classification

The classification model, which processes the recorded audio stream on the sensor units, was improved following the convolutional neural network (CNN) architecture proposed in [6]. The network is based on the VGG model paradigm that includes pairs of convolutional layers with small filter size and no intermediate pooling operation. In the applied architecture, four such layer pairs are concatenated with increasing number of filters (32, 64, 128, 256). After each layer pair, we apply a (3,3) max pooling in order to gradually decrease the spatial resolution and to encourage learning more abstract spectrogram patterns in higher layers. In order to improve the model's generalization towards unseen input data, we apply batch normalization between the convolutional layers, global average pooling between the convolutional and dense layers, dropout (0.25) between the final dense layers, as well as L2 regularization (0.01) on the penultimate dense layer.

Given the project-specific set of acoustic classes discussed in the previous section, we assembled training data from various publically available datasets such as the Urban Sound dataset[1], the TUT acoustic scenes 2016 dataset[2], as well as the freefield1010 dataset[3].

5.3. Results & Observations

Figure 3 shows examples of two-second long spectrogram patches for all 9 classes as they are processed by the neural network. These plot give a better intuition of the difficulty of the classification task. The classes "conversation", "music", "roadworks", and "siren" show distinctive patterns of either harmonic components (horizontal lines) or percussive components (vertical lines). For the vehicle-related classes "car", "train", "tram", and "truck", the sounds are more complex and noisy. The choice of an analysis window of two seconds (and a processing hop-size of one second) allows for a pseudo real-time processing of the recorded audio streams at the sensor units. However, such short analysis windows cannot capture slowly changing sounds such as passing vehicles. The noisy nature of wind also becomes apparent from the plot.

Figure 4 shows the receiver operation characteristic (ROC) curves as well as the precision-recall curves, which we obtained in a classification experiment on a separate test set, which was not used in the model training process. For each class, we obtained an

[1]https://urbansounddataset.weebly.com/urbansound.html

[2]https://zenodo.org/record/45739

[3]https://c4dm.eecs.qmul.ac.uk/rdr/handle/123456789/35

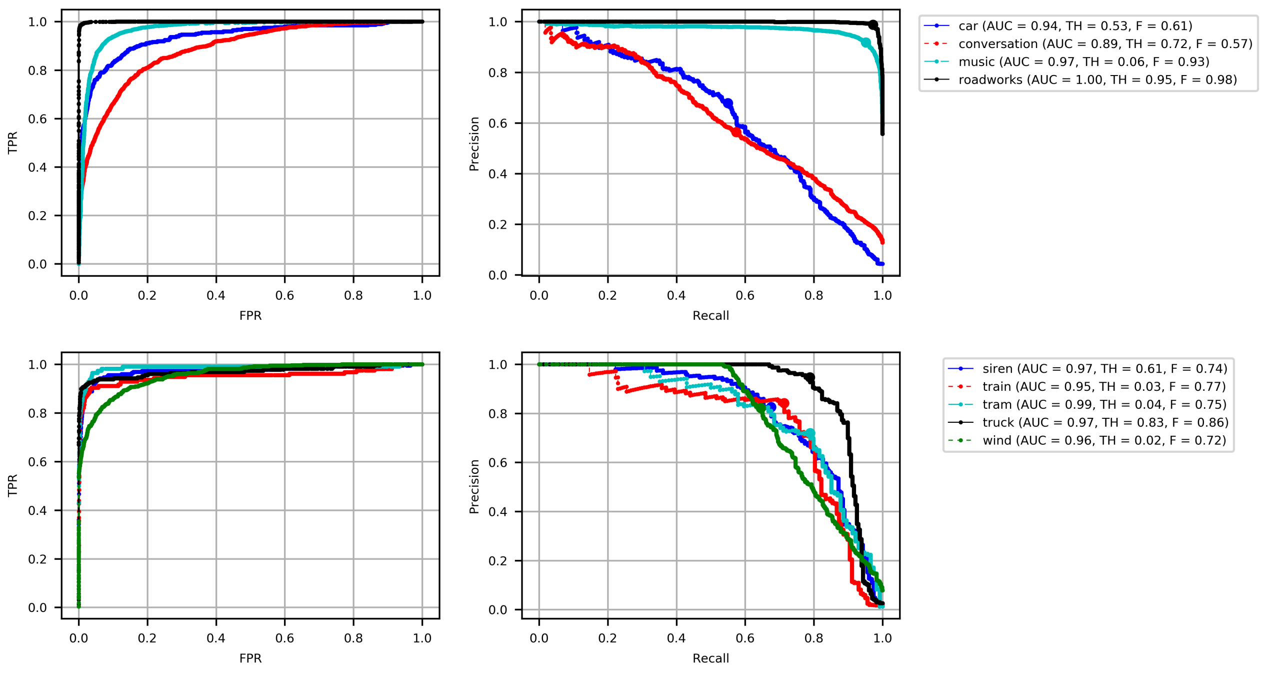


Figure 4: Receiver operation curves (ROC) and precision-recall curves for all 10 sound classes. Area-under-the-curve (AUC), optimal class-wise decision thresholds and best f-measure values are given in brackets.

optimal decision treshold on the sigmoid output values from the final network layer by maximizing the f-score. While all classes get a high AUC (area-under-the-curve) value, the precision-recall curves provide a better insight into the classifier performance.

When analyzing test recordings, which were made on the target site with the sensor, we observed a constant level of noise, either from environmental factors such as wind and rain or from the sensor hardware itself. Future steps for improving the model could be a stronger focus on data augmentation, where the high-quality audio recordings in the training data sets will be mixed with on-site background noise recordings to make the model more robust. Also we plan to test recently proposed methods for domain adaptation to achieve more robust classification results in different test environments.

6. WEB APPLICATION

We used the field test to evaluate the practical suitability of the web application. Based on continuous communication with the local city administration, we received regular feedback and optimized the main functionalities of the web application. This mainly concerned the identification of the prominent noise sources as well as the overall documentation of the noise level measurements. By providing the estimated noise source class probabilities from the model, non-expert users can better interpret the confidence of the acoustic scene classifier.

7. ACKNOWLEDGMENTS

The Stadtlärm project is funded by the German Federal Ministry for Economic Affairs and Energy (BMWi, ZF4248201LF6, ZF4248301LF6, ZF4072003LF6, ZF4085703LF6).

8. REFERENCES

[1] J. Abeßer, M. Götze, S. Kühnlenz, R. Gräfe, C. Kühn, T. Clauß, and H. Lukashevich, “A distributed sensor network for monitoring noise level and noise sources in urban environments,” in *Proceedings of the 6th IEEE International Conference on Future Internet of Things and Cloud (FiCloud)*, 2018, pp. 318–324.

[2] J. P. Bello, C. Mydlarz, and J. Salamon, “Sound analysis in smart cities,” in *Computational Analysis of Sound Scenes and Events*, T. Virtanen, M. D. Plumbley, and D. P. W. Ellis, Eds. Springer International Publishing, 2018, pp. 373–397.

[3] A. J. Fairbrass, M. Firman, C. Williams, G. J. Brostow, H. Titheridge, and K. E. Jones, “Citynet—deep learning tools for urban ecoacoustic assessment,” *Methods in Ecology and Evolution*, pp. 1–12, 2018.

[4] P. Maijala, Z. Shuyang, T. Heittola, and T. Virtanen, “Environmental noise monitoring using source classification in sensors,” *Applied Acoustics*, vol. 129, pp. 258 – 267, 2018.

[5] M. Götze, R. Peukert, T. Hutschenreuther, and H. Töpfer, “An open platform for distributed noise monitoring,” in *Proceedings of the 25th Telecommunications Forum (TELFOR) 2017*, Belgrade, Serbia, 2017, pp. 1–4.

[6] Y. Sakashita and M. Aono, “Acoustic scene classification by ensemble of spectrograms based in adaptive temporal division,” in *Detection and Classification of Acoustic Scenes and Events Challenge (DCASE)*, 2018.

URBAN SOUND TAGGING USING CONVOLUTIONAL NEURAL NETWORKS

Sainath Adapa

FindHotel
Amsterdam, Netherlands
adapasainath@gmail.com

ABSTRACT

In this paper, we propose a framework for environmental sound classification in a low-data context (less than 100 labeled examples per class). We show that using pre-trained image classification models along with the usage of data augmentation techniques results in higher performance over alternative approaches. We applied this system to the task of Urban Sound Tagging, part of the DCASE 2019. The objective was to label different sources of noise from raw audio data. A modified form of MobileNetV2, a convolutional neural network (CNN) model was trained to classify both coarse and fine tags jointly. The proposed model uses log-scaled Mel-spectrogram as the representation format for the audio data. Mixup, Random erasing, scaling, and shifting are used as data augmentation techniques. A second model that uses scaled labels was built to account for human errors in the annotations. The proposed model achieved the first rank on the leaderboard with Micro-AUPRC values of 0.751 and 0.860 on fine and coarse tags, respectively.

Index Terms— DCASE, machine listening, audio tagging, convolutional neural networks

1. INTRODUCTION

The IEEE AASP challenge on Detection and Classification of Acoustic Scenes and Events (DCASE) [1], now in its fifth edition, is a recurring set of challenges aimed at developing computational scene and event analysis methods. In Task 5, Urban Sound Tagging, the objective is to predict the presence or absence of 23 different tags in audio recordings. Each of these tags represents a source of noise and thus a cause of noise complaints in New York City. Solutions for this task, such as the one proposed in this paper, will help inspire the development of solutions for monitoring, analysis, and mitigation of urban noise.

2. RELATED WORK

The current task of Urban Sound Tagging is part of the broader research area of Environmental Sound Classification [1]. Convolutional neural networks (CNNs) that use Log-scaled Mel-spectrogram as the feature representation have been proven to be useful for this use case [2, 3], and have also achieved leading performance in recent DCASE tasks [4, 5, 6]. Extensions to the CNN framework, in the form of Convolutional Recurrent Neural Networks (CRNNs) have been proposed [7]. Transformation of the raw audio waveform into the Mel-spectrogram representation is a "lossy" operation [8]. As such, there has been ongoing research into evaluating alternatives such as using Scattering transform [9], Gammatone filter bank [7] representations, as well as directly employing one-dimensional CNN on the raw audio signal [10]. Operating in the context of noisy labels [11] or in a low-data regime [12] (both of which are properties of the present task) are two other active research areas in this domain. One particular approach for dealing with small labeled datasets is the usage of pre-trained models to generate embeddings that can be used for downstream audio classification tasks. VGGish[13], SoundNet[14], and L^3-Net[15] are examples of such models.

3. DATASET

For this challenge, SONYC [16] has provided 2351 recordings as part of the *train set*, and 443 recordings as a part of the *validate set*. All the recordings, acquired from different acoustic sensors in New York City, are Mono channel, sampled at 44.1kHz, and are ten seconds in length. The private *evaluation set* consisted of 274 recordings. Labels for these recordings were revealed only at the end of the challenge. A single recording might contain multiple noise sources. Hence, this is a task of multi-label classification.

The 23 noise tags, termed *fine-grained tags*, are further grouped into a list of 7 *coarse-grained tags*. This hierarchical relationship is illustrated in Figure 1. Each recording was annotated by three Zooniverse[2] volunteers. Additional annotations, specifically for *validate set*, were performed by the SONYC team members and ground truth is then agreed upon by the SONYC team. Since the *fine-grained tags* are not always easily distinguishable, annotators were given the choice of assigning seven tags of the form "other/unknown" for such cases. Each of these seven tags termed "incomplete tags," correspond to a different coarse category.

4. PROPOSED FRAMEWORK

4.1. CNN Architecture

In this work, we use a modified form of MobileNetV2 [18]. The architecture of MobileNetV2 contains a 2D convolution layer at the beginning, followed by 19 *Bottleneck residual blocks* (described in Table 1). Spatial average of the output from the final residual block is computed and used for classification via a Linear layer.

The proposed model makes few modifications to the above-described architecture. The input Log Mel-spectrogram data is sent to the MobileNetV2 after passing it through two convolution layers. This process transforms the single-channel input into a three-channel tensor to match the input size of original MobileNetV2 architecture. Instead of the spatial average, Max pooling is applied

[1]http://dcase.community/

[2]https://www.zooniverse.org/

https://doi.org/10.33682/8axe-9243

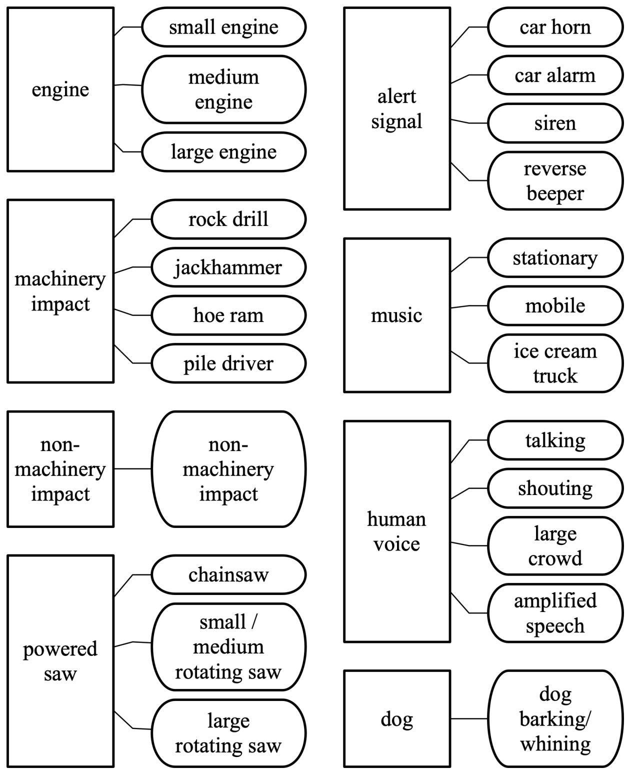


Figure 1: Hierarchical taxonomy of tags. Rectangular and round boxes respectively denote coarse and fine tags respectively. [17]

Input	Operator	Output
$h \times w \times k$	1x1 ,	$h \times w \times (tk)$
$h \times w \times tk$	3x3 s=s,	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1x1	$\frac{h}{s} \times \frac{w}{s} \times k'$

Table 1: *Bottleneck residual block* transforming from k to k' channels, with stride s, and expansion factor t.

to the output from the final residual block. Additionally, the single linear layer at the end is replaced by two linear layers. The full architecture is described in Table 2.

4.2. Initialization with Pre-trained weights

In many fields, including in the acoustic area, CNNs exhibit better performance with an increase in the number of layers [19, 20]. However, it has been observed that deeper neural networks are harder to train and prone to overfitting, especially in the context of limited data [21].

Many of the *fine-grained tags* have less than 100 training examples with positive annotations, thus placing the current task into a *low-data regime* context [12]. Since the proposed architecture has a large (24) number of layers, we initialized all the unmodified layers of the network with weights from the MobileNetV2 model trained on ImageNet [22, 23]. Kaiming initialization [24] is used for the remaining layers. Since the domain of audio classification is different from image classification, we do not employ a Fine-tuning approach [25] here. All the layers are jointly trained from the beginning. When all the layers with ImageNet weights were frozen at that parameters, the model performed worse than the baseline model (Section 6) showing the need for joint training of the whole network.

Operator	t	c	n	s
conv2d	**-**	**10**	**1**	**1**
conv2d	**-**	**3**	**1**	**1**
conv2d	-	32	1	2
bottleneck	1	16	1	1
bottleneck	6	24	2	2
bottleneck	6	32	3	2
bottleneck	6	64	4	2
bottleneck	6	96	3	1
bottleneck	6	160	3	2
bottleneck	6	320	1	1
conv2d 1x1	-	1280	1	1
maxpool	**-**	**1280**	**1**	**-**
linear	**-**	**512**	**1**	**-**
linear	**-**	**k**	**1**	**-**

Table 2: Each line describes a sequence of 1 or more identical (modulo stride) layers, repeated n times. All layers in the same sequence have the same number c of output channels. The first layer of each sequence has a stride s and all others use stride 1. All spatial convolutions use 3×3 kernels (except for the first two which use 1×1 kernels). The expansion factor t is always applied to the input size as described in Table 1. Modifications to the MobileNetV2 architecture are highlighted in bold.

	ImageNet pre-trained weights	Kaimin initialization
train set loss	0.1401 ± 0.0017	0.1493 ± 0.0019
validate set loss	0.1200 ± 0.0008	0.1266 ± 0.0022

Table 3: Final Binary Cross-entropy loss values at the end of training. 5 repetitions of training runs from scratch were performed.

The rationale behind the use of ImageNet weights is that the kind of filters that the ImageNet based model has learned are applicable in the current scenario of Spectrograms as well. Especially the filters in the initial layers that detect general patterns like edges and textures[26] are easily transferable to the present case. With the described initialization, we noticed faster and better convergence (illustrated in Figure 2 and Table 3) when compared to initializing all the layers with Kaimin initialization. Similar gains were observed previously in the context of Acoustic Bird Detection [3].

Other pre-trained models such as ResNeXt[27], and EfficientNet[28] were also tested. The observed metrics were at the same level as the MobileNetV2 architecture. Since the performance is similar, MobileNetV2 was chosen as it has the least number of parameters among the models tried.

4.3. Preprocessing and Data augmentation

The proposed model uses Log Mel-spectrogram as the representation format for the input data. Librosa [29] toolbox was used to compute the Mel-spectrogram. For the Short-time Fourier transform (STFT), *window length* of 2560 and *hop length* of 694 was

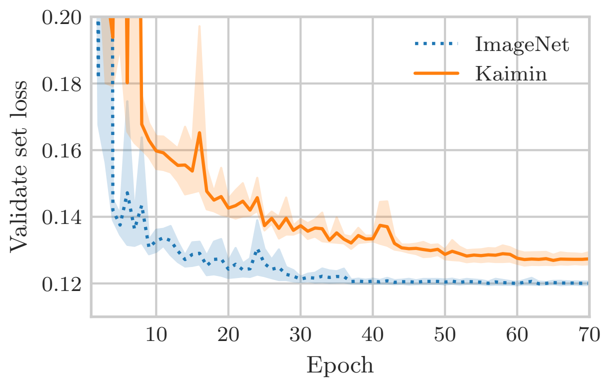


Figure 2: Trajectory of *validate set* loss during training, demonstrating that using pre-trained ImageNet weights results in faster convergence.

	Fine-level Micro-AUPRC	Coarse-level Micro-AUPRC
No data augmentation	0.716	0.819
Only Mixup	0.745	0.840
Only Random erasing	0.732	0.820
Only Random rotate	0.728	0.832
Only Shifting time	0.719	0.822
Only Grid distortion	0.753	0.842
Pitch shifting and Time stretching	0.732	0.834
All the techniques	0.772	0.855

Table 4: Performance on the *validate set*, demonstrating the gains due to data augmentation

used. For the Mel-frequency bins computation, the lowest and the highest frequencies were set at 20Hz and 22050Hz, respectively, with the number of bins being 128.[3] No re-sampling or additional preprocessing steps were performed.

Several data augmentation techniques were used to supplement the training data. Deformations such as Time stretching and Pitch shifting that were previously shown to help in sound classification were employed [2]. Also, image augmentation methods such as Random rotate, Grid distortion [30], and Random erasing [31] were used. Mixup [32], an approach that linearly mixes two random training examples was used as well. Table 4 shows the impact of Data augmentations, when each of the methods were applied separately.

4.4. Re-labeling

For the *validate set*, we have access to both the ground truth and the three sets of annotations by Zooniverse volunteers. When the ground truth of a label is positive, 36% of annotations (by Zooniverse volunteers) do not match with the ground truth. If the quality of the labels can be improved, it is quite possible that the accuracy of the model can be increased as well. Hence, a logistic regression model that takes the annotations as input and estimates the ground truth label was developed. This model was trained on the *validate* set, and then the ground truth estimate for the *train set* was generated. Table 5 shows a sample of predictions from the model.

Coarse label	Fine label	Positive annotations count	Predicted score
music	uncertain	1	0.10
music	uncertain	3	0.98
music	stationary	2	0.88
powered saw	chainsaw	3	0.98
machinery impact	-	0	0.05

Table 5: Predictions for few cases from the automatic re-labeling model

5. MODEL TRAINING

5.1. Evaluation metric

Area under the precision-recall curve using the micro-averaged precision and recall values (Micro-AUPRC) is used as the classification metric for this task. Micro-F1 and Macro-AUPRC values are reported as secondary metrics. Detailed information about the evaluation process is available on the task website [17].

5.2. Training

Two models were trained for this challenge:

M1: The first model generates probabilities for both the fine and coarse labels. During training, whenever the annotation is ”unknown/other”, loss for the fine tags corresponding to this coarse tag was masked out. Hence, this model does not generate predictions for *uncertain* fine labels. Since there are three sets of annotations for each training example, one by each Zooniverse volunteer, the loss is computed against each annotation set separately. Average of the three loss values is taken as the final loss value for a training example.

M2: For the second model, predictions from the re-labeling model described in Section 4.4 are used as labels. This model generates probabilities for both the fine and coarse labels, including the *uncertain* fine labels.

Both the models use identical input data representation and employ the same data augmentation techniques (mentioned in Section 4.3). They also use Binary Cross-entropy loss as the optimization metric. The models are trained on the *train set* using the *validate set* to determine the stopping point.

Training was done on PyTorch [33]. AMSGrad variant of the Adam algorithm [34, 35] with a learning rate of 1e-3 was utilized for optimization. Whenever the loss on *validate* set stopped improving for five *epochs*, the learning rate was reduced by a factor of 10. Regularization in the form of Early stopping was used to prevent overfitting [36]. At the time of prediction, test-time augmentation (TTA) in the form of Time shifting was used.

6. RESULTS

The baseline system mentioned on the task page [17] computes VGGish embeddings [13] of the audio files and builds a multi-label

[3]https://www.kaggle.com/daisukelab/fat2019_prep_mels1

	Fine-level prediction			Coarse-level prediction		
	Macro AUPRC	Micro F1	Micro AUPRC	Macro AUPRC	Micro F1	Micro AUPRC
Baseline	0.531	0.450	0.619	0.619	0.664	0.742
M1	0.645	0.484	0.751	0.718	0.631	0.860
M2	0.622	0.575	0.721	0.723	0.745	0.847

Table 6: Performance on the private *evaluation set*

	Coarse-level prediction			Fine-level prediction		
	Baseline	M1	M2	Baseline	M1	M2
Engine	0.832	0.888	0.878	0.638	0.665	0.673
Machinery impact	0.454	0.627	0.578	0.539	0.718	0.604
Non-machinery impact	0.170	0.361	0.344	0.182	0.362	0.374
Powered saw	0.709	0.684	0.643	0.478	0.486	0.378
Alert signal	0.727	0.897	0.875	0.543	0.858	0.832
Music	0.246	0.404	0.586	0.168	0.289	0.351
Human voice	0.886	0.947	0.949	0.777	0.841	0.833
Dog	0.929	0.937	0.931	0.922	0.936	0.931

Table 7: Class-wise AUPRC on the private *evaluation set*

logistic regression model on top of the embeddings. For this baseline system, a label for an audio recording is considered positive if at least one annotator has labeled the audio clip with that tag. Table 6 shows the performance of the baseline system compared against the proposed models on the private *evaluation set*. The proposed models[4] exhibit improved Micro-AUPRC values for both *fine-grained* and *coarse-grained* labels when compared against the baseline model. Moreover, it can be observed that re-labeling didn't prove effective; it helped improve the Micro-F1 score significantly, but it didn't help raise Micro-AUPRC or Macro-AUPRC.

Class-wise AUPRC performance is reported in Table 7. The modified MobileNetV2 architecture improves over the Baseline model performance for all classes (except one) at both coarse and fine-level prediction. In the case of coarse-level prediction, the AUPRC performance for "Powered saw" is lesser than that of Baseline.

7. CONCLUSIONS AND FUTURE DIRECTIONS

In this paper, we presented our solution to Task 5 (Urban Sound Tagging) of the DCASE 2019 challenge. Our approach involved using a pre-trained image classification model and modifying it for audio classification. We also employed data augmentation techniques to help with the training process. This resulted in our model achieving Micro-AUPRC values of 0.751 and 0.860 on Fine and Coarse tags, respectively thus obtaining the first rank on the leaderboard. We thus demonstrated that impressive gains could be made when compared to using audio embeddings, even in a low-resource scenario such as the one presented here.

As noted in [37], AUPRC only partially correlates with cross-entropy, i.e., decrease in Binary cross-entropy loss may not always result in increase in AUPRC. Exploring loss functions that are more related to AUPRC metric is an avenue for improvement. Depending on the type of class to be predicted, different input representations (such as STFT, HPSS, Log-Mel) might be better [38]. Thus, an ensemble model that uses these different representations can surpass the one proposed in this paper. This ensemble can also involve models that use VGGish or L^3-Net embeddings.

[4]https://github.com/sainathadapa/urban-sound-tagging

8. REFERENCES

[1] O. Houix, G. Lemaitre, N. Misdariis, P. Susini, and I. Urdapilleta, "A lexical analysis of environmental sound categories." *Journal of Experimental Psychology: Applied*, vol. 18, no. 1, p. 52, 2012.

[2] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[3] M. Lasseck, "Acoustic bird detection with deep convolutional neural networks," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 143–147.

[4] I.-Y. Jeong and H. Lim, "Audio tagging system using densely connected convolutional networks," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 197–201.

[5] O. Akiyama and J. Sato, "Multitask learning and semi-supervised learning with noisy data for audio tagging," DCASE2019 Challenge, Tech. Rep., June 2019.

[6] H. Chen, Z. Liu, Z. Liu, P. Zhang, and Y. Yan, "Integrating the data augmentation scheme with various classifiers for acoustic scene modeling," DCASE2019 Challenge, Tech. Rep., June 2019.

[7] Z. Zhang, S. Xu, T. Qiao, S. Zhang, and S. Cao, "Attention based convolutional recurrent neural network for environmental sound classification," *arXiv preprint arXiv:1907.02230*, 2019.

[8] L. Wyse, "Audio spectrogram representations for processing with convolutional neural networks," *arXiv preprint arXiv:1706.09559*, 2017.

[9] J. Salamon and J. P. Bello, "Feature learning with deep scattering for urban sound analysis," in *2015 23rd European Signal Processing Conference (EUSIPCO)*. IEEE, 2015, pp. 724–728.

[10] S. Abdoli, P. Cardinal, and A. L. Koerich, "End-to-end environmental sound classification using a 1d convolutional neural network," *Expert Systems with Applications*, 2019.

[11] E. Fonseca, M. Plakal, D. P. Ellis, F. Font, X. Favory, and X. Serra, "Learning sound event classifiers from web audio with noisy labels," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 21–25.

[12] J. Pons, J. Serrà, and X. Serra, "Training neural audio classifiers with few data," *ArXiv*, vol. 1810.10274, 2018.

[13] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017, pp. 131–135.

[14] Y. Aytar, C. Vondrick, and A. Torralba, “Soundnet: Learning sound representations from unlabeled video,” in *Advances in neural information processing systems*, 2016, pp. 892–900.

[15] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, “Look, listen, and learn more: Design choices for deep audio embeddings,” in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3852–3856.

[16] J. P. Bello, C. Silva, O. Nov, R. L. Dubois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, “Sonyc: A system for monitoring, analyzing, and mitigating urban noise pollution,” *Communications of the ACM*, vol. 62, no. 2, pp. 68–77, Feb 2019.

[17] http://dcase.community/challenge2019/task-urban-sound-tagging.

[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[19] D. Eigen, J. Rolfe, R. Fergus, and Y. LeCun, “Understanding deep architectures using a recursive convolutional network,” *arXiv preprint arXiv:1312.1847*, 2013.

[20] F. Seide, G. Li, and D. Yu, “Conversational speech transcription using context-dependent deep neural networks,” in *Twelfth annual conference of the international speech communication association*, 2011.

[21] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, “Imagenet large scale visual recognition challenge,” *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[23] https://github.com/tensorflow/models/blob/master/research/slim/nets/mobilenet/README.md.

[24] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[26] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *European conference on computer vision*. Springer, 2014, pp. 818–833.

[27] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[28] M. Tan and Q. V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” *arXiv preprint arXiv:1905.11946*, 2019.

[29] B. McFee, M. McVicar, S. Balke, V. Lostanlen, C. Thom, C. Raffel, D. Lee, K. Lee, O. Nieto, F. Zalkow, D. Ellis, E. Battenberg, R. Yamamoto, J. Moore, Z. Wei, R. Bittner, K. Choi, nullmightybofo, P. Friesch, F.-R. Stter, Thassilo, M. Vollrath, S. K. Golu, nehz, S. Waloschek, Seth, R. Naktinis, D. Repetto, C. F. Hawthorne, and C. Carr, “librosa/librosa: 0.6.3,” Feb. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2564164

[30] E. K. V. I. I. A. Buslaev, A. Parinov and A. A. Kalinin, “Albumentations: fast and flexible image augmentations,” *ArXiv e-prints*, 2018.

[31] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, “Random erasing data augmentation,” *arXiv preprint arXiv:1708.04896*, 2017.

[32] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” *arXiv preprint arXiv:1710.09412*, 2017.

[33] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in PyTorch,” in *NIPS Autodiff Workshop*, 2017.

[34] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[35] S. J. Reddi, S. Kale, and S. Kumar, “On the convergence of adam and beyond,” *arXiv preprint arXiv:1904.09237*, 2019.

[36] L. Prechelt, “Early stopping-but when?” in *Neural Networks: Tricks of the trade*. Springer, 1998, pp. 55–69.

[37] C. Gousseau, “VGG CNN for urban sound tagging,” DCASE2019 Challenge, Tech. Rep., September 2019.

[38] J. Bai and C. Chen, “Urban sound tagging with multi-feature fusion system,” DCASE2019 Challenge, Tech. Rep., September 2019.

A MULTI-ROOM REVERBERANT DATASET FOR SOUND EVENT LOCALIZATION AND DETECTION

Sharath Adavanne, Archontis Politis, and Tuomas Virtanen

Audio Research Group, Tampere University, Finland

ABSTRACT

This paper presents the sound event localization and detection (SELD) task setup for the DCASE 2019 challenge. The goal of the SELD task is to detect the temporal activities of a known set of sound event classes, and further localize them in space when active. As part of the challenge, a synthesized dataset where each sound event associated with a spatial coordinate represented using azimuth and elevation angles is provided. These sound events are spatialized using real-life impulse responses collected at multiple spatial coordinates in five different rooms with varying dimensions and material properties. A baseline SELD method employing a convolutional recurrent neural network is used to generate benchmark scores for this reverberant dataset. The benchmark scores are obtained using the recommended cross-validation setup.

Index Terms— Sound event localization and detection, sound event detection, direction of arrival, deep neural networks

1. INTRODUCTION

The goals of the sound event localization and detection (SELD) task includes recognizing a known set of sound event classes such as 'dog bark', 'bird call', and 'human speech' in the acoustic scene, detecting their individual onset and offset times, and further localizing them in space when active. Such a SELD method can automatically describe human and social activities with a spatial dimension, and help machines to interact with the world more seamlessly. Specifically, SELD can be an important module in assisted listening systems, scene information visualization systems, immersive interactive media, and spatial machine cognition for scene-based deployment of services.

The number of existing methods for the SELD task are limited [1–5] providing ample research opportunity. Thus to promote the research and development of SELD methods we propose to organize the SELD task at DCASE 2019[1]. Previously, the SELD task has been treated as two standalone tasks of sound event detection (SED) and direction of arrival (DOA) estimation [1]. The SED in [1] was performed using a classifier-based on Gaussian mixture model - hidden Markov model, and the DOA estimation using the steered response power (SRP). In the presence of multiple overlapping sound events, this approach resulted in the data association problem of assigning the individual sound events detected in a time-frame to their corresponding DOA locations. This data association problem was overcome in [2] by using a sound-model-based localization instead of the SRP method.

Recent methods have proposed to jointly learn the SELD subtasks of SED and DOA estimation using deep neural networks (DNN). Based on the DOA estimation approach, these methods can be broadly categorized into classification [3] and regression approaches [5]. The classification approaches estimate a discrete set of angles, whereas the regression approaches estimate continuous angles. As the classification approach, Hirvonen [3] employed a convolutional neural network and treated SELD as a multiclass-multilabel classification task. Power spectrograms extracted from multichannel audio were used as the acoustic feature and mapped to two sound classes at eight different azimuth angles. Formally, the SELD task was performed by learning an acoustic model, parameterized by parameters $\mathbf{W}$, that estimates the probability of each sound class to be active at a certain time-frame, and discrete spatial angle $P(\mathbf{Y}|\mathbf{X},\mathbf{W})$, where $\mathbf{X} \in \mathbb{R}^{K\times T\times F}$ is the frame-wise acoustic feature for each of the K channels of audio with feature-length F, and number of time-frames T. $\mathbf{Y} \in \mathbb{R}^{T\times C\times U}$ is the class-wise SELD probabilities for C sound classes and U number of azimuth angles. The SELD activity could then be obtained from the class-wise probabilities $\mathbf{Y}$ by applying a binary threshold. Finally, the onset and offset times of the individual sound event classes, and their respective azimuth locations could be obtained from the presence of prediction in consecutive time-frames.

As the regression approach, we recently proposed a convolutional recurrent neural network, SELDnet [5], that was shown to perform significantly better than [3]. In terms of acoustic features $\mathbf{X}$, the SELDnet employed the naive phase and magnitude components of the spectrogram, thereby avoiding any task- or method-specific feature extraction. These features were mapped to two outputs using a joint acoustic model $\mathbf{W}$. As the first output, SED was performed as a multiclass-multilabel classification by estimating the class-wise probabilities $\mathbf{Y}_{\text{SED}} \in \mathbb{R}^{T\times C}$ as $P(\mathbf{Y}_{\text{SED}}|\mathbf{X},\mathbf{W})$. The second output, DOA estimation was performed as multioutput regression task by estimating directly the L dimensional spatial location as $\mathbf{Y}_{\text{DOA}} \in \mathbb{R}^{T\times L\times G\times C}$ for each of the C classes as $f_{\mathbf{W}} : \mathbf{X} \mapsto \mathbf{Y}_{\text{DOA}}$. At each time-frame, G spatial coordinates are estimated per sound class and can be chosen based on the complexity of the sound scene and recording array setup capabilities. In [5], one trajectory was estimated per sound class ($G = 1$), and the respective DOA was represented using its 3D Cartesian coordinates along x, y, and z axes ($L = 3$).

The two DNN-based approaches for SELD, i.e., classification [3] and regression approach [5], have their respective advantages and restrictions. For instance, the resolution of DOA estimation in a classification approach is limited to the fixed set of angles used during training, and the performance on unseen DOA values is unknown. For datasets with a higher number of sound classes and DOA angles, the number of output nodes of the classifier increases rapidly. Training such a large multilabel classifier, where the training labels per frame have a few positives classes representing active sound class and location in comparison to a large number of negative classes, poses problems of imbalanced dataset train-

This work has received funding from the European Research Council under the ERC Grant Agreement 637422 EVERYSOUND.

[1]`http://dcase.community/challenge2019/task-sound-event-localization-and-detection`

https://doi.org/10.33682/1xwd-5v76

ing. Additionally, such a large output classifier requires a larger dataset to have sufficient examples for each class. On the other hand, the regression approach performs seamlessly on unseen DOA values, does not face the imbalanced dataset problems, and can learn from smaller datasets. As discussed earlier, algorithmically the two approaches can potentially recognize multiple instances of the same sound class occurring simultaneously, but such a scenario has never been evaluated and hence their performances are unknown. Additionally, it was observed that the DOA estimation of the classification approach for seen locations was more accurate than the regression approach. This was concluded to be a result of incomplete learning of the regression mapping function due to the small size of dataset [5].

Data-driven SELD approaches [3,5] require sufficient data with annotation of sound event activities and their spatial location. Annotating such a real-life recording to produce a large dataset is a tedious task. One of the ways to overcome this is to develop methods that can learn to perform real-life SELD from a large synthesized dataset and a smaller real-life dataset. The performance of such methods is directly related to the similarity of the synthesized dataset to the real-life sound scene. In [5] we proposed to create such realistic sound scene by convolving real-life impulse responses with real-life isolated sound events, and summing them with real-life ambient sound. These sound scenes were created to have both isolated, and overlapping sound events. The ambient sound was added to the recording at different signal-to-noise ratios (SNRs) to simulate varying real-life conditions. However, all the impulse responses for the dataset in [5] were collected from a single environment. Learning real-life SELD with such restricted dataset is difficult. One of the approaches to overcome this is to train the methods with larger acoustic variability in the training data. In this regard, for the DCASE 2019 SELD task, we employ impulse responses collected from five different environments with varying room dimensions and reverberant properties. Additionally, in order to support research focused on specific audio formats, we provide an identical dataset in two formats of four-channels each: first-order Ambisonics and microphone array recordings.

To summarize, we propose the SELD task for the DCASE 2019 challenge to promote SELD research. We present a challenging multi-room reverberant dataset with varying numbers of overlapping sound events, and a fixed evaluation setup to compare the performance of different methods. As the benchmark, we provide a modified version of the recently proposed SELDnet[2] [5] and report the results on the multi-room reverberant dataset.

2. MULTI-ROOM REVERBERANT DATASET

The SELD task in DCASE 2019 provides two datasets, TAU Spatial Sound Events 2019 - Ambisonic (FOA) and TAU Spatial Sound Events 2019 - Microphone Array (MIC), of identical sound scenes with the only difference in the format of the audio. The FOA dataset provides four-channel First-Order Ambisonic recordings while the MIC dataset provides four-channel directional microphone recordings from a tetrahedral array configuration. Both formats are extracted from the same microphone array. The SELD methods can be developed on either one of the two or both the datasets to exploit their mutual information. Both the datasets, consists of a development[3] and evaluation[4] set. The development set consists of 400 one-minute recordings sampled at 48000 Hz, divided into four cross-validation splits of 100 recordings each. The evaluation set consists of 100 one-minute recordings. These recordings were synthesized using spatial room impulse response (IRs) collected from five indoor environments, at 504 unique combinations of azimuth-elevation-distance. In order to synthesize these recordings, the collected IRs were convolved with isolated sound events from DCASE 2016 task 2. Additionally, half the number of recordings have up to two temporally overlapping sound events, and the remaining have no overlapping. Finally, to create a realistic sound scene recording, natural ambient noise collected in the IR recording environments was added to the synthesized recordings such that the average SNR of the sound events was 30 dB. The only explicit difference between each of the development dataset splits and evaluation dataset is the isolated sound event examples employed.

2.1. Real-life Impulse Response Collection

The real-life IR recordings were collected using an Eigenmike spherical microphone array. A Genelec G Two loudspeaker was used to playback a maximum length sequence (MLS) around the Eigenmike. The MLS playback level was ensured to be 30 dB greater than the ambient sound level during the recording. The IRs were obtained in the STFT domain using a least-squares regression between the known measurement signal (MLS) and far-field recording independently at each frequency. These IRs were collected in the following directions: a) 36 IRs at every 10° azimuth angle, for 9 elevations from -40° to 40° at 10° increments, at 1 m distance from the Eigenmike, resulting in 324 discrete DOAs. b) 36 IRs at every 10° azimuth angle, for 5 elevations from -20° to 20° at 10° increments, at 2 m distance from the Eigenmike, resulting in 180 discrete DOAs. The IRs were recorded at five different indoor environments inside the Tampere University campus at Hervanta, Finland during non-office hours. These environments had varying room dimensions, furniture, flooring and roof materials. Additionally, we also collected 30 minutes of ambient noise recordings from these five environments with the IR recording setup unchanged during office hours thereby obtaining realistic ambient noise. We refer the readers to the DCASE 2019 challenge webpage for description on individual environments.

2.2. Dataset Synthesis

The isolated sound events dataset[5] from DCASE 2016 task 2 consists of 11 classes, each with 20 examples. These examples were randomly split into five sets with an equal number of examples for each class; the first four sets were used to synthesize the four splits of the development dataset, while the remaining one set was used for the evaluation dataset. Each of the one-minute recordings were generated by convolving randomly chosen sound event examples with a corresponding random IR to spatially position them at a given distance, azimuth and elevation angles. The IRs chosen for each recording are all from the same environment. Further, these spatialized sound events were temporally positioned using randomly chosen start times following the maximum number of overlapping sound events criterion. Finally, ambient noise collected at the respective IR environment was added to the synthesized recording such that the average SNR of the sound events is 30 dB.

Since the number of channels in the IRs is equal to the number of microphones in Eigenmike (32), in order to create the MIC

[2]https://github.com/sharathadavanne/seld-dcase2019

[3]https://doi.org/10.5281/zenodo.2599196

[4]https://doi.org/10.5281/zenodo.3377088

[5]http://dcase.community/challenge2016/task-sound-event-detection-in-synthetic-audio

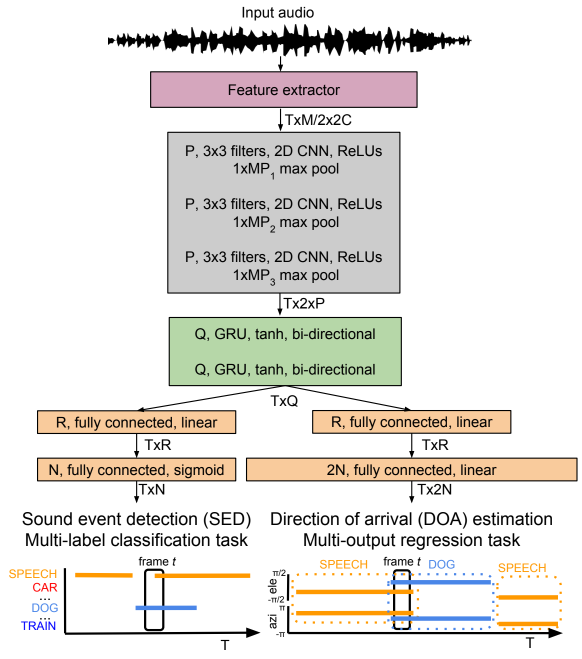

Input audio
Feature extractor
TxM/2x2C
P, 3x3 filters, 2D CNN, ReLUs
1xMP1 max pool
P, 3x3 filters, 2D CNN, ReLUs
1xMP2 max pool
P, 3x3 filters, 2D CNN, ReLUs
1xMP3 max pool
Tx2xP
Q, GRU, tanh, bi-directional
Q, GRU, tanh, bi-directional
TxQ
R, fully connected, linear
R, fully connected, linear
TxR
TxR
N, fully connected, sigmoid
2N, fully connected, linear
TxN
Tx2N
Sound event detection (SED)
Multi-label classification task
Direction of arrival (DOA) estimation
Multi-output regression task
SPEECH
CAR
DOG
TRAIN
frame t
T
azi
ele
SPEECH
DOG
SPEECH

Figure 1: Convolutional recurrent neural network for SELD.

dataset we select four microphones that have a nearly-uniform tetrahedral coverage of the sphere. Those are the channels 6, 10, 26, and 22 that corresponds to microphone positions (45°, 35°, 4.2 cm), (-45°, -35°, 4.2 cm), (135°, -35°, 4.2 cm) and (-135°, 35°, 4.2 cm). The spherical coordinate system in use is right-handed with the front at (0°, 0°), left at (90°, 0°) and top at (0°, 90°). Finally, the FOA dataset is obtained by converting the 32-channel microphone signals to the first-order Ambisonics format, by means of encoding filters based on anechoic measurements of the Eigenmike array response, generated with the methods detailed in [6].

2.3. Array Response

For model-based localization approaches, the array response may be considered known. The following theoretical spatial responses (steering vectors) modeling the two formats describe the directional response of each channel to a source incident from DOA given by azimuth angle ϕ and elevation angle θ.

For the FOA format, the array response is given by the real orthonormalized spherical harmonics:

$$H_1(\phi,\theta,f) = 1 \quad (1)$$
$$H_2(\phi,\theta,f) = \sqrt{3} * \sin(\phi) * \cos(\theta) \quad (2)$$
$$H_3(\phi,\theta,f) = \sqrt{3} * \sin(\theta) \quad (3)$$
$$H_4(\phi,\theta,f) = \sqrt{3} * \cos(\phi) * \cos(\theta). \quad (4)$$

For the tetrahedral array of microphones mounted on spherical baffle, similar to Eigenmike, an analytical expression for the directional array response is given by the expansion:

$$H_m(\phi_m,\theta_m,\phi,\theta,\omega) = \frac{1}{(\omega R/c)^2}\sum_{n=0}^{30}\frac{\mathrm{i}^{n-1}}{h_n'^{(2)}(\omega R/c)}(2n+1)P_n(\cos(\gamma_m)), \quad (5)$$

where m is the channel number, (ϕ_m, θ_m) are the specific microphone's azimuth and elevation position, $\omega = 2\pi f$ is the angular frequency, $R = 0.042$ m is the array radius, $c = 343$ m/s is the speed of sound, $\cos(\gamma_m)$ is the cosine angle between the microphone position and the DOA, P_n is the unnormalized Legendre polynomial of degree n, and $h_n'^{(2)}$ is the derivative with respect to the argument of a spherical Hankel function of the second kind. The expansion is limited to 30 terms which provide negligible modeling error up to 20 kHz. Note that the Ambisonics format is frequency-independent, something that holds true to about 9 kHz for Eigenmike and deviates gradually from the ideal response for higher frequencies.

Table 1: Cross-validation setup

Folds	Splits		
	Training	Validation	Testing
Fold 1	3, 4	2	1
Fold 2	4, 1	3	2
Fold 3	1, 2	4	3
Fold 4	2, 3	1	4

3. BASELINE METHOD

As the benchmark method, we employ the SELDnet [5]. Contrary to [5], where the DOA estimation is performed as multi-output regression of the 3D Cartesian DOA vector components $x, y, z \in [-1, 1]$, in this benchmark implementation we directly estimate the azimuth $\phi \in [-\pi, \pi]$ and elevation $\theta \in [-\pi/2, \pi/2]$ angles. Accordingly, the activation function of the DOA estimation layer is changed from tanh to linear. The remaining architecture remains unchanged and is illustrated in Figure 1. The input to the method is a multichannel audio of 48 kHz sampling rate, from which the phase and magnitude components of the spectrogram are extracted using 2048-point discrete Fourier transform from 40 ms length Hanning window and 20 ms hop length. A sequence of T spectrogram frames (T = 128) is then fed to the three convolutional layers that extract shift-invariant features using 64 filters each. Batch normalization is used after each convolutional layer. Dimensionality reduction of the input spectrogram feature is performed using max pooling operation only along the frequency axis. The temporal axis is untouched to keep the resolution of the output unchanged from the input dimension.

The temporal structure of the sound events is modeled using two bidirectional recurrent layers with 128 gated recurrent units (GRU) each. Finally, the output of the recurrent layer is shared between two fully-connected layer branches each producing the SED as multiclass multilabel classification and DOA as multi-output regression; together producing the SELD output. The SED output obtained is the class-wise probabilities for the C classes in the dataset at each of the T frames of input spectrogram sequence, resulting in a dimension of $T \times C$. The localization output estimates one single DOA for each of the C classes at every time-frame T, i.e., if multiple instances of the same sound class occur in a time frame the SELDnet localizes either one or oscillates between multiple instances. The overall dimension of localization output is $T \times 2C$, where $2C$ represents the class-wise azimuth and elevation angles. A sound event class is said to be active if its probability in SED output is greater than the threshold of 0.5, otherwise, the sound class is considered to be absent. The presence of sound class in consecutive time-frames gives the onset and offset times, and the corresponding DOA estimates from the localization output gives the spatial location with respect to time.

A cross-entropy loss is employed for detection output, while a mean square error loss on the spherical distance between reference and estimated locations is employed for the localization output. The combined convolutional recurrent neural network architecture is trained using Adam optimizer and a weighted combination of the two output losses. Specifically, the localization output is weighted 50 times more than the detection output.

4. EVALUATION

4.1. Evaluation Setup

The development dataset consists of four cross-validation splits as shown in Table 1. Participants are required to report the performance of their method on the testing splits of the four folds. The performance metrics are calculated by accumulating the required statistics from all the folds [7], and not as the average of the metrics of the individual folds. For the evaluation dataset, participants are allowed to decide the training procedure, i.e. the amount of training and validation files in the development dataset and the number of ensemble models.

4.2. Metrics

The SELD task is evaluated with individual metrics for SED and DOA estimation. For SED, we use the F-score and error rate (ER) calculated in one-second segments [8]. For DOA estimation we use two frame-wise metrics [9]: DOA error and frame recall. The DOA error is the average angular error in degrees between the predicted and reference DOAs. For a recording of length T time frames, let $\mathbf{DOA}_R^t$ be the list of all reference DOAs at time-frame t and $\mathbf{DOA}_E^t$ be the list of all estimated DOAs. The DOA error is now defined as

$$DOA\,error = \frac{1}{\sum_{t=1}^{T} D_E^t} \sum_{t=1}^{T} \mathcal{H}(\mathbf{DOA}_R^t, \mathbf{DOA}_E^t), \quad (6)$$

where D_E^t is the number of DOAs in $\mathbf{DOA}_E^t$ at t-th frame, and $\mathcal{H}$ is the Hungarian algorithm for solving assignment problem, i.e., matching the individual estimated DOAs with the respective reference DOAs. The Hungarian algorithm solves this by estimating the pair-wise costs between individual predicted and reference DOA using the spherical distance between them, $\sigma = \arccos(\sin\lambda_E \sin\lambda_R + \cos\lambda_E \cos\lambda_R \cos(|\phi_R - \phi_E|))$, where the reference DOA is represented by the azimuth angle $\phi_R \in [-\pi, \pi)$ and elevation angle $\lambda_R \in [-\pi/2, \pi/2]$, and the estimated DOA is represented with (ϕ_E, λ_E) in the similar range as reference DOA.

In order to account for time frames where the number of estimated and reference DOAs are unequal, we report a frame recall type metric, which is calculated as, $Frame\,recall = \sum_{t=1}^{T} \mathbb{1}(D_R^t = D_E^t)/T$, with D_R^t the number of DOAs in $\mathbf{DOA}_R^t$ at t-th frame, $\mathbb{1}()$ the indicator function returning one if the $(D_R^t = D_E^t)$ condition is met and zero otherwise. The submitted methods will be ranked individually for all four metrics of SED and DOA estimation, and the final positions will be obtained using the cumulative minimum of the ranks.

Table 2: Evaluation scores for cross-validation folds.

	FOA				MIC			
Fold	ER	F	DE	FR	ER	F	DE	FR
1	**0.25**	**85.0**	30.4	**86.6**	0.32	81.9	32.0	84.5
2	0.39	77.0	**25.9**	85.0	0.37	79.0	30.5	83.1
3	0.30	83.1	27.9	**86.6**	**0.29**	**83.4**	30.9	**85.4**
4	0.42	74.7	30.2	83.3	0.42	76.2	**29.8**	83.1

5. RESULTS

The results obtained with the SELDnet for different folds of the development dataset are presented in Table 2. Although the folds are identical for the FOA and MIC datasets, the SELDnet is observed to perform better on fold 1 for FOA and fold 3 for MIC datasets. This suggests that the spectral and spatial information in the two formats are not identical and potentially methods can benefit from mutual information from the two datasets.

The overall results with respect to the different number of overlapping sound events and different reverberant environments are presented in Table 3. It is observed that the general performance of SELDnet on FOA format is marginally better than MIC for both development and evaluation datasets. The SELDnet is seen to perform better when there is no polyphony across datasets. Finally, the SELDnet trained with all five environments is seen to perform the best in the first environment, across datasets. This environment had carpet flooring and multiple cushioned furniture that is known to reduce the overall reverberation, and hence resulted in a better SELD performance in comparison to the other four environments.

6. CONCLUSION

In this paper, we proposed the sound event localization and detection (SELD) task for the DCASE 2019 challenge to promote SELD research. An acoustically challenging multi-room reverberant dataset is provided for the task. This dataset is synthesized with isolated sound events that are spatially positioned using real-life impulse responses collected from five-different rooms that have different acoustic properties. Additionally, in order to support research focused on specific audio-formats, the dataset provides Ambisonic and microphone array recordings of identical sound scenes, that are of four-channels each. Further, the dataset provides a pre-defined four-fold cross-validation split for evaluating the performance of competing methods. As the baseline results for the dataset, we report the performance of a benchmark SELD method based on convolutional recurrent neural network.

Table 3: SELDnet performance on overlapping sound events and reverberant scenes. ID - identifier, DE - DOA Error, FR - Frame Recall

		Development dataset scores								Evaluation dataset scores							
		FOA				MIC				FOA				MIC			
	ID	ER	F	DE	FR	ER	F	DE	FR	ER	F	DE	FR	ER	F	DE	FR
Overlap	1	**0.32**	**82.2**	**23.1**	**93.0**	**0.35**	**81.2**	**25.9**	**92.3**	**0.26**	**87.3**	**18.5**	**92.9**	**0.27**	**86.2**	**33.5**	**92.4**
	2	0.35	78.6	31.6	77.8	**0.35**	79.1	33.6	75.8	0.29	84.3	28.0	78.5	0.31	81.6	40.7	74.4
Indoor environment	1	**0.30**	**81.6**	**28.3**	85.3	**0.33**	80.2	30.4	83.9	**0.24**	**87.5**	24.2	**87.6**	**0.27**	**84.3**	**37.5**	81.5
	2	0.38	78.6	28.5	**86.7**	0.36	80.2	30.7	**86.2**	0.41	80.3	**23.3**	79.2	0.33	82.0	39.0	**85.1**
	3	0.33	80.0	28.7	84.3	0.35	79.9	30.7	82.5	0.26	85.3	25.4	85.8	0.29	83.5	37.4	82.7
	4	0.37	79.3	**28.3**	84.9	0.35	**80.4**	**30.3**	83.7	0.27	86.1	24.0	87.1	0.31	82.2	36.4	83.3
	5	0.34	80.0	29.0	85.7	0.36	79.5	31.9	83.4	**0.24**	87.4	26.0	88.9	0.28	84.1	40.1	84.3
Total		0.34	79.9	28.5	85.4	0.35	80.0	30.8	84.0	0.28	85.4	24.6	85.7	0.30	83.2	38.1	83.3

7. REFERENCES

[1] T. Butko, F. G. Pla, C. Segura, C. Nadeu, and J. Hernando, “Two-source acoustic event detection and localization: Online implementation in a smart-room,” in *European Signal Processing Conference*, 2011.

[2] R. Chakraborty and C. Nadeu, “Sound-model-based acoustic source localization using distributed microphone arrays,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014.

[3] T. Hirvonen, “Classification of spatial audio location and content using convolutional neural networks,” in *Audio Engineering Society Convention 138*, 2015.

[4] K. Lopatka, J. Kotus, and A. Czyzewsk, “Detection, classification and localization of acoustic events in the presence of background noise for acoustic surveillance of hazardous situations,” *Multimedia Tools and Applications Journal*, vol. 75, no. 17, 2016.

[5] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, “Sound event localization and detection of overlapping sources using convolutional recurrent neural networks,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, March 2019.

[6] A. Politis and H. Gamper, “Comparing modeled and measurement-based spherical harmonic encoding filters for spherical microphone arrays,” in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 224–228.

[7] G. Forman and M. Scholz, “Apples-to-apples in cross-validation studies: Pitfalls in classifier performance measurement,” *SIGKDD Explor. Newsl.*, vol. 12, no. 1, pp. 49–57, November 2010. [Online]. Available: http://doi.acm.org/10.1145/1882471.1882479

[8] A. Mesaros, T. Heittola, and T. Virtanen, “Metrics for polyphonic sound event detection,” in *Applied Sciences*, vol. 6, no. 6, 2016.

[9] S. Adavanne, A. Politis, and T. Virtanen, “Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network,” in *European Signal Processing Conference*, 2018.

SOUND EVENT CLASSIFICATION AND DETECTION WITH WEAKLY LABELED DATA

Sharath Adavanne[*], *Haytham M. Fayek, and Vladimir Tourbabin*

Facebook Reality Labs, Redmond, WA, USA

ABSTRACT

The Sound Event Classification (SEC) task involves recognizing the set of active sound events in an audio recording. The Sound Event Detection (SED) task involves, in addition to SEC, detecting the temporal onset and offset of every sound event in an audio recording. Generally, SEC and SED are treated as supervised classification tasks that require labeled datasets. SEC only requires weak labels, i.e., annotation of active sound events, without the temporal information, whereas SED requires strong labels, i.e., annotation of the onset and offset times of every sound event, which makes annotation for SED more tedious than for SEC. In this paper, we propose two methods for joint SEC and SED using weakly labeled data: a Fully Convolutional Network (FCN) and a novel method that combines a Convolutional Neural Network with an attention layer (CNNatt). Unlike most prior work, the proposed methods do not assume that the weak labels are active during the entire recording and can scale to large datasets. We report state-of-the-art SEC results obtained with the largest weakly labeled dataset — Audioset.

***Index Terms*—** Convolutional neural network, sound classification, sound event detection, weakly supervised learning

1. INTRODUCTION

Sound Event Classification (SEC) is the task of recognizing the set of active sound events in a given audio recording. Additionally, detecting the temporal activity of each sound event, i.e., onset and offset times, is referred to as Sound Event Detection (SED). SEC and SED can be helpful in query based multimedia retrieval [1], acoustic scene analysis [2, 3], and bio-diversity monitoring [4–6]. The SED task requires datasets that are strongly labeled [7–9], i.e., annotation of active sound events and their respective onset and offset times. On the other hand, the SEC task requires weakly labeled datasets, that only provide annotation of the set of active sound events for every recording [5, 10, 11]. In terms of complexity, it is more tedious to annotate strongly labeled datasets than weakly labeled datasets.

The SEC task has traditionally been approached with Convolutional Neural Network (CNN) architectures [5, 10, 12]. Whereas for the SED task, which requires temporal localization of sound events, the joint architecture of CNNs with recurrent neural networks, referred to as Convolutional Recurrent Neural Network (CRNN) [8, 13], has shown consistently good results across SED datasets. Recently, it was shown in [9] that on large SED datasets, the performance of CNN architectures is comparable to CRNN architectures when the detection is happening at one-second resolution. In this paper, we aim to perform SED at a similar resolution using a large dataset. Given that the training time of CNN architectures is relatively faster than comparable CRNN architectures, we focus on CNN architectures.

Recently, methods have been proposed to jointly learn SEC and SED from just the weakly labeled data [14–18], in order to overcome the complexity of annotating strongly labeled datasets. Prior work in [14] used multiple established CNN architectures from the computer vision domain and applied them to this task, but these methods assumed that the weak labels were active throughout the recording during training, and is hereafter referred to as Strong Label Assumption Training (SLAT). This assumption leads to poor SEC performance, as shown in [17]. As an alternative to SLAT, the authors in [16] proposed a Fully Convolutional Network (FCN) based method that enabled learning from the weakly labeled dataset without assuming the presence of weak labels active throughout the recording; such a training approach is hereafter referred to as Weak Label Assumption Training (WLAT). Similar FCN based WLAT methods were also proposed in [17, 19], but all of these methods have only been evaluated on small datasets, and their performance on large datasets is unknown. In this paper, we study the performance of FCN on the largest publicly available dataset — Audioset [20].

In addition to the FCN-based approach, an alternative WLAT method is proposed that combines CNN and an attention layer (CNNatt). The attention layer enables the CNNatt to automatically learn to attend to relevant time segments of the audio during inference. Thus, in the current task, given a weak label, an attention layer can identify the relevant time segments in the audio where the weak label is active, and consequently provide strong labels.

To summarize, we study the performance of FCN for the task of joint SEC and SED from a large weakly labeled dataset, and further propose a novel CNNatt for the same task. The contributions of this paper are as follows. We present, for the first time since the benchmark work in [14], a study using the complete Audioset. Unlike [14], which used SLAT, the two methods in this paper, FCN and CNNatt, use WLAT to jointly perform SEC and SED. Finally, since Audioset provides just the weak labels, we only present the quantitative results for the SEC performance and compare them with the recently published baselines [14, 21, 22]. The SED performance is evaluated subjectively by visualizing the outputs and manual listening inspection.

2. METHOD

The input for the two methods — FCN and CNNatt — is a single channel audio recording. A feature extraction block produces F-band log mel-band energies for each of the T frames of input audio. The feature sequence of dimension $T \times F$ for each recording is then mapped to the C classes (SEC) as a multi-class multi-label classification task. Additionally, as an intermediate output, both studied methods generate frame-wise results for the C classes (SED) of dimension $T_N \times C$. The time-dimensionality of the SED output T_N is smaller than the input T as a result of multiple max pooling operations. During training, only the audio recording and its respective weak label(s) — one-hot encoded — are used. During inference, given an input audio, both methods produce two outputs

[*]This work was performed during an internship at Facebook.

https://doi.org/10.33682/fx8n-cm43

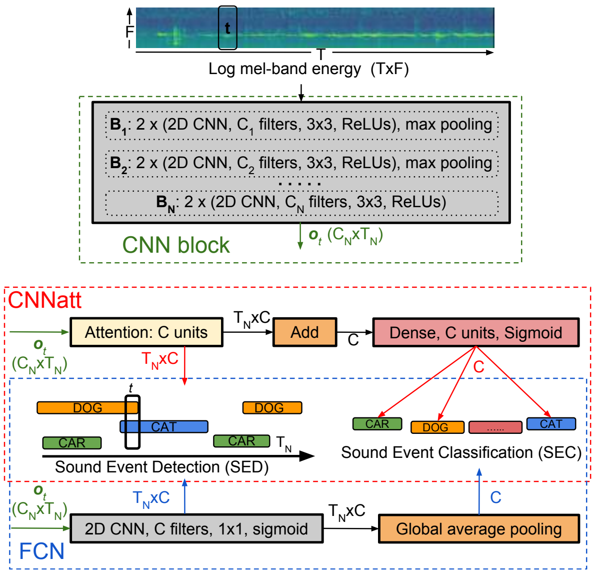

F
t
T
Log mel-band energy (TxF)
B1: 2 x (2D CNN, C1 filters, 3x3, ReLUs), max pooling
B2: 2 x (2D CNN, C2 filters, 3x3, ReLUs), max pooling
BN: 2 x (2D CNN, CN filters, 3x3, ReLUs)
CNN block
ot (CNxTN)
CNNatt
Attention: C units
TNxC
Add
C
Dense, C units, Sigmoid
DOG
CAT
CAR
Sound Event Detection (SED)
Sound Event Classification (SEC)
2D CNN, C filters, 1x1, sigmoid
Global average pooling
FCN

Figure 1: The Fully Convolutional Network (FCN) and Convolutional Neural Network with attention layer (CNNatt) methods for joint learning of SEC and SED from weakly labeled dataset.

in sequence: first, the strong labels (SED), followed by the weak labels (SEC). These outputs are the respective class probabilities in the continuous range of [0, 1]. A value closer to one signifies that the sound class is active, and closer to zero signifies that it is absent. The details of the feature extraction and the two methods studied are presented below.

2.1. Feature Extraction

The log mel-band energy features are extracted for each frame of 1024 samples with 50% overlap using a 1024-point fast Fourier transform. A total of 40 bands are extracted in the frequency range of 0-8000 Hz from an audio recording sampled at 16 kHz. For a 10 s audio input, the feature extraction step produces a sequence of $T = 320$ frames and $F = 40$ features.

2.2. Neural Network

2.2.1. Fully Convolutional Network (FCN)

Figure 1 presents the overall structure of the FCN. The input is a $T \times F$ dimension sequence of the extracted features. The initial layers of the network consist of 2D convolutional layers that learn local shift-invariant features. Each of the convolutional layers has filters with a 3×3 receptive field, with the output dimension kept the same as the input dimension using zero padding. Batch normalization [23] is performed on this output, followed by a Rectified Linear Unit (ReLU) activation and a dropout layer [24]. The audio features dimensionality is reduced by performing max pooling after every second convolutional layer, such that the temporal- and feature-dimensionality in the final convolutional layer N with C_N filters is reduced to T_N and F_N, respectively. In the reduced dimensionality, each frame in T_N represents one second of input audio and $F_N = 1$. These multi-layered convolutional layers are, hereafter, referred to together as the CNN block, and its output $\mathbf{o}_t$ is an embedding of dimension $C_N \times T_N$, as seen in Figure 1.

The embedding from the CNN block is fed to a single 2D convolutional layer with C filters (equal to the number of classes in the dataset), a receptive field of dimension 1×1 and sigmoid activation to support multi-class multi-label classification. Given the CNN block embedding of dimension $C_N \times T_N$, the newly added layer produces SED results of dimension $T_N \times C$. Further, the SEC results are obtained from the SED results by performing a global average pooling across T_N. The FCN was tuned as described in Section 3.4.

2.2.2. Convolutional Neural Network with attention layer (CNNatt)

The overall structure of the CNNatt is shown in Figure 1. Given a feature sequence of $T \times F$ dimension, a CNN block similar to the FCN generates output $\mathbf{o}_t$ of dimension $C_N \times T_N$. This is fed to an attention layer identical to that described in [21, 22]. The attention layer performs the following operation on it:

$$\mathbf{a}_t = cls(\mathbf{o}_t) \odot (atn(\mathbf{o}_t) / \sum_{t=0}^{T} atn(\mathbf{o}_t)), \quad (1)$$

where $\odot$ signifies element-wise multiplication. The $atn()$ function guides the network to be attentive to certain time frames, while the $cls()$ function performs the classification for each input time frame t. The $atn()$ and $cls()$ functions are implemented as 2D convolutional layers with C filters each and a receptive field of dimension 1×1. The $atn()$ function employs a softmax activation, while the $cls()$ function is implemented with a sigmoid activation. The output of the attention layer $\mathbf{a}_t$ produces the frame-wise SED results of dimension $T_N \times C$. Further, the SEC results are obtained from $\mathbf{a}_t$ by adding the activations across T_N and feeding them to a fully connected dense layer with C units and sigmoid activation. The CNNatt was tuned as described in Section 3.4.

The proposed implementation of the two methods enables them to operate on input audio of variable length, but with a minimum length criterion arising from the multiple max pooling operations employed. Both methods were trained for 100 epochs using binary cross entropy loss calculated between the predicted SEC output and the weak labels in the reference annotation of the dataset. As the optimizer, we employ Adam [25], a first-order adaptive variant of stochastic gradient descent, with the parameters introduced in [25], $\alpha = 0.001$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. Early stopping was used during training to avoid overfitting. The training was stopped if the mean Average Precision (mAP) score (see Section 3.2) did not improve for 25 epochs. The methods were implemented using PyTorch and trained in data parallel mode over eight GPUs.

3. EVALUATION

3.1. Dataset

We used the complete dataset, Audioset [20], in this paper. The dataset provides a pre-defined development and evaluation split. At the time of this study, only about 94% of the YouTube videos of Audioset were active. The audio recordings for these videos were pre-processed to have a sampling rate of 16 kHz, and a single channel. Although the two methods are invariant to the length of the input audio, the Audioset recordings used are of a constant length of ten seconds. The complete Audioset has C = 527 classes with a highly imbalanced distribution (see [20] for more details).

3.2. Metrics

The mean Average Precision (mAP) metric is used to evaluate the performance of our methods for SEC, due to class imbalance, similar to that in prior studies on Audioset [20–22]. The mAP is defined as the mean of the area under the precision-recall curve across the C classes,

$$mAP = \frac{1}{C} \sum_{i=1}^{C} \sum_{m=1}^{M} P_{m,i}(R_{m,i} - R_{m-1,i}), \quad (2)$$

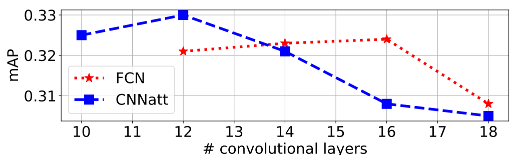


Figure 2: The mAP scores obtained with respect to different number of convolutional layers of FCN and CNNatt.

where $P_{m,i}$ and $R_{m,i}$ are the precision and recall values of class i at M different threshold values.

Finally, since the strong labels of Audioset are unavailable, we only visualize and manually inspect the SED performance.

3.3. Baseline Methods

We compare the performance of the two methods with four baseline methods [14, 21, 22]. Note that the dense method in [14] is the only method that was trained using the complete Audioset dataset; the other three baseline methods [14, 21, 22] study the performance on the Audioset embeddings obtained using a network trained on a dataset larger than Audioset.

The first method proposed in [14] uses log mel-band energy features similar to those used in this paper. A 64-band feature is mapped to 527 classes of Audioset using a multi-layered fully connected dense network with SLAT.

Among the methods using Audioset embeddings, the embeddings used in the second method proposed in [14] are different from the ones used in [21] and [22]. The embeddings in [14] were obtained using a ResNet-50 network and SLAT on a much larger YT-100M dataset (20× larger than Audioset — not available publicly). This network was then used as a feature extractor to generate embeddings from each of the Audioset recordings, hereafter referred to as ResNet embeddings. Finally, the benchmark scores on the ResNet embeddings were obtained using a multi-layered fully connected dense network similar to the one described above.

In comparison, the two recent methods — single attention [21] and multiple attention [22] — use embeddings that were generated using a VGGish network instead of a ResNet-50 and SLAT on a YT-8M dataset instead of a YT-100M dataset. This is referred to as VGGish embeddings hereafter and is publicly available. Unlike YT-100M, the YT-8M dataset is publicly available, but the exact splits used to learn the VGGish embeddings described above are unknown. The single attention method [21] uses multiple layers of a fully connected dense network, followed by an attention layer as the classification layer, whereas the multiple attention method [22] uses multiple attention layers located between fully connected layers, and concatenates the output of these attention layers to perform the final classification.

3.4. Experiments

Hyper-parameter tuning of both methods is performed to identify the best configuration for Audioset. Since the attention and dense layers of CNNatt and the final classification convolutional layer of FCN are dependent on the output number of classes, the only tunable part is the CNN block. In order to restrict the number of possible options to tune, we made sure that the number of filters in a convolutional layer doubles after every two layers. For example, $C_2 = 2C_1$ in Figure 1. The number of layers in the CNN block was tuned randomly [26] in the range of five to twenty with the number of filters in the first layer varying in the set $\in \{16, 32, 64\}$. In order to study the effect of regularization, the dropout layer was tuned in the set $\in \{0, 0.15, 0.3, 0.5, 0.75\}$.

Table 1: The mAP scores on Audioset with different methods

Methods on Audioset recordings	mAP
Random chance	0.005
Dense [14]	0.137
FCN	0.324
CNNatt	**0.330**
Method on Audioset ResNet embeddings*+	
Dense [14]	0.314
Method on Audioset VGGish embeddings*#	
Single attention [21]	0.327
Multiple attention [22]	**0.360**

*Embeddings from network trained on dataset larger than Audioset.
+The YT-100M dataset used to train the ResNet is not publicly available.
#The YT-8M dataset used to train the embeddings network is publicly available, but the exact splits used to produce the embeddings are unknown.

The SEC performance is evaluated on the evaluation split of Audioset and compared with the existing baselines using Audioset recordings [14] and embeddings [14, 21, 22].

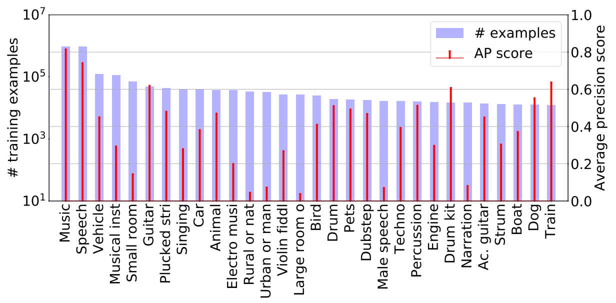
Finally, since the Audioset dataset lacks strong labels, we only perform a subjective analysis of SED through manually listening and visualizing the SED output of the two methods on a subset of Audioset examples.

4. RESULTS AND DISCUSSION

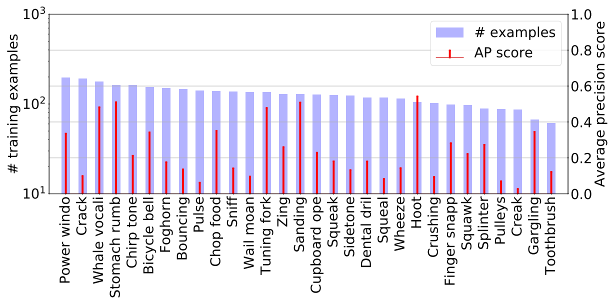
The best configuration for the FCN that obtained the highest mAP score had 16 convolutional layers including the (last) classification convolutional layer, with the first layer having 16 filters. The best CNNatt configuration had 12 convolutional layers in the CNN block, starting with 16 filters in the first layer, and followed by an attention and dense layer. Further, using zero dropout gave the best results for both methods. In terms of the number of parameters, the CNNatt uses only about 20% of the 25M parameters in FCN. The performance for other configurations of FCN and CNNatt when the first convolutional layer had 16 filters is visualized in Figure 2. Here, it can be observed that the CNNatt achieves better mAP scores than FCN with just 10 convolutional layers.

A classifier generating random results on Audioset obtains a mAP score of 0.005, as seen in Table 1. In comparison, the baseline dense method [14] trained on Audioset recordings obtained a mAP score of 0.137. This is a 27× improvement over the random results generating classifier. The FCN improved 2.36× over the dense method [14] and obtained a mAP score of 0.324. In fact, this score is higher than the dense method using ResNet embeddings [14], which obtained a mAP score of 0.314. This suggests that the FCN with WLAT outlearns the ResNet-50 with SLAT on a much larger YT-100M dataset.

The second method, CNNatt, obtained a best mAP score of 0.330. This is a significant result considering that the CNNatt learns to perform SEC better than FCN using only 20% of FCN's parameters. In fact, CNNatt performs better than the single attention method [21] trained using VGGish embeddings obtained from a VGG network and SLAT on a dataset larger than Audioset. This makes CNNatt the state-of-the-art for SEC using the complete Audioset recordings. The class-wise average precision score obtained with CNNatt on the evaluation split, and the corresponding number of examples in the development split is visualized in Figure 3. Among the top 30 frequent classes in Fig 3a we observe that the CNNatt performs better on *sound event* classes (E.g. Speech, Music, Car, Guitar, and Dog), and poorly on *sound scene* classes such



(a) Top 30 frequent classes



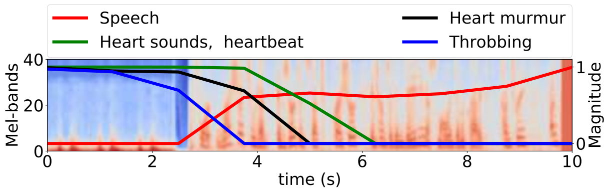
(b) Bottom 30 frequent classes

Figure 3: Visualization of the class-wise number of training examples and the corresponding average precision (AP) scores obtained with the CNNatt.

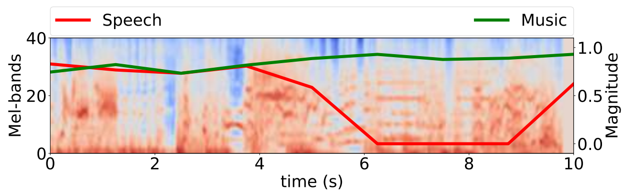
as Small room, Large room, Rural, and Urban classes.

The SED results obtained with the FCN for recordings in the evaluation split of Audioset are visualized in Figures 4a, 4b, 4c, and 4d. For example, in Figure 4a according to the reference annotation, the recording contains the classes: speech and heart murmur. From the spectrogram, we can distinguish a heartbeat-like repetitive structure in the first 2.5 s, and speech beyond 2.5 s. Similar temporal activity is observed in the overlaid class magnitude plot, which shows the SED output from the FCN. To simplify the visualization, we only show the classes whose SED output magnitude is greater than 0.5 throughout the recording. In addition to the heart murmur class, the FCN also recognized the first 2.5 s as heart sounds and throbbing, which are classes that sound similar to a heart murmur. Similarly, in Figure 4b, among the reference classes, the FCN detected the speech and music classes successfully, but missed the bang class (occurs from 1.5 s to 2.1 s) that was part of the music. In Figure 4c, the FCN detected the reference speech and music classes correctly, but missed the oink class (occurs from 6.3 s to 8.2 s), and successfully detected the burping class (first 3 s) that was missing in the reference annotation. Finally, in Figure 4d, the FCN missed the sniff sound class but successfully detected the chewing class that was missing in the reference annotation. Additionally, the FCN also over-predicted the speech class beyond 4 s.

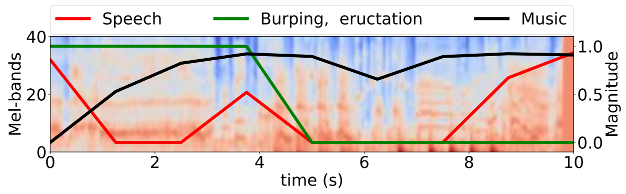
In general, although the FCN missed detection of few short duration and low prior sound classes, we observe from the SED Figures 4a, 4b, 4c, and 4d a good recall of most of these low prior (10^{-5}) sound classes such as chewing, burping, heart sounds, and murmur. Another observation from these figures is that the onset and offset of the sound events are not of high precision. We believe that this is a result of both the dimensionality reduction (max pooling) operation within both methods and the limitation of learning strong labels from a weakly labeled dataset. Similar SED results were observed in all the recordings studied. Further, the SED performance of the CNNatt was comparable to that of the FCN with



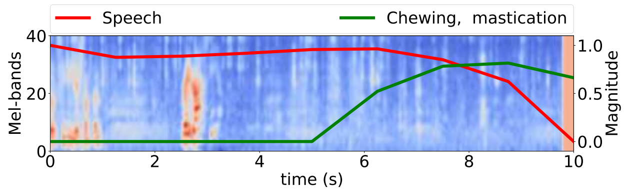
(a) YouTube recording '-1nilez17Dg' at 30 s, with speech and heart murmur sound classes in reference annotation



(b) YouTube recording '-53zl3bPmpM' at 210 s, with music, speech, and bang sound classes in reference annotation



(c) YouTube recording '-2xiZDEuHd8' at 30 s, with music, speech, and oink sound classes in reference annotation



(d) YouTube recording '-21_SXelVNo' at 30 s, with speech and sniff sound classes in reference annotation annotation

Figure 4: Visualization of SED results from FCN, and the input features for selected recordings from Audioset evaluation split.

no characteristic difference. This suggests that given only the weak labels, the proposed methods can estimate their temporal activities with good confidence.

5. CONCLUSION

In this paper, we studied two methods that perform joint SEC and SED using weakly labeled data, and evaluated the methods on the largest weakly labeled dataset — Audioset. The first method was based on a Fully Convolutional Network (FCN) and obtained a mean Average Precision (mAP) score of 0.324. The second novel method comprised multiple convolutional layers followed by an attention layer. This method was seen to perform better than the FCN with only 20% of the 25 M parameters in the FCN, and obtained a state-of-the-art mAP score of 0.330. In comparison to the baseline method trained on Audioset recordings, which was the previous state-of-the-art, the two methods improve the mAP score by at least a factor of 2.36. In fact, the two methods performed better than methods trained on Audioset embeddings that were obtained from learning on datasets larger than Audioset. This improvement in performance is a result of using a more powerful classifier and not assuming that the weak labels are active throughout the recording during training.

6. REFERENCES

[1] M. Xu, C. Xu, L. Duan, J. S. Jin, and S. Luo, "Audio keywords generation for sports video analysis," in *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 2008.

[2] A. Harma, M. F. McKinney, and J. Skowronek, "Automatic surveillance of the acoustic activity in our living environment," in *IEEE International Conference on Multimedia and Expo (ICME)*, 2005.

[3] M. Crocco, M. Cristani, A. Trucco, and V. Murino, "Audio surveillance: A systematic review," in *ACM Computing Surveys (CSUR)*, 2016.

[4] E. Browning, R. Gibb, P. Glover-Kapfer, and K. E. Jones, "Passive acoustic monitoring in ecology and conservation," in *World Wildlife Fund Conservation Technology Series 1(2)*, 2017.

[5] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015.

[6] B. J. Furnas and R. L. Callas, "Using automated recorders and occupancy models to monitor common forest birds across a large geographic region," in *Journal of Wildlife Management*, vol. 79, no. 2, 2014, p. 325337.

[7] "Sound event detection in real life audio." Detection and Classification of Acoustic Scenes and Events (DCASE), 2016. [Online]. Available: http://www.cs.tut.fi/sgn/arg/dcase2016/task-sound-event-detection-in-real-life-audio

[8] S. Adavanne, P. Pertilä, and T. Virtanen, "Sound event detection using spatial features and convolutional recurrent neural network," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[9] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 344–348.

[10] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[11] S. Adavanne, K. Drossos, E. Cakir, and T. Virtanen, "Stacked convolutional and recurrent neural networks for bird audio detection," in *European Signal Processing Conference (EUSIPCO)*, 2017.

[12] "Audio tagging with noisy labels." Detection and Classification of Acoustic Scenes and Events (DCASE), 2019. [Online]. Available: http://dcase.community/challenge2019/task-audio-tagging-results#machine-learning-characteristics

[13] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, 2017.

[14] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "Cnn architectures for large-scale audio classification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[15] A. Kumar and B. Raj, "Audio event detection using weakly labeled data," in *ACM on Multimedia Conference*, 2016.

[16] T.-W. Su, J.-Y. Liu, and Y.-H. Yang, "Weakly-supervised audio event detection using event-specific gaussian filters and fully convolutional networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[17] A. Kumar and B. Raj, "Deep CNN framework for audio event recognition using weakly labeled web data," in *Machine Learning for Audio Signal Processing Workshop at NIPS*, 2017.

[18] S. Adavanne and T. Virtanen, "Sound event detection using weakly labeled dataset with stacked convolutional and recurrent neural network," in *Workshop on Detection and classification of acoustic scenes and events (DCASE)*, 2017.

[19] A. Kumar, M. Khadkevich, and C. Fügen, "Knowledge transfer from weakly labeled audio using convolutional neural network for sound events and scenes," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[20] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: an ontology and human-labeled dataset for audio events," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[21] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, "Audio set classification with attention model: A probabilistic perspective," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[22] C. Yu, K. S. Barsim, Q. Kong, and B. Yang, "Multi-level attention model for weakly supervised audio classification," in *European Signal Processing Conference (EUSIPCO)*, 2018.

[23] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," in *Journal of Machine Learning Research (JMLR)*, 2014.

[25] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[26] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," in *Journal of Machine Learning Research (JMLR)*, 2012.

LOCALIZATION, DETECTION AND TRACKING OF MULTIPLE MOVING SOUND SOURCES WITH A CONVOLUTIONAL RECURRENT NEURAL NETWORK

Sharath Adavanne, Archontis Politis, and Tuomas Virtanen

Audio Research Group, Tampere University, Finland, firstname.lastname@tuni.fi

ABSTRACT

This paper investigates the joint localization, detection, and tracking of sound events using a convolutional recurrent neural network (CRNN). We use a CRNN previously proposed for the localization and detection of stationary sources, and show that the recurrent layers enable the spatial tracking of moving sources when trained with dynamic scenes. The tracking performance of the CRNN is compared with a stand-alone tracking method that combines a multi-source direction of arrival estimator and a particle filter. Their respective performance is evaluated in various acoustic conditions such as anechoic and reverberant scenarios, stationary and moving sources at several angular velocities, and with a varying number of overlapping sources. The results show that the CRNN manages to track multiple sources more consistently than the parametric method across acoustic scenarios, but at the cost of higher localization error.

Index Terms— Multiple object tracking, recurrent neural network, sound event detection, acoustic localization

1. INTRODUCTION

Sound event localization, detection, and tracking (SELDT) is the combined task of identifying the temporal onset and offset of potentially temporally-overlapping sound events, recognizing their classes, and tracking their respective spatial trajectory when they are active. Performing SELDT successfully provides an automatic description of the acoustic scene that can be employed by machines to interact naturally with their surroundings. Applications such as teleconferencing systems and robots can use this information for tracking the sound event of interest [1–6]. Furthermore, smart cities and smart homes can use it for audio surveillance [7–9].

The joint localization and detection in static scenes with spatially stationary sources have been studied with different parametric [5, 8, 10, 11] and deep neural network (DNN) [12] based methods. However, these methods do not employ any temporal modeling required for the tracking of moving sources in dynamic scenes. Recently, we proposed a convolutional recurrent neural network (SELDnet) that was shown to perform significantly better localization and detection than the only other existing DNN-based method [12]. SELDnet's capabilities to localize events in full azimuth and elevation under matched and unmatched acoustic conditions, and without relying on features dependent on specific microphone arrays, were studied and presented in [13]. However, all the existing DNN-based methods including[12, 13] have only studied static scenes.

On the other hand, stand-alone tracking methods have been widely studied for both stationary and moving sources based on spatial information only [14–20], additional spectral information [21, 22], or in conjunction with visual information [23]. Such parametric methods often require manual tuning of multiple parameters corresponding to the scene composition and dynamics, and new sets of parameters have to be identified manually for different sound scenes. Furthermore, tracking usually focuses on distinguishing source trajectories, with no regard to source signal content. In the case of temporally overlapping trajectories, track identities are assigned to individual trajectories, but these identities are not source dependent and are generally re-used for trajectories from different sources across the audio recording. A balance between consistent association and localization determines the tracker's performance in most cases. Alternatively, a detect-before-track approach, as in the proposed SELDnet, circumvents the association problem by first detecting the active sound events, and then assigning a track to each detected event. As long as such a system is able to react to time-varying conditions, with temporally and spatially overlapping sound events from both stationary and moving sources, it is also able to detect and track the sound events of interest.

In this work, we study the multi-source tracking capabilities of a detection and localization system based on our recently proposed SELDnet [13]. To the best of the authors knowledge, this is the first DNN-based SELDT studies. We show that training the SELDnet with dynamic scene data results in tracking, in addition to localization and detection. This tracking ability is enabled by the recurrent layers of the SELDnet that can model the evolution of spatial parameters as a sequence prediction task given the sequential features and their spatial trajectory information. We show that the recurrent layers are crucial for tracking, and in comparison to stand-alone trackers they additionally perform detection. Unlike the parametric tracking methods discussed earlier, the recurrent layer is a generic tracking method that learns directly from the data without manual tracker-engineering. Finally, we show that the tracking performance of SELDnet is comparable with stand-alone parametric tracking methods through evaluation on five datasets, representing scenarios with stationary and moving sources at different angular velocities, anechoic and reverberant environments, and different numbers of overlapping sources. The method and all the studied datasets are publicly available[1].

2. METHOD

The block diagram of SELDnet [13] is illustrated in Figure 1. The input to SELDnet is a multichannel audio recording, from which a feature extraction block extracts the phase and magnitude components of the spectrogram from each channel. The SELDnet maps the input spectrogram of T-frames length to two outputs of the same length – sound event detection, and tracking; together they

This work has received funding from the European Research Council under the ERC Grant Agreement 637422 EVERYSOUND.

[1] https://github.com/sharathadavanne/seld-net

https://doi.org/10.33682/xb0q-a335

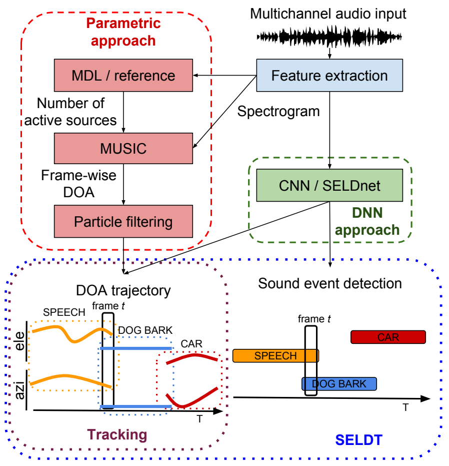


Figure 1: Workflows for the parametric tracking and DNN-based SELDT approaches. The sound class coloring and naming for the tracking task is only shown here to visualize the concept better. In practice tracking methods do not produce sound class labels as shown in Figure 3.

produce the SELDT output. The detection output is the class-wise probabilities for the C classes in the dataset of dimension $T \times C$, and is obtained as a multiclass multilabel classification task. The tracking output is a single direction of arrival (DOA) estimate per time frame for each sound class C as a multi-output regression task. Thus, when multiple instances of the same sound class are temporally overlapping, the SELDnet tracks only one instance or oscillate between the multiple instances. The estimated DOA is represented using 3D Cartesian coordinates of a point on a unit sphere around the microphone. The overall tracking output is of dimension $T \times 3C$, where $3C$ represents the three axes of the 3D Cartesian coordinates of a DOA for each class in C. Finally, to obtain the SELDT results, the class-wise probabilities of the detection output are binarized with a threshold of 0.5, anything greater represents the presence of the sound class and smaller represents the absence. The presence of a sound class in consecutive frames gives the onset and offset times, and the corresponding frame-wise DOA estimates from the tracking output when the sound class is active gives the DOA trajectory.

The SELDnet architecture used in this paper is identical to [13], with three convolutional layers of 64 filters each, followed by two-layers of 128-node gated recurrent units. The convolutional layers in the SELDnet are used as a feature extractor to produce robust features for detection and tracking. The recurrent layers are employed to model the temporal structure and the trajectory of the sound events. The output of the recurrent layers is shared between two branches of dense layers each with 128 units producing the detection and tracking estimates. The training and inference procedures of SELDnet are similar to [13] and is identical for both static and dynamic scenes, i.e., the same SELDnet designed for static scenes performs tracking when trained with moving scene data.

The recurrent layers utilize the current input frame along with the information learned from the previous input frames to produce the output for the current frame. This process is similar to a particle filter, which is a popular stand-alone parametric tracker and is also used as a baseline in this paper (see Section 3.3). The particle filter prediction at the current time frame is influenced by both the knowledge accumulated from the previous time frames and the input at the current time frame. For the tracking task of this paper, the particle filter requires the specific knowledge of the sound scene such as the spatial distribution of sound events, their respective velocity ranges when active, and their probability of birth and death. Such concepts are not explicitly modeled in the recurrent layers used in SELDnet, rather they learn equivalent information directly from the input convolutional layer features and corresponding target outputs in the development dataset. In fact, recurrent layers have been shown to work as generic trackers [24] that can learn temporal associations of the target source from any sequential input features. Unlike the particle filters that only work with conceptual representations such as frame-wise multiple DOAs for tracking, the recurrent layers work seamlessly with both conceptual and latent representations such as convolutional layer features.

Finally, by training the recurrent layers in SELDnet using the loss calculated from both detection and tracking, the recurrent layers learn associations between DOAs from neighboring frames corresponding to the same sound class and hence produce the SELDT results. In general, unlike the parametric trackers, the recurrent layers perform similar tracking of the frame-wise DOAs in addition to also detecting their corresponding sound classes. Further, the recurrent layers do not need complicated problem-specific tracker- or feature-engineering that are required by the parametric trackers. A more theoretical relationship between recurrent layers and particle filter is presented in [25].

Table 1: Summary of Datasets

Sources	Sound scene	Impulse response	Acronym
Stationary [13]	Anechoic	Synthetic	ANSYN
	Reverberant		RESYN
		Real-life	REAL
Moving	Anechoic	Synthetic	MANSYN
	Reverberant	Real-life	MREAL

3. EVALUATION PROCEDURE

3.1. Datasets

The performance of SELDnet is evaluated on five datasets that are summarized in Table 1. We continue to use the stationary source datasets: ANSYN, RESYN and REAL from our previous work [13] to evaluate the tracking performance of the parametric tracker that was missing in [13], and compare with SELDnet. The recordings in ANSYN and RESYN are synthesized in anechoic and reverberant environments respectively. The recordings in REAL are synthesized by convolving isolated real-life sound events with real-life impulse responses collected at different spatial locations within a room. Further, we create moving-source versions of the ANSYN and REAL datasets, hereafter referred as MANYSYN and MREAL, to evaluate the performance on moving sources. The recordings of all datasets are 30 seconds long and captured in the four-channel first-order Ambisonics format [26]. Each dataset has three subsets with no temporally overlapping sources $O1$, maximum two $O2$, and maximum three temporally overlapping sources $O3$. Each of these subsets has three cross-validation splits consisting of 240 recordings for development and 60 for evaluation. All the synthetic impulse response datasets (ANSYN, RESYN and MANYSN) have sound events from 11 classes and DOAs with full azimuth range and elevation range $\in [-60, 60)$. The real-life impulse response datasets (REAL and MREAL) have 8 sound event classes and DOAs in full azimuth range and elevation range $\in [-40, 40)$. During the synthesis of stationary source datasets, all the sound events are placed in a spatial grid of 10° resolution for both azimuth and elevation angles. We refer the readers to [13] for more details on these datasets.

The anechoic moving source dataset MANSYN has the same sound event classes as ANSYN and is synthesized as follows. Every event is assigned a spatial trajectory on an arc with a constant distance from the microphone (in the range 1-10 m) and moving with a constant angular velocity for its duration. Due to the choice of the ambisonic spatial recording format, the steering vectors for a plane wave source or point source in the far field are frequency-independent. Hence, there is no need for a time-variant convolution or impulse response interpolation scheme as the source is moving; the spatial encoding of the monophonic signal was done sample-by-sample using instantaneous ambisonic encoding vectors for the respective DOA of the moving source. The synthesized trajectories in MANSYN vary in both azimuth and elevation, and are simulated to have a constant angular velocity in the range $\in [-90°, 90°]/s$ with $10°/s$ steps. Similarly, the MREAL dataset was synthesized with real-life impulse responses from [13] that were sampled at $1°$ resolution along azimuth only. Hence, unlike MANSYN, the sound events in MREAL (that are identical to REAL) have motion only along the azimuth with a constant angular velocity in the range $\in [-90°, 90°]/s$ and $10°/s$ steps.

3.2. Metrics

The evaluation of the SELDT performance is done using individual metrics for detection and tracking identical to [13]. As the detection metric, we use the F-score and error rate calculated in segments of one-second with no overlap [27]. An ideal detection method will have an F-score of one and an error rate of zero. As the tracking metric, we use two frame-wise metrics: the frame recall and DOA error. The frame recall gives the percentage of frames in which the number of predicted DOAs is equal to the reference. The DOA error is calculated as the angle in degrees between the predicted and reference DOA. In order to associate multiple estimated DOAs with the reference, we use the Hungarian algorithm [28] to identify the smallest mean angular distance and use it as DOA error. An ideal tracking method has a frame recall of one and DOA error of zero (see [13] for more details).

3.3. Baseline Method

In the absence of publicly available implementations of multiple moving sound sources trackers, we use a combination of MUSIC [29] and an RBMCDA particle filter [30] to obtain tracking results in a similar fashion as in [15] and further made it publicly available [2]. The workflow of the baseline method is shown in Figure 1. MUSIC is a widely used [13, 31] subspace-based high-resolution DOA estimation method that can detect multiple narrowband sources. It relies on an eigendecomposition of the narrowband spatial covariance matrix computed from the multichannel spectrogram, and it additionally requires a source number estimate in order to distinguish between a signal and noise subspace. Herein, the number of active sources is taken from the reference of the dataset. To obtain broadband DOA estimates, the narrowband covariance matrices are averaged across three consecutive frames and frequency bins from 50 Hz to 8 kHz. We perform 2D spherical peak-finding on the resulting MUSIC pseudospectrum generated on a 2D angular grid with a $10°$ resolution for stationary and $1°$ for moving sources, in both azimuth and elevation. The final output of MUSIC MUS_{GT} is a list of frame-wise DOAs corresponding to the highest peaks equal to the number of active sources in each frame.

The second stage of the parametric method involves a particle filter that produces tracking results by processing the frame-wise DOA information of MUSIC MUS_{GT}. The particle filter assumes that the number of sources at each time frame is unknown and tracks them with respect to time using a fixed number of particles. At each time frame, the particle filter receives multiple DOAs and, based on knowledge accumulated from the previous time frames, it assigns each new DOA to one of the existing trajectories, clutter (noise), or a newborn source. Additionally, it also decides if any of the existing trajectories have died. The final output of the particle filter MUS_{GT}^{PF} produces the temporal onset-offset and the DOA trajectory for each of the active sound events. We refer the reader to [30] for the details of this approach.

[2]https://github.com/sharathadavanne/multiple-target-tracking

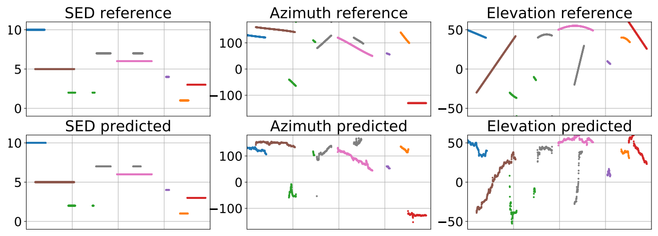


Figure 2: Visualization of the SELDnet predictions and its respective reference for a MANSYN $O2$ dataset recording. The horizontal-axis of all sub-plots represents the same time frames. The vertical-axis represents sound event class indices for the detection subplots, and DOA azimuth and elevation angles in degrees for remaining subplots.

3.4. Experiments

In all our experiments, the baseline particle filter parameters and the sequence length of input spectrogram for SELDnet was tuned using the development set of the respective subset. The performance of the tuned method was tested on the evaluation set of the subset, and the respective metrics averaged across the three cross-validation splits of the subset are reported.

Unlike the DNN-based method, the parametric method requires additional information on the number of active sources per frame to estimate the corresponding DOAs. However, SELDnet obtains this information from the data itself. In order to have a fair comparison, we used the minimum description length (MDL) [32] principle to estimate the number of sources from the input spectrogram and use it with MUSIC, resulting in the MUSIC output of MUS_{MDL} and the corresponding particle filter output of MUS_{MDL}^{PF}.

Finally, we studied the importance of recurrent layers for the SELDT task by removing them from SELDnet and evaluating the model containing only convolutional and dense layers, referred to as CNN hereafter. The best CNN architecture across datasets had five convolutional layers with 64 filters each.

4. RESULTS AND DISCUSSION

On tuning the input sequence length for SELDnet, it was observed that a sequence of 256 frames gave the best scores for the reverberant datasets, and 512 frames gave the best scores for the anechoic datasets. The SELDnet predictions and the corresponding references are visualized in Figure 2 for a respective 1000 frame test sequence from MANSYN $O2$ dataset. Each sound class is represented with a unique color across subplots. We see that the detected sound events are accurate in comparison to reference. The DOA predictions are seen to vary around the reference trajectory with a small deviation. This shows that SELDnet can successfully track and recognize multiple overlapping and moving sources.

Figure 3 visualizes the tracking predictions and their respective references for SELDnet and the baseline method MUS_{GT}^{PF}. In gen-

Table 2: Evaluation results on different datasets. Since the number of active sources information is used in $\mathrm{MUS_{GT}}$, the frame recall is always 100% and hence not reported. DE: DOA error, FR: Frame recall, F: F-score, SCOF: Same class overlapping frames

		ANSYN			RESYN			REAL			MANSYN			MREAL		
Tracking results		*O1*	*O2*	*O3*	*O1*	*O2*	*O3*	*O1*	*O2*	*O3*	*O1*	*O2*	*O3*	*O1*	*O2*	*O3*
$\mathrm{MUS_{GT}}$	DE	1.3	5.0	12.2	21.7	28.9	32.5	15.1	33.9	44.1	0.6	14.8	28.0	16.4	34.1	43.9
$\mathrm{MUS_{GT}^{PF}}$	DE	0.1	1.1	2.3	4.0	5.2	6.1	3.3	8.8	12.0	0.2	4.2	8.0	3.6	8.1	11.9
	FR	97.0	88.5	74.3	83.8	55.6	37.3	93.0	71.0	44.7	98.7	92.3	75.1	91.0	69.9	48.3
Methods estimating the number of active sources directly from input data																
$\mathrm{MUS_{MDL}}$	DE	0.5	14.2	24.0	22.3	31.9	38.5	25.3	36.2	44.1	4.2	17.8	28.5	26.5	35.9	44.9
	FR	93.9	**89.4**	**86.7**	61.7	45.6	52.5	53.6	35.7	**57.5**	63.8	48.1	51.85	53.4	35.2	**58.9**
$\mathrm{MUS_{MDL}^{PF}}$	DE	**0.1**	**4.4**	**7.2**	**6.4**	**10.6**	**12.7**	**9.3**	**10.9**	**13.7**	**3.5**	**6.8**	**8.0**	**13.6**	**11.2**	**13.6**
	FR	96.3	83.5	67.7	52.0	34.1	24.2	52.7	40.1	29.6	64	49.9	39.8	58.7	34.4	27.5
CNN	DE	25.7	25.2	26.9	39.1	35.1	31.4	32.0	34.9	37.1	26.1	25.8	28.2	36.6	39.3	40.2
	FR	80.2	45.6	32.2	69.5	45.8	29.7	45.1	28.4	16.9	83.7	58.1	38.3	44.5	26.2	16.3
SELDnet	DE	3.4	13.8	17.3	9.2	20.2	26.0	26.6	33.7	36.1	6.0	12.3	18.6	36.5	39.6	38.5
	FR	**99.4**	85.6	70.2	**95.8**	**74.9**	**56.4**	**64.9**	**41.5**	24.6	**98.5**	**94.6**	**80.7**	**69.6**	**42.8**	28.9
Detection results																
CNN	ER	0.52	0.46	0.51	0.44	0.45	0.54	0.52	0.51	0.51	0.59	0.47	0.48	0.46	0.49	0.52
	F	70.1	66.5	68	57	54.9	42.7	50.1	49.5	48.9	65.6	62.7	60.1	55.4	50.9	48.8
SELDnet	ER	**0.04**	**0.16**	**0.19**	**0.1**	**0.29**	**0.32**	**0.4**	**0.49**	**0.53**	**0.07**	**0.1**	**0.2**	**0.37**	**0.45**	**0.49**
	F	**97.7**	**89**	**85.6**	**92.5**	**79.6**	**76.5**	**60.3**	**53.1**	**51.1**	**95.3**	**93.2**	**87.4**	**64.4**	**56.4**	**52.3**
SCOF (in %)		0.0	4.2	12.1	0.0	4.2	12.1	0.0	7.6	23.0	0.0	3.0	9.1	0.0	7.1	20.9

eral, the performance of the two methods is visually comparable. Both methods are often confused in similar situations, for example in the intervals of 4-5 s, 10-13 s, and 23-25 s.

The SELDnet, by design, is restricted to recognize just one DOA for a given sound class. But in real life, there can be multiple instances of the same sound class occurring simultaneously. This is also seen in the datasets studied, the last row (SCOF) in the Table 2 presents the percentage of frames in which the same class is overlapping with itself. In comparison, the parametric method has no such restriction by design and can potentially perform better than SELDnet in these frames (even though, highly correlated sound events coming from different DOAs can easily degrade the performance of parametric methods such as MUSIC). The performance of the two methods in such a scenario can be observed in the 10-13 s interval of Figure 3. The SELDnet tracks only one of the two sources, while the parametric method tracks both overlapping sources and introduces an additional false track between the two trajectories.

Table 2 presents the quantitative results of the studied methods. The general trend is as follows. The higher the number of overlapping sources, the lower the tracking performance by both SELDnet and the parametric method. Across datasets, the DOA error improves considerably with the use of the temporal particle filter tracker, but at the cost of lower frame recall. By using MDL instead of reference information for the source number, the overall performance of the parametric approach reduces ($MUS_{GT}^{PF} > MUS_{MDL}^{PF}$). This reduction is especially observed in the frame recall metric, that drops significantly for reverberant and moving source scenario datasets, indicating the need for more robust source detection and counting schemes.

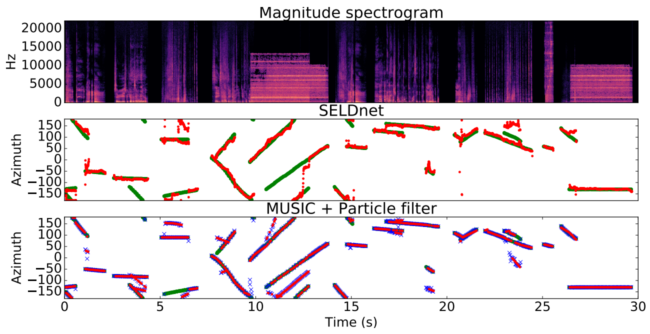


Figure 3: The tracking results of the two proposed methods are visualized for a MANSYN $O2$ dataset recording. The top figure shows the input spectrogram. The center and bottom figures show the output of SELDnet and $\mathrm{MUS_{GT}^{PF}}$ tracker in red, and the groundtruth in green. The blue crosses in the bottom figure represents the frame-wise DOA output of MUSIC

The frame recall of SELDnet is observed to be consistently better than MUS_{MDL}^{PF}, but the DOA estimation is poorer across datasets. A similar relationship is observed between SELDnet and MUS_{GT}^{PF} for all the datasets generated with simulated impulse responses, while for the real-life impulse response datasets the frame recall of SELDnet is poorer than MUS_{GT}^{PF}. That could indicate the need for more extensive learning for real-life impulse response datasets, with larger datasets and stronger models.

Using recurrent layers definitely helps the SELDT task. It was observed from visualizations that the tracking performance by the CNN was poor, with spurious and high variance DOA tracks, thus resulting in poor DOA error across datasets as seen in Table 2. This suggests that the recurrent layers are crucial for SELDT task and perform a similar task as an RBMCDA particle filter of identifying the relevant frame-wise DOAs and associating these DOAs corresponding to the same sound class across time frames.

5. CONCLUSION

In this paper, we presented the first deep neural network based method, SELDnet, for the combined tasks of detecting the temporal onset and offset time for each sound event in a dynamic acoustic scene, localizing them in space and tracking their position when active, and finally recognizing the sound event class. The SELDnet performance was evaluated on five different datasets containing stationary and moving sources, anechoic and reverberant scenarios, and a different number of overlapping sources. It was shown that the recurrent layers employed by the SELDnet were crucial for the tracking performance. Further, the tracking performance of SELDnet was compared against a stand-alone parametric method based on multiple signal classification and particle filter. In general, the SELDnet tracking performance was comparable to the parametric method and achieved a higher frame recall for tracking but at a higher angular error.

6. REFERENCES

[1] R. Takeda and K. Komatani, “Sound source localization based on deep neural networks with directional activate function exploiting phase information,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[2] ——, “Discriminative multiple sound source localization based on deep neural networks using independent location model,” in *IEEE Spoken Language Technology Workshop (SLT)*, 2016.

[3] N. Yalta, K. Nakadai, and T. Ogata, “Sound source localization using deep learning models,” in *Journal of Robotics and Mechatronics*, vol. 29, no. 1, 2017.

[4] W. He, P. Motlicek, and J.-M. Odobez, “Deep neural networks for multiple speaker detection and localization,” in *Int. Conf. on Robotics and Automation (ICRA)*, 2018.

[5] T. Butko, F. G. Pla, C. Segura, C. Nadeu, and J. Hernando, “Two-source acoustic event detection and localization: Online implementation in a smart-room,” in *European Signal Processing Conference (EUSIPCO)*, 2011.

[6] P. Swietojanski, A. Ghoshal, and S. Renals, “Convolutional neural networks for distant speech recognition,” in *IEEE Signal Processing Letters*, vol. 21, 2014.

[7] M. Crocco, M. Cristani, A. Trucco, and V. Murino, “Audio surveillance: A systematic review,” in *ACM Computing Surveys (CSUR)*, 2016.

[8] C. Grobler, C. Kruger, B. Silva, and G. Hancke, “Sound based localization and identification in industrial environments,” in *IEEE Industrial Electronics Society (IECON)*, 2017.

[9] P. W. Wessels, J. V. Sande, and F. V. der Eerden, “Detection and localization of impulsive sound events for environmental noise assessment,” in *The Journal of the Acoustical Society of America 141*, vol. 141, no. 5, 2017.

[10] R. Chakraborty and C. Nadeu, “Sound-model-based acoustic source localization using distributed microphone arrays,” in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[11] K. Lopatka, J. Kotus, and A. Czyzewsk, “Detection, classification and localization of acoustic events in the presence of background noise for acoustic surveillance of hazardous situations,” *Multimedia Tools and Applications Journal*, vol. 75, no. 17, 2016.

[12] T. Hirvonen, “Classification of spatial audio location and content using convolutional neural networks,” in *Audio Engineering Society Convention 138*, 2015.

[13] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, “Sound event localization and detection of overlapping sources using convolutional recurrent neural networks,” in *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, 2018.

[14] I. Potamitis, H. Chen, and G. Tremoulis, “Tracking of Multiple Moving Speakers With Multiple Microphone Arrays,” *IEEE Transactions on Speech and Audio Processing*, vol. 12, no. 5, pp. 520–529, 2004.

[15] J. M. Valin, F. Michaud, and J. Rouat, “Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering,” *Robotics and Autonomous Systems*, vol. 55, no. 3, pp. 216–228, 2007.

[16] N. Roman and D. Wang, “Binaural tracking of multiple moving sources,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 16, no. 4, pp. 728–739, 2008.

[17] X. Zhong and J. R. Hopgood, “Time-frequency masking based multiple acoustic sources tracking applying Rao-Blackwellised Monte Carlo data association,” in *IEEE Workshop on Statistical Signal Processing (SSP)*, 2009.

[18] M. F. Fallon and S. J. Godsill, “Acoustic source localization and tracking of a time-varying number of speakers,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 20, no. 4, pp. 1409–1415, 2012.

[19] J. Traa and P. Smaragdis, “Multiple speaker tracking with the Factorial von Mises-Fisher Filter,” in *IEEE Int. Workshop on Machine Learning for Signal Processing (MLSP)*, 2014.

[20] O. Schwartz and S. Gannot, “Speaker tracking using recursive EM algorithms,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 22, no. 2, pp. 392–402, 2014.

[21] J. Nix and V. Hohmann, “Combined estimation of spectral envelopes and sound source direction of concurrent voices by multidimensional statistical filtering,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 3, pp. 995–1008, 2007.

[22] J. Woodruff and D. Wang, “Binaural detection, localization, and segregation in reverberant environments based on joint pitch and azimuth cues,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 21, no. 4, pp. 806–815, 2013.

[23] N. Strobel, S. Spors, and R. Rabenstein, “Joint Audio-Video Signal Processing for Object Localization and Tracking,” in *Microphone Arrays*. Springer, 2001, pp. 203–225.

[24] J. Gu, X. Yang, S. De Mello, and J. Kautz, “Dynamic facial analysis: From bayesian filtering to recurrent neural network,” in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[25] Y. J. Choe, J. Shin, and N. Spencer, “Probabilistic interpretations of recurrent neural networks,” *Probabilistic Graphical Models*, 2017.

[26] V. Pulkki, A. Politis, M.-V. Laitinen, J. Vilkamo, and J. Ahonen, “First-order directional audio coding (DirAC),” in *Parametric Time-Frequency Domain Spatial Audio*. John Wiley & Sons, 2017, pp. 89–140.

[27] A. Mesaros, T. Heittola, and T. Virtanen, “Metrics for polyphonic sound event detection,” in *Applied Sciences*, vol. 6, no. 6, 2016.

[28] H. W. Kuhn, “The hungarian method for the assignment problem,” in *Naval Research Logistics Quarterly*, no. 2, 1955, p. 8397.

[29] R. O. Schmidt, “Multiple emitter location and signal parameter estimation,” in *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, 1986.

[30] S. Särkkä, A. Vehtari, and J. Lampinen, “Rao-blackwellized particle filter for multiple target tracking,” *Information Fusion*, vol. 8, no. 1, pp. 2–15, 2007.

[31] S. Adavanne, A. Politis, and T. Virtanen, “Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network,” in *European Signal Processing Conference (EUSIPCO)*, 2018.

[32] J. Rissanen, “Modeling by shortest data description,” *Automatica*, vol. 14, no. 5, pp. 465–471, 1978.

DCASE 2019 TASK 2: MULTITASK LEARNING, SEMI-SUPERVISED LEARNING AND MODEL ENSEMBLE WITH NOISY DATA FOR AUDIO TAGGING

Osamu Akiyama[*] *and Junya Sato*

Faculty of Medicine, Osaka University, Osaka, Japan
{oakiyama1986, junya.sto}@gmail.com

ABSTRACT

This paper describes our approach to the DCASE 2019 challenge Task 2: Audio tagging with noisy labels and minimal supervision. This task is a multi-label audio classification with 80 classes. The training data is composed of a small amount of reliably labeled data (curated data) and a larger amount of data with unreliable labels (noisy data). Additionally, there is a difference in data distribution between curated data and noisy data. To tackle these difficulties, we propose three strategies. The first is multitask learning using noisy data. The second is semi-supervised learning using noisy data and labels that are relabeled using trained models' predictions. The third is an ensemble method that averages models trained with different time length. By using these methods, our solution was ranked in 3rd place on the public leaderboard (LB) with a label-weighted label-ranking average precision (lwlrap) score of 0.750 and ranked in 4th place on the private LB with a lwlrap score of 0.75787. The code of our solution is available at https://github.com/OsciiArt/Freesound-Audio-Tagging-2019.

Index Terms— Audio-Tagging, Noisy Labels, Multitask Learning, Semi-supervised Learning, Model Ensemble

1. INTRODUCTION

An automatic general-purpose audio tagging system can be useful for various usages, including sound annotating or video captioning. However, there are no such systems with adequate performance because of the difficulty of this task. To build such a system using machine learning techniques, an audio dataset with reliable labels is required. However, it is difficult to obtain large-scale dataset with reliable labels because manual annotation by humans is time-consuming. In contrast, it is easy to infer labels automatically using metadata of websites like Freesound [1] or Flickr [2] that collect audio and metadata from collaborators. Nevertheless, automatically inferred labels are inevitable to have a certain amount of label noise.

DCASE 2019 challenge Task 2: Audio tagging with noisy labels and minimal supervision [3] is a multi-label audio classification task with 80 classes. The FSDKaggle2019 dataset was provided for this challenge. The main motivation of this task is to facilitate research of audio classification leveraging a small amount of reliably labeled data (curated data) and a larger amount of data with unreliable labels (noisy data) with a large number of categories.

This task has three main challenges. First, this is a multi-label classification task, which is more difficult than a single-label classification task. Second, most of the training data labels are so unreliable that the performance of a classification model trained with them would be lower than a one trained without them. Third, there is a difference in data distribution between curated data and noisy data because they come from different sources. Therefore, domain adaptation approaches would be required.

2. OUR PROPOSALS

2.1. MULTITASK LEARNING

In this task, the curated data and noisy data are labeled in a different manner, therefore treating them as the same one makes the model performance worse. To tackle this problem, we used a multitask learning approach [4, 5], in which a model learns multiple tasks simultaneously. The aim of multitask learning is to get a more generalized model by learning representations shared between 2 tasks. We treated learning with curated data and noisy data as different tasks and performed multitask learning. In our proposal, a convolution layer architecture learns the feature representations shared between curated and noisy data, and the two separated sequences of full-connect (FC) layers learn the difference between the two data (Fig. 1). In this way, we can get the advantages of representation learning from noisy data and avoid the disadvantages of noisy label perturbation. We set the loss weight ratio of curated and noisy as 1:1.

2.2. SEMI-SUPERVISED LEARNING

Because treating the noisy labels the same as the curated labels makes the model performance worse, it may be promising to do semi-supervised learning [6] (SSL) using the noisy data without the noisy labels. However, this task is different from the data that SSL is generally applied in two points. The first, there is a difference in data distribution between labeled data and unlabeled data. It is reported that applying SSL to such data makes model performance worse [6]. The second, this is a multi-label classification task. Most of SSL methods are for single-label classification task. We tried Pseudo-Label [7], Mean Teacher [8], and MixMatch [9] and all of them were not successful in improving lwlrap score. In the original Pseudo-Label, guessed labels are made from predictions of the training model itself, but we made guessed labels from trained models because it is a popular approach.

[*] This work was supported by Osaka University Medical Doctor Scientist Training Program.

https://doi.org/10.33682/0avf-bm61

Therefore, we propose an SSL method that is robust to data distribution difference and can handle multi-label data (Fig. 1). For each noisy data sample, we guess the label using trained models. The guessed label is processed by a sharpening function [9], which sharpens the predicted categorical distribution by adjusting the "temperature." We call this soft pseudo label. The basic Pseudo-Label is a hard label with only one positive label so that it cannot apply to multi-label data. In contrast, the soft pseudo label is sharpened label distribution and suits for multi-label data. The soft pseudo label is expected to be more robust to data distribution difference because it is smoother than the hard label. Learning with soft pseudo labels is performed in parallel with multitask learning. As the temperature of the sharpening function, we tried a value of 1, 1.5, and 2. A value of 2 was the optimum. The predictions used for the guessed labels were obtained from a ResNet model with multitask learning (Table 1 #4) using Snapshot Ensembles [10] and 5-fold cross validation (CV) averaging with all the folds and cycle snapshots of 5-fold CV. We used mean squared error (MSE) as a loss function. We set loss weight of SSL as 20. To get the benefit of mixup [11] more, we mixed curated data and its label to soft pseudo label data with a ratio of 1:1.

2.3. ENSEMBLE

To obtain the benefit of ensemble, we prepared models trained with various conditions and averaged the categorical distribution predicted by the models with weighted ratio (model averaging). As the variety of models, we employed 5-fold CV averaging, Snapshot Ensembles [10] and models trained with waveform or log mel spectrogram. K-fold CV averaging is averaging of predictions of all the models of k-fold CV on the test data. Snapshot Ensembles is averaging of predictions of model snapshots which is model weights of each cycle's end in model training with cyclic cosine learning rate [12]. As an approach specific to this competition data, we averaged models trained with different cropping length of time, we call this cropping length averaging. There is a difference in time length average among classes. Therefore, models trained with different time length are expected to become experts for different classes, and they give a variety to the model ensemble.

3. METHODOLOGY

3.1. DATASET

The FSDKaggle2019 dataset was provided for this challenge [3]. This dataset consists of four subsets: curated train data with 4,970 audio samples, noisy train data with 19,815 samples, public test data with 1,120 samples, and private test data with 3,361 samples. Each audio sample is labeled with 80 classes, including human sounds, domestic sounds, musical instruments, vehicles, and animals. Curated train data and test data are collected from Freesound dataset [1] and labeled manually by humans. Noisy train data is collected from Yahoo Flickr Creative Commons 100M dataset (YFCC) [2] and labeled using automated heuristics applied to the audio content and metadata of the original Flickr clips. All audio samples are single-channel waveforms with a sampling rate of 44.1kHz. In curated data, the duration of the audio samples ranges from 0.3 to 30 second, and the number of clips per class is 75,

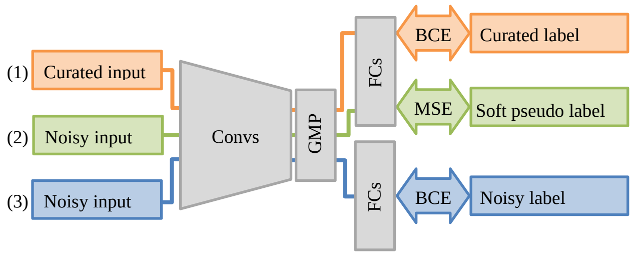


Figure 1: Overall architecture of our proposed model. The model is trained with three methods concurrently. (1) Basic classification (2) Soft pseudo label (3) Multitask learning with noisy label. Conv: convolution layer, GMP: global max pooling, FC: full-connect layers, BCE: binary cross-entropy, MSE: mean squared error.

except in a few cases. In noisy data, the duration of the audio samples ranges from 1 to 15 second, and the number of clips per class is 300, except in a few cases.

3.2. PREPROCESSING

We used both waveform and log mel spectrogram as input data. These two data types are expected to compensate for each other.

3.2.1. Waveform

We tried a sampling rate of 44.1 kHz (original data) and 22.05 kHz, and we found that 44.1 kHz was better. Each input data was regularized into a range of from -1 to +1 by dividing by 32,768, the full range of 16-bit audio.

3.2.2. Log mel spectrogram

For the log mel spectrogram transformation, we used 128 mel frequency channels. We tried 64 and 256, but model performance decreased. We used the short-time Fourier transform hop size of 347 that makes log mel spectrogram 128 Hz time resolution. Data samples of the log mel spectrogram were converted from power to dB after all augmentations were applied. After that, each data sample was normalized with the mean and standard deviation of each single data sample. Therefore, the mean and standard deviation values change every time, and this works as a kind of augmentation. Normalization using the mean and standard deviation of all the data decreased model performance.

3.3. AUGMENTATIONS

3.3.1. Augmentations for log mel spectrogram

Mixup/BC learning [11, 13] is an augmentation that mixes two pairs of inputs and labels with some ratio. The mixing rate is selected from a Beta distribution. We set a parameter α of the Beta distribution to 1.0, which makes the Beta distribution equal to a uniform distribution. We applied mixup with a ratio of 0.5.

SpecAugment [14] is an augmentation method for log mel spectrogram consists of three kinds of deformation. The first is time warping that deforms time-series in the time direction. The other two augmentations are time and frequency masking, modifications of Cutout [15], that masks a block of consecutive time steps or mel frequency channels. We applied frequency masking, and masking width is chosen from 8 to 32 from a uniform distribution. Time warping and time masking are not effective in this task, and we did not apply them to our models. We applied frequency masking with a ratio of 0.5.

For training, audio samples which have various time lengths are converted to a fixed length by random cropping. The sound samples which have short length than the cropping length are extended to the cropping length by zero paddings. We tried 2, 4, and 8 seconds (256, 512, and 1024 dimensions) as a cropping length and 4 seconds scores the best. Averaging models trained with 4-second cropping and 8-second cropping achieved a better score.

Expecting more strong augmentation effect, after basic cropping, we shorten data samples in a range of 25 - 100% of the basic cropping length by additional cropping and extend to the basic cropping length by zero padding. For data samples with a time length shorter than the basic cropping length, we shorten data samples in a range of 25 - 100% of original length by additional cropping and extend to the basic cropping length by zero paddings. We applied this additional cropping with a ratio of 0.5.

As another augmentation, we used gain augmentation with a factor randomly selected from a range of 0.80 - 1.20 with a ratio of 0.5. We tried scaling augmentation and white noise augmentation, but model performance decreased.

3.3.2. *Augmentations for waveform*

We applied mixup to waveform input. We used a parameter α of 1.0 for the Beta distribution as same as the case of log mel spectrogram.

We applied cropping to waveform input. We tried 1.51, 3.02, and 4.54 seconds (66,650, 133,300, and 200,000 dimensions) as a cropping length, and we found that 4.54 seconds is optimal. Averaging models trained with 3.02-second cropping and 4.54-second cropping achieved a better score.

We used scale augmentation with a factor randomly selected from a range of 0.8 - 1.25 and gain augmentation with a factor randomly selected from a range of 0.501 - 2.00.

3.4. MODEL ARCHITECTURE

3.4.1. *ResNet*

We selected ResNet [16] as a log mel spectrogram-based model because it is a widely-used image classification model and relatively simple. We compared ResNet18, ResNet34 and SE-ResNeXt50 [17] and ResNet34 performed the best. The number of trainable parameters, including the multitask module is 44,210,576. We applied a global max pooling (GMP) after convolutional layers to make a model adaptive to various input length.

3.4.2. *EnvNet*

We selected EnvNet-v2 [13] as a waveform-based model because it is state of the art of a waveform-based model. The number of trainable parameters, including the multitask module is 4,128,912. As same as ResNet, we applied a GMP after convolutional layers to allow variable input length.

3.4.3. *Multitask module*

For multitask learning, two separate FC layer sequences follow after convolution layers and GMP. The contents of both sequences are the same and consist of FC (1024 units) - ReLU - dropout [18] (drop rate = 0.2) - FC (1024 units) - ReLU - dropout (drop rate = 0.1) - FC (80 units) - sigmoid. Sigmoid is replaced by softmax in model E and F of EnvNet (Table 2).

3.5. TRAINING

3.5.1. *ResNet*

We used Adam [19] for optimization. We used cyclic cosine learning rate for learning rate schedule. In each cycle, the learning rate is started with 1e-3 and decrease to 1e-6. There are 64 epochs per cycle. We used a batch size of 32 or 64. We used binary cross-entropy (BCE) as a loss function for basic classification and multitask learning with noisy data. We used mean squared error as a loss function for the soft pseudo label. The model weights of each cycle's end were saved and used for Snapshot Ensembles.

3.5.2. *EnvNet*

We used stochastic gradient descent (SGD) for optimization. We used cyclic cosine learning rate for learning rate schedule. In each cycle, the learning rate is started with 1e-1 and decrease to 1e-6. There are 80 epochs per cycle. We used binary cross-entropy as a loss function for the model using sigmoid and Kullback-Leibler divergence for the model using softmax. We used a batch size of 64 for the model using sigmoid and 16 for the model using softmax.

3.6. POSTPROCESSING AND ENSEMBLE

Prediction using the full length of audio input scores better than prediction using test time augmentation (TTA) with cropped audio input. This may be because essential components for classification is concentrated on the beginning part of audio samples. Prediction with cropping of the beginning part scores better than prediction with cropping of the latter part. In order to speed up the calculation, audio samples with similar lengths were grouped, and the lengths of samples in the same group were adjusted to the same length by zero paddings and converted to mini-batches. The patience for the difference of length within a group (patience rate) was adjusted based on the prediction speed.

We found that padding augmentation is effective TTA. Padding augmentation is an augmentation method that applies zero paddings to both sides of audio samples with various length and averages prediction results. In the training phase, we applied padding to input data to make the sample size the same. Because there is a correlation between time length and class, models are thought to learn that there is a correlation between padding length and class. We think that padding augmentation reduces this bias and gives better predictions.

For model averaging, we prepared models trained with various conditions, as mentioned in section 2.3. In order to reduce prediction time, the cycles and padding lengths used for the ensemble were chosen based on CV (For more details, please refer

#	condition	CV lwlrap
1	1 × 512, Crop = 512, BS = 64	0.724
2	1 × 512, Crop = 512, BS = 64, **Augs**	0.829
3	**8 × 64**, Crop = 512, BS = 64, Augs	0.829
4	8 × 64, Crop = 512, BS = 64, Augs, **MTL** (**model A**)	0.849
5	**7 × 64**, Crop = 512, BS = 32, Augs, **AC**, MTL, **5-fold SPL**, **use #1 weights as pretrained weights** (**model B**)	0.870
6	7 × 64, Crop = 512, BS = 32, Augs, AC, MTL, **1-fold SPL**, use #1 weights as pretrained weights	0.858
7	**6 × 64**, **Crop = 1,024**, BS = 64, Augs, MTL (**model C**)	0.840

Table 1: Comparison of each learning condition of ResNet34. CV lwlrap is calculated based on the best epoch of each fold in 5-fold CV. *m* × *x*: *m* cycles of *n* epochs, Crop: cropping length, BS: batch size, Augs: MixUp, frequency masking, and gain augmentation, MTL: multitask learning, AC: additional cropping, SPL: soft pseudo label.

#	condition	CV lwlrap
8	1 × 400, Crop = 133,300, BS = 16, Augs, MTL, softmax	0.809
9	**3 × 80**, Crop = 133,300, **BS = 64**, Augs, MTL, **sigmoid**, **use #8 weights as pre-trained weight** (**model D**)	0.814
10	**5 × 80**, Crop = 133,300, **BS = 16**, Augs, MTL, **softmax**, use #8 weights as pre-trained weight (**model E**)	0.818
11	**10 × 80**, **Crop = 200,000**, BS = 16, Augs, MTL, softmax (**model F**)	0.820

Table 2: Comparison of each learning condition of EnvNet-v2. CV lwlrap is calculated based on the best epoch of each fold in 5-fold CV except for #8, which is calculated based on the final epoch. Augs: MixUp, gain, and scaling augmentation.

to our repository). For the final submission, we used the predictions of model A – F with 5-fold averaging, Snapshot Ensembles, and padding augmentation. The weights of model averaging are model A:B:C:D:E:F = 3:4:3:1:1:1, which is chosen based on CV. The total number of predictions is 170 (submission 1). In the simplified version submission, we omitted padding augmentation, and the total number of predictions is 95 (submission 2).

4. RESULT

Table 1 and 2 show the results of each learning condition. The score is lwlrap of 5-fold CV. By multitask learning, the CV lwlrap improved from 0.829 to 0.849 (Table 1 #3 and #4) and score on the public LB increased + 0.021. By soft pseudo labeling, The CV lwlrap improved from 0.849 to 0.870 (Table 1 #4 and #5). On the other hand, on the test data (private LB), improvement in score was smaller (+0.009). We used predictions of all fold of models to generate soft pseudo label so that high CV is maybe because of indirect label leak. However, even if we use labels generated by only the same fold model, which has no label leak, CV was improved as compared to one without SSL (Table 1 #4 and #6).

Table 3 shows the results of each model averaging condition.

#	condition	CV lwlrap
1	model A, cycle = 1-8, Pad = 8, 32	0.868
2	model B, cycle = 1-7, Pad = 8, 32	0.886
3	model C, cycle = 1-6, Pad = 8, 32	0.862
4	model D, cycle = 1-3, Pad = 8k, 32k	0.815
5	model E, cycle = 1-5, Pad = 8k, 32k	0.818
6	model F, cycle = 5-10, Pad = 8k, 32k	0.820
7	model A + C	0.876
8	model A + B + C	0.890
9	model D + E + F	0.836
10	submission 1	0.896
11	submission 2	0.895

Table 3: Comparison of model averaging. Pad: padding augmentation.

In every condition, we employed 5-fold CV averaging and Snapshot Ensembles. By Snapshot Ensembles and padding augmentation, the CV lwlrap increased +0.039 (Table 1 #4 and Table 3 #1). By cropping length averaging, the CV lwlrap increased +0.008 (Table 3 #1 and #7). By averaging models trained with log mel spectrogram and waveform, the CV lwlrap increased +0.008 (Table 3 #8 and #10).

On the public LB, submission 1 was ranked in 3rd place with a lwlrap score of 0.750. On the private LB, submission 2 was ranked in 4th place with a lwlrap score of 0.75787.

5. DISCUSSION

Our proposed methods showed meaningful results in the task, but there is room for improvement. First, we used shared convolution layers and separated FC layers for multitask learning, but we did not evaluate whether this model architecture is optimal. The optimized architecture may get more benefit from multitask learning.

Second, soft pseudo label failed to achieve reliable CV because of implicit label leak from soft pseudo label. The procedure of soft pseudo label is very similar to model distillation [20], which uses averages of trained models' predictions for training in the purpose of transferring knowledge of trained models to a single smaller model. Therefore, soft pseudo labels obtained from models trained with other CV folds have knowledge of labels of out-of-fold, and this can be label leaks. Establishing the way of reliable CV would make soft pseudo label more useful.

Third, we found that model averaging using models trained with different time length improves the score. This result suggests that training a single model with various time length would be successful and contributes to reducing the number of models.

Fourth, zero padding in training time makes models learn that there is a correlation between zero values and classes. Full-length prediction and padding augmentation can reduce this unpreferable bias. However, to avoid zero padding and concatenate several clones of the sound file instead may be more promising.

6. CONCLUSION

This paper describes our approach to the DCASE 2019 challenge Task 2, which is a difficult task because of multi-label and noisy label. We propose three strategies, multitask learning with noisy data, SSL with soft pseudo label and ensemble of cropping length averaging. By using these methods, our solution ranked in 4th place on the private LB.

7. REFERENCES

[1] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, “Freesound datasets: a platform for the creation of open audio datasets,” in *ISMIR 2017*, 2017, pp. 486-493

[2] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li, “YFCC100M: The New Data in Multimedia Research,” *Commun ACM*, 2016, 59(2), pp. 64‑73.

[3] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, and X. Serra, “Audio tagging with noisy labels and minimal supervision,” *arXiv preprint arXiv:1906.02975*, 2019.

[4] S. Ruder, “An Overview of Multi-Task Learning in Deep Neural Networks,” *arXiv preprint arXiv:1706.05098*, 2017.

[5] T. Lee, T. Gong, S. Padhy, A. Rouditchenko, and A. Ndirango, “Label-efficient audio classification through multitask learning and self-supervision,” in *ICLR 2019 Workshop LLD*, 2019.

[6] A. Oliver, A. Odena, C. Raffel, E. D. Cubuk, and I.J. Goodfellow, “Realistic Evaluation of Deep Semi-Supervised Learning Algorithms,” *arXiv preprint arXiv: 1804.09170*, 2018.

[7] D.-H. Lee, “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks,” in *ICML Workshop on Challenges in Representation Learning*, 2013.

[8] A. Tarvainen, and H. Valpola. “Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results,” *arXiv preprint arXiv:1703.01780*, 2017.

[9] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, “MixMatch: A Holistic Approach to Semi-Supervised Learning,” *arXiv preprint arXiv:1905.02249*, 2019.

[10] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, “Snapshot Ensembles: Train 1, get M for free,” *arXiv preprint arXiv:1704.00109*, 2017.

[11] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” *arXiv preprint arXiv:1710.09412*, 2017.

[12] I. Loshchilov, F. Hutter, “SGDR: Stochastic Gradient Descent with Warm Restarts,” *arXiv preprint arXiv:1608.03983*, 2016.

[13] Y. Tokozume, Y. Ushiku, and T Harada, “Learning from Between-class Examples for Deep Sound Recognition,” *arXiv preprint arXiv:1711.10282*, 2017.

[14] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, “SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition,” *arXiv preprint arXiv:1904.08779*, 2019.

[15] T. DeVries, and G. W. Taylor, “Improved Regularization of Convolutional Neural Networks with Cutout,” *arXiv preprint arXiv:1708.04552*, 2017.

[16] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

[17] J. Hu, L. Shen, and G. Sun, “Squeeze-and-Excitation Networks,” *arXiv preprint arXiv:1709.01507*, 2017.

[18] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting,” *arXiv preprint arXiv: 1207.0580*, 2015.

[19] D. P. Kingma, and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[20] G. Hinton, O. Vinyals, and J. Dean, “Distilling the Knowledge in a Neural Network,” *arXiv preprint arXiv: 1503.02531*, 2015.

POLYPHONIC SOUND EVENT DETECTION AND LOCALIZATION USING A TWO-STAGE STRATEGY

Yin Cao,[1*] *Qiuqiang Kong,*[1*] *Turab Iqbal,*[1] *Fengyan An,*[2] *Wenwu Wang,*[1] *Mark D. Plumbley*[1]

[1]Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey, UK
{yin.cao, q.kong, t.iqbal, w.wang, m.plumbley}@surrey.ac.uk
[2]School of Mechanical & Automotive Engineering, Qingdao University of Technology, China
anfy@qut.edu.cn

ABSTRACT

Sound event detection (SED) and localization refer to recognizing sound events and estimating their spatial and temporal locations. Using neural networks has become the prevailing method for SED. In the area of sound localization, which is usually performed by estimating the direction of arrival (DOA), learning-based methods have recently been developed. In this paper, it is experimentally shown that the trained SED model is able to contribute to the direction of arrival estimation (DOAE). However, joint training of SED and DOAE degrades the performance of both. Based on these results, a two-stage polyphonic sound event detection and localization method is proposed. The method learns SED first, after which the learned feature layers are transferred for DOAE. It then uses the SED ground truth as a mask to train DOAE. The proposed method is evaluated on the DCASE 2019 Task 3 dataset, which contains different overlapping sound events in different environments. Experimental results show that the proposed method is able to improve the performance of both SED and DOAE, and also performs significantly better than the baseline method.

***Index Terms*—** Sound event detection, source localization, direction of arrival, convolutional recurrent neural networks

1. INTRODUCTION

Sound event detection is a rapidly developing research area that aims to analyze and recognize a variety of sounds in urban and natural environments. Compared to sound tagging, event detection also involves estimating the time of occurrence of sounds. Automatic recognition of sound events would have a major impact in a number of applications [1]. For instance, sound indexing and sharing, bio-acoustic scene analysis for animal ecology, smart home automatic audio event recognition (baby cry detection, window break alarm), and sound analysis in smart cities (security surveillance).

Recently, approaches based on neural networks have been shown to be especially effective for SED [2]. Unlike audio tagging problems [3–5], which only aim to detect whether the sound events are present in a sound clip, SED also involves predicting temporal information of events. Early neural network architectures utilized fully-connected layers to detect temporally-overlapping sound events [6]. More recently, due to their success in image recognition, convolutional neural networks (CNNs) have become the prevailing architecture in this area [7–10]. Such methods use suitable time-frequency representations of audio, which are analogous to the image inputs in computer vision. Another popular type of neural network is the recurrent neural network (RNN), which has the ability to learn long temporal patterns present in the data, making it suitable for SED [11]. Hybrids containing both CNN and RNN layers, known as convolutional recurrent neural networks (CRNNs), have also been proposed, which have led to state-of-the-art performance in SED [4, 12].

Sound source localization, which focuses on identifying the locations of sound sources, on the other hand, has been an active research topic for decades [13]. It plays an important role in applications such as robotic listening, speech enhancement, source separation, and acoustic visualization. Unlike the dominance of neural-network-based techniques in SED, DOAE is mainly studied using two methods: parametric-based methods and learning-based methods.

Parametric-based DOAE methods can be divided into three categories [13]: time difference of arrival (TDOA) estimation, maximized steered response power (SRP) of a beamformer, and high-resolution spectral estimation. Generalized cross-correlation (GCC) methods are the most widely-used approaches for TDOA estimation [14, 15]. Since the TDOA information is conveyed in the phase rather than the amplitude of the cross-spectrum, the GCC Phase Transform (GCC-PHAT) was proposed, which eliminates the effect of the amplitude while leaving only the phase [14]. The primary limitation of parametric-based GCC methods is the inability to accommodate multi-source scenarios.

Learning-based DOAE methods have the advantages of good generalization under different levels of reverberation and noise. They are designed to enable the system to learn the connections between input features and the DOA. There has already been a series of research addressing DOAE using deep neural networks [16–24]. Results show that they are promising and comparable to parametric methods. However, these neural-network-based methods are mainly based on static sources. In addition to spectrum-based features, GCC-based features, which can effectively supply time difference information, have also been used as the input features [16, 17, 21–23]. In order to further improve learning-based methods, more practical real-world sources need to be considered.

In real-world applications, a sound event is always transmitted in one or several directions. Given this fact, it is reasonable to combine sound event detection and localization with not only estimating their respective associated spatial location, but also identifying the type and temporal information of sound. Therefore, it is worthwhile to study them together and investigate the effects and potential connections between them. Recently, DCASE 2019 introduced Task 3, which is Sound Event Localization and Detection (SELD) for overlapping sound sources [25]. A recently-developed system known as

* Equal contribution.

https://doi.org/10.33682/4jhy-bj81

SELDnet was used as the baseline system. SELDnet uses magnitude and phase spectrograms as input features and trains the SED and DOAE objectives jointly [26]. However, phase spectrograms are hard for neural networks to learn from, and further relationships between SED and DOAE have not been revealed.

In this paper, joint training of SED and DOAE is implemented first with log mel spectrograms and GCC-PHAT as the input features. According to the experimental results, SED is able to contribute to the performance of DOAE, while joint training of SED and DOAE degrades the performance of both. To solve this problem, a new two-stage method for polyphonic sound event detection and localization is proposed. This method deals with sound event detection and localization in two stages: the SED stage and the DOAE stage, corresponding to the SED branch and the DOAE branch in the model, respectively. During training, the SED branch is trained first only for SED, after which the learned feature layers are transferred to the DOAE branch. The DOAE branch fine-tunes the transferred feature layers and uses the SED ground truth as a mask to learn only DOAE. During inference, the SED branch estimates the SED predictions first, which are used as the mask for the DOAE branch to infer predictions. The experimental results show that by using the proposed method, DOAE can benefit from the SED predictions; both SED and DOAE can be improved at the same time. The proposed method performs significantly better than the baseline method.

The rest of the paper is organized as follows. In Section 2, the proposed learning method is described in detail. Section 3 introduces the dataset used, other methods for comparison, and experimental results. Finally, conclusions are summarized in Section 4.

2. LEARNING METHOD

Joint training of SED and DOAE was first proposed in [26]. Their system is also used as the baseline system for DCASE 2019 Task 3. In this baseline, temporal consecutive magnitude and phase spectrograms are extracted as the input features from the time-domain audio waveform, which are then fed into a CRNN. Its loss is a weighted combination of the SED loss and the DOAE loss. Therefore, it can be imagined that this baseline system has an intrinsic trade-off between SED and DOAE according to the loss weight selected. In this paper, a two-stage polyphonic sound event detection and localization network is proposed to exploit their mutual strength.

2.1. Features

Selecting which features to use is an important factor for audio-related neural network applications. In this paper, the input signal format is of two types: First-Order of Ambisonics (FOA) or tetrahedral microphone array [25]. Log mel spectrograms and GCC-PHAT, which contains phase difference information between all microphone pairs, are chosen as the input features. Ambisonics and GCC-PHAT are explained in this section.

2.1.1. Ambisonics

Ambisonics was developed as a spatial sound encoding approach several decades ago [27]. It is based on the spherical harmonic (SH) decomposition of the sound field. Ambisonics encoding for plane-wave sound fields can be expressed as

$$\mathbf{b}(t) = \sum_{n=0}^{N} \mathbf{y}_n s_n(t), \tag{1}$$

where $s_n(t)$ is the n-th plane-wave source signal, N is the total number of sources, and $\mathbf{y}_n$ is the vector of the spherical harmonic function values for direction (θ_n, ϕ_n), and can be expressed as

$$\begin{aligned}\mathbf{y}_n = \big[&Y_0^0(\theta_n,\phi_n), Y_1^{-1}(\theta_n,\phi_n), Y_1^0(\theta_n,\phi_n),\\ &Y_1^1(\theta_n,\phi_n),\ldots,Y_L^{-L}(\theta_n,\phi_n),\ldots,Y_L^0(\theta_n,\phi_n),\\ &\ldots,Y_L^L(\theta_n,\phi_n)\big]^T,\end{aligned} \tag{2}$$

where L indicates the order of Ambisonics. It can be seen that Ambisonics contains the information of the source DOA. In addition, a higher directional resolution relates to a higher order of Ambisonics. Order-L of Ambisonics needs at least $(L+1)^2$ microphones to encode. In real applications, the sound field is recorded using a spherical microphone array and converted into Ambisonics.

2.1.2. Generalized Cross-Correlation

GCC is widely used in TDOA estimation by means of maximizing the cross-correlation function to obtain the lag time between two microphones. The cross-correlation function is usually calculated through the inverse-FFT of the cross power spectrum. GCC-PHAT is the phase-transformed version of GCC, which whitens the cross power spectrum to eliminate the influence of the amplitude, leaving only the phase. GCC-PHAT can be expressed as

$$GCC_{ij}(t,\tau) = \mathcal{F}^{-1}_{f\to\tau}\frac{X_i(f,t)X_j^*(f,t)}{|X_i(f,t)||X_j(f,t)|}, \tag{3}$$

where $\mathcal{F}^{-1}_{f\to\tau}$ is the inverse-FFT from f to τ, $X_i(f,t)$ is the Short-Time Fourier Transform (STFT) of the i-th microphone signal, and * denotes the conjugate. TDOA, which is the lag time $\Delta\tau$ between two microphones, can then be estimated by maximizing GCC with respect to τ. Nevertheless, this estimation is usually not stable, especially in high reverberation and low SNR environments, and does not directly work for multiple sources. However, $GCC_{ij}(t,\tau)$ contains all of the time delay information and is generally short-time stationary. $GCC_{ij}(t,\tau)$ can also be considered as a GCC spectrogram with τ corresponding to the number of mel-band filters. That is, GCC-PHAT can be stacked with a log mel spectrogram as the input features. In order to determine the size of GCC-PHAT, the largest distance between two microphones $d_{\max}$ needs to be used. The maximum delayed samples corresponding to $\Delta\tau_{\max}$ can be estimated by $d_{\max}/c \cdot f_s$, where c is the sound speed and f_s is the sample rate. In this paper, log mel and GCC-PHAT are stacked as the input features, considering the possibility of the advance and the delay of GCC. The number of mel-bands, therefore, should be no smaller than the doubled number of delayed samples plus one [14].

2.2. Network architecture

The network is shown in Fig. 1, and has two branches, the SED branch and the DOAE branch. During training, the extracted features, which have shape $C \times T \times F$, are first sent to the SED branch. C indicates the number of feature maps, T is the size of time bins, and F is the number of mel-band filters or delayed samples of GCC-PHAT. The CNN layers, which are also named as feature layers in this paper, are constructed with 4 groups of 2D CNN layers (Convs) with 2×2 average-pooling after each of them. Each Convs' group consists of two 2D Convs, with a receptive field of 3×3, a stride of 1×1, and a padding size of 1×1 [10]. The Convs' kernels are able to filter across all of the channels of the input features or

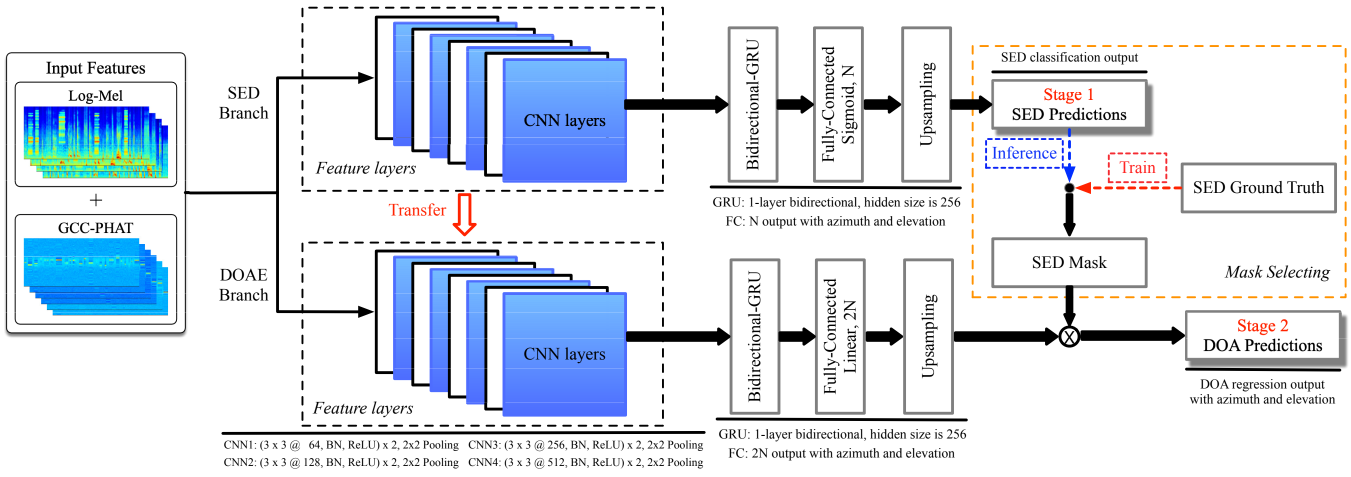

Input Features
Log-Mel
+
GCC-PHAT
SED Branch
DOAE Branch
CNN layers
Feature layers
Transfer
CNN layers
Feature layers
CNN1: (3 x 3 @ 64, BN, ReLU) x 2, 2x2 Pooling CNN3: (3 x 3 @ 256, BN, ReLU) x 2, 2x2 Pooling
CNN2: (3 x 3 @ 128, BN, ReLU) x 2, 2x2 Pooling CNN4: (3 x 3 @ 512, BN, ReLU) x 2, 2x2 Pooling
Bidirectional-GRU
Fully-Connected Sigmoid, N
Upsampling
GRU: 1-layer bidirectional, hidden size is 256
FC: N output with azimuth and elevation
Bidirectional-GRU
Fully-Connected Linear, 2N
Upsampling
GRU: 1-layer bidirectional, hidden size is 256
FC: 2N output with azimuth and elevation
SED classification output
Stage 1
SED Predictions
Inference
Train
SED Ground Truth
SED Mask
Mask Selecting
Stage 2
DOA Predictions
DOA regression output with azimuth and elevation

Figure 1: The diagram of the proposed two-stage sound event detection and localization network. SED ground truth is used as the mask to train DOAE branch. SED predictions are used as the mask to infer DOAE.

the feature maps from the last layer, hence are able to learn inter-channel information. CNN layers are capable of learning local temporal and frequency information to better abstract the event-level information. Each single CNN layer is followed by a Batch Normalization layer [28] and a ReLU activation. After the CNN layers, the data has shape $C_{out} \times T/16 \times F/16$, where C_{out} is the number of output feature maps of the last CNN layer. It is then sent to a global average-pooling layer to reduce the dimension of F. After this, the data is reshaped to have shape $T/16 \times C_{out}$ and is fed to a bidirectional GRU. The output size is maintained and is sent to a fully-connected layer with output size $T/16 \times N$, where N is the number of event classes. The sigmoid activation function is used afterwards with an upsampling in the temporal dimension to ensure the output size is consistent with T. The SED predictions can then be obtained through an activation threshold. Binary cross-entropy is used for this multi-label classification task.

The DOAE branch is then trained. The CNN layers are transferred from the SED branch and are fine-tuned. The output of the fully-connected layer for the DOAE branch is a vector of $N \times 2$ linear values, which are azimuth and elevation angles for N events. They are then masked by the SED ground truth during training to determine if the corresponding angles are currently active. Finally, the mean absolute error is chosen as the DOAE regression loss.

During inference, the SED branch first computes the SED predictions, which are then used as the SED mask to obtain the DOAE. The reason for building this network architecture is to enhance the representation ability of a single network so that each branch is only responsible for one task, while the DOAE branch can still incorporate the benefits contributed from SED.

3. EXPERIMENTAL STUDY

The proposed two-stage polyphonic sound event detection and localization method is compared with other methods described in Section 3.2. They are evaluated on the DCASE 2019 Task 3 dataset [25]. This task is for sound event detection and localization. The dataset provides two formats of data: 1) First-Order of Ambisonics; 2) tetrahedral microphone array. The development set consists of 400 one minute long recordings, divided into four cross-validation splits. There are 11 kinds of isolated sound events in total. The audio recordings are mixtures of isolated sound events and natural ambient noise. The sound events are convolved with impulse responses collected from five indoor locations, resulting in 324 unique combinations of azimuth-elevation angles. One challenging problem in this dataset is that the sound events in the audio recordings have a polyphony of up to two, which means sound events from different locations may overlap. The source code for this paper is released on GitHub[1].

3.1. Evaluation metrics

Polyphonic sound event detection and localization are evaluated with individual metrics for SED and DOAE. For SED, segment-based error rate (ER) and F-score [29] are calculated in one-second lengths. A lower ER or a higher F-score indicates better performance. In addition, mean average precision (mAP), which is the area under the precision and recall curve, is used to evaluate the frame-level tagging performance. The mAP is used here because it does not depend on the threshold selection, hence is able to better objectively evaluate the performance. A higher mAP indicates better performance. For DOAE, DOA error and frame recall are used [24]. A lower DOA error or a higher frame recall indicates better performance.

3.2. Methods for comparison

In order to show the effectiveness of the proposed method, several other methods are compared, including

- **Baseline**, which is the baseline method used in DCASE 2019 Task 3, uses magnitude and phase spectrograms as the input features. The features are then fed to a CRNN network. The loss of SED and DOAE are combined and jointly trained.
- **SELDnet**, which has the same architecture with the baseline but using log mel and GCC-PHAT spectrograms as the input features.
- **DOA**, which uses log mel and GCC-PHAT spectrograms as the input features to only estimate DOA. It transfers the CNN layers from the SED network and utilizes SED ground truth as the mask.

[1]https://github.com/yinkalario/Two-Stage-Polyphonic-Sound-Event-Detection-and-Localization

Table 1: Performance for the development dataset.

		MIC-ARRAY					FOA				
Methods	Net	ER	F	mAP	DOA	FR	ER	F	mAP	DOA	FR
Baseline	CRNN	0.350	0.800	–	30.8°	0.840	0.340	0.799	–	28.5°	0.854
SELDnet	CNN	0.277	0.844	0.718	11.0°	0.827	0.281	0.843	0.718	10.9°	0.828
	CRNN	0.213	0.879	0.770	11.3°	0.847	0.221	0.876	0.768	12.6°	0.844
DOA	CNN	–	–	–	13.3°	–	–	–	–	13.1°	–
	CRNN	–	–	–	11.9°	–	–	–	–	11.9°	–
DOA-NT	CNN	–	–	–	14.7°	–	–	–	–	14.5°	–
	CRNN	–	–	–	14.0°	–	–	–	–	14.3°	–
Two-Stage	CNN	0.251	0.862	0.749	10.9°	0.832	0.248	0.864	0.756	10.8°	0.832
	CRNN	**0.167**	**0.909**	**0.819**	**9.85°**	**0.863**	**0.181**	**0.898**	**0.800**	**9.84°**	**0.857**

- **DOA-NT**, is the same as DOA method except for not transferring CNN layers. Both DOA and DOA-NT only estimate DOAs.

All of the above-mentioned methods are evaluated on both CNNs and CRNNs. The CNN has the same architecture as the CRNN but without the recurrent layer. Furthermore, microphone array signals do not need extra encoding processes. It is more convenient to use in practice, whereas the encoding of FOA may contain extra spatial information. Therefore, it is worthwhile to evaluate these methods with both the microphone array and FOA data.

3.3. Hyper-parameters

To extract the input features, the sample rate of STFT is set to 32kHz. A 1024-point Hanning window with a hop size of 320 points is utilized. In the DCASE 2019 Task 3 dataset, the largest microphone distance is 4.82cm [25]. According to Section 2.1.2, the number of mel-band filters and the delays of GCC-PHAT is set to be 64. For 4 channels of signals, up to 10 input channels of signals are sent to the network. The audio clips are segmented to have a fixed length of 2 seconds with a 1-second overlap for training. The learning rate is set to 0.001 for the first 30 epochs and is then decayed by 10% every epoch. The final results are calculated after 50 epochs.

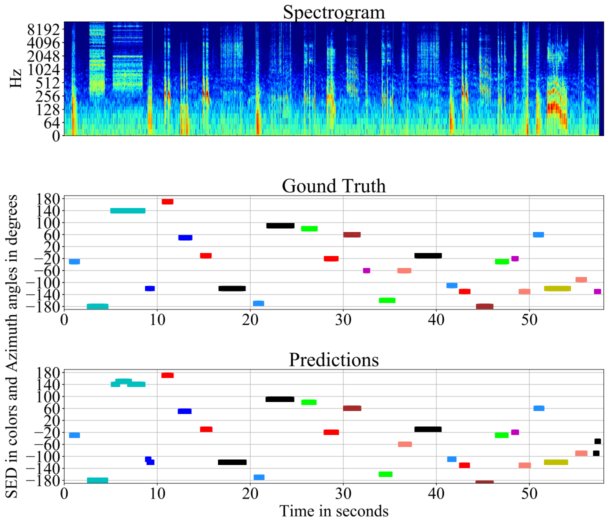


Figure 2: SED and DOAE Azimuth results for proposed two-stage method. Different colors indicate different classes of events.

3.4. Results

The experimental results are shown in Table 1. SELDnet with log mel and GCC-PHAT spectrograms as the input features was implemented first to compare with the baseline method. It can be seen from both microphone array data and FOA data that with log mel and GCC-PHAT spectrograms as the input features, SELDnet outperforms the baseline system using magnitude and phase spectrograms. Log mel spectrograms are more effective input features than magnitude spectrograms, not only due to their better performance, but they are also more compact. GCC-PHAT spectrograms, which mainly contain the time difference information, show their advantages over phase spectrograms. The results of DOA and DOA-NT show that with trained CNN layers transferred, DOA error is consistently lower than not transferring, which indicates that SED information contributes to the DOAE performance; it can also be observed that the convergence speed is much faster with CNN layers transferred. Comparing SELDnet with DOA-NT, it also shows that the joint training is better than the training of DOAE without CNN layers transferred, which also proves SED contributes to DOAE. The proposed two-stage method is presented in the end. The metrics scores are the best among all the methods. Compared with SELDnet, it indicates that the joint training of SED and DOAE degrades the performance of both. This two-stage method minimizes each loss individually, hence the network representation ability is enhanced for each sub-task, while the contribution from SED to DOAE is still preserved by transferring CNN layers to the DOAE branch.

Comparing microphone array data and FOA data, the results do not show FOA is better, which means FOA does not necessarily contain more spatial information than microphone array signals with four channels. On the other hand, in most cases, CRNNs perform better than CNNs, which indicates that long temporal information may be useful for both SED and DOAE. A visualization of SED and DOAE using the proposed method for one clip is shown in Fig. 2. It can be seen that most of the SED and DOAE predictions are accurate in both temporal and spatial dimensions.

4. CONCLUSION

Treating sound event detection and localization as a combined task is reasonable. In this paper, it shows that SED information can be used to improve the performance of DOAE. However, joint training of SED and DOAE degrades the performance of both. A two-stage polyphonic sound event detection and localization method is proposed to solve this problem. The proposed method uses log mel and GCC-PHAT spectrograms as the input features and has two branches of SED and DOAE. The SED branch is trained first, after which the trained feature layers are transferred to the DOAE branch. The DOAE branch then uses the SED ground truth as a mask to train DOAE. Experimental results show that the proposed method is able to enhance the network representation ability for each branch, while still keeping the contributions from SED to DOAE. The proposed method is shown to significantly outperform the baseline method.

5. ACKNOWLEDGMENT

This research was supported by EPSRC grant EP/N014111/1 “Making Sense of Sounds” and was partially supported by the H2020 Project entitled AudioCommons funded by the European Commission with Grand Agreement number 688382.

6. REFERENCES

[1] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational Analysis of Sound Scenes and Events*. Springer, 2018, ch. 1, pp. 3–12.

[2] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 2, pp. 379–393, Feb. 2018.

[3] K. Choi, G. Fazekas, and M. Sandler, "Automatic tagging using deep convolutional neural networks," in *17th International Society for Music Information Retrieval Conference*, 2016.

[4] Y. Xu, Q. Kong, Q. Huang, W. Wang, and M. D. Plumbley, "Convolutional gated recurrent neural network incorporating spatial features for audio tagging," in *International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 3461–3466.

[5] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, X. Favory, J. Pons, and X. Serra, "General-purpose tagging of Freesound audio with AudioSet labels: task description, dataset, and baseline," in *In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018.

[6] E. Çakır, T. Heittola, H. Huttunen, and T. Virtanen, "Polyphonic sound event detection using multi label deep neural networks," in *International Joint Conference on Neural Networks (IJCNN)*, 2015.

[7] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015.

[8] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[9] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, Mar. 2017.

[10] Q. Kong, Y. Cao, T. Iqbal, Y. Xu, W. Wang, and M. D. Plumbley, "Cross-task learning for audio tagging, sound event detection and spatial localization: Dcase 2019 baseline systems," *arXiv preprint arXiv:1904.03476*, 2019.

[11] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[12] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.

[13] M. Brandstein and D. Ward, *Microphone Arrays: Signal Processing Techniques and Applications*. Springer Science & Business Media, 2013, ch. 8, pp. 157–180.

[14] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, 1976.

[15] J. Scheuing and B. Yang, "Correlation-based TDOA-estimation for multiple sources in reverberant environments," in *Speech and Audio Processing in Adverse Environments*. Springer Berlin Heidelberg, 2008, pp. 381–416.

[16] X. Xiao, S. Zhao, X. Zhong, D. L. Jones, E. S. Chng, and H. Li, "A learning-based approach to direction of arrival estimation in noisy and reverberant environments," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[17] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza, "A neural network based algorithm for speaker localization in a multi-room environment," in *IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2016.

[18] N. Ma, T. May, and G. J. Brown, "Exploiting deep neural networks and head movements for robust binaural localization of multiple sources in reverberant environments," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 12, pp. 2444–2453, 2017.

[19] S. Chakrabarty and E. A. P. Habets, "Broadband DOA estimation using convolutional neural networks trained with noise signals," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017.

[20] S. Chakrabarty and E. A. P. Habets, "Multi-speaker localization using convolutional neural network trained with noise," *arXiv preprint arXiv:1712.04276*, 2017.

[21] W. He, P. Motlicek, and J. Odobez, "Deep neural networks for multiple speaker detection and localization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[22] E. L. Ferguson, S. B. Williams, and C. T. Jin, "Sound source localization in a multipath environment using convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[23] Y. Sun, J. Chen, C. Yuen, and S. Rahardja, "Indoor sound source localization with probabilistic neural network," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 8, pp. 6403–6413, 2018.

[24] S. Adavanne, A. Politis, and T. Virtanen, "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network," in *26th European Signal Processing Conference (EUSIPCO)*, 2018.

[25] http://dcase.community/challenge2019/task-sound-event-localization-and-detection.

[26] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, 2019.

[27] M. A. Gerzon, "Ambisonics in multichannel broadcasting and video," *Journal of the Audio Engineering Society*, vol. 33, no. 11, pp. 859–871, 1985.

[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015.

[29] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, pp. 162–178, 2016.

SONYC URBAN SOUND TAGGING (SONYC-UST): A MULTILABEL DATASET FROM AN URBAN ACOUSTIC SENSOR NETWORK

Mark Cartwright[1,2,3*], *Ana Elisa Mendez Mendez*[1], *Jason Cramer*[1], *Vincent Lostanlen*[1,2,4], *Graham Dove*[6], *Ho-Hsiang Wu*[1], *Justin Salamon*[5], *Oded Nov*[6], *and Juan Pablo Bello*[1,2,3]

[1] Music and Audio Resarch Lab, New York University, NY, USA
[2] Center for Urban Science and Progress, New York University, NY, USA
[3] Department of Computer Science and Engineering, New York University, NY, USA
[4] Cornell Lab of Ornithology, Cornell University, NY, USA
[5] Adobe Research, San Francisco, CA, USA
[6] Department of Technology Management and Innovation, New York University, NY, USA

ABSTRACT

SONYC Urban Sound Tagging (SONYC-UST) is a dataset for the development and evaluation of machine listening systems for real-world urban noise monitoring. It consists of 3068 audio recordings from the "Sounds of New York City" (SONYC) acoustic sensor network. Via the Zooniverse citizen science platform, volunteers tagged the presence of 23 fine-grained classes that were chosen in consultation with the New York City Department of Environmental Protection. These 23 fine-grained classes can be grouped into eight coarse-grained classes. In this work, we describe the collection of this dataset, metrics used to evaluate tagging systems, and the results of a simple baseline model.

Index Terms— Audio databases, Urban noise pollution, Sound event detection

1. INTRODUCTION

Noise pollution is a major concern for urban residents and has negative effects on residents' health [1, 2, 3] and learning [2, 4]. To mitigate the recurrence of harmful sounds, the City of New York employs a legal enforcement strategy guided by a "noise code". For example, jackhammers can only operate on weekdays; pet owners are held accountable for their animals' noises; ice cream trucks may only play their jingles while in motion; blasting a car horn is restricted to situations of imminent danger. After a city resident complains about noise, the New York City Department of Environmental Protection (DEP) sends an inspector to investigate the complaint. If the inspector is able to confirm that the offending noise violates the noise code, they incentivize the manager of the noise source to reduce their noise footprint in compliance with the code. Unfortunately, this complaint-driven enforcement approach results in a mitigation response biased to neighborhoods who complain the most, not necessarily the areas in which noise causes the most harm. In addition, due to the transient nature of sound, the offending noise source may have already ceased by the time an inspector arrives on site to investigate the complaint.

Sounds of New York City (SONYC) is a research project investigating data-driven approaches to mitigating urban noise pollution. One of its aims is to map the spatiotemporal distribution of noise at the scale of a megacity like New York City, in real time, and throughout multiple years. With such a map, city officials could better understand noise in the city; more effectively allocate city resources for mitigation; and develop informed mitigation strategies while alleviating the biases inherent to complaint-driven approaches. To this end, SONYC has designed an acoustic sensor for noise pollution monitoring that combines relatively high quality sound acquisition with a relatively low production cost [5]. Between 2016 and 2019, over 50 different sensors have been assembled and deployed in various areas of New York City.

Each SONYC sensor measures the sound pressure level (SPL) of its immediate vicinity, but it does not infer and report the causes of changes in SPL. From a perceptual standpoint, not all sources of outdoor noise are equally unpleasant, nor are they equally enforcible with respect to the noise code. Therefore, it is necessary to resort to computational methods for detection and classification of acoustic scenes and events (DCASE) in the context of automated noise pollution monitoring. To address this, the sensors have also been collecting non-contiguous 10 s audio recordings during deployment and have collectively gathered over 100 M recordings.

There are several attributes of urban sound event detection that make it a challenging task. Sound sources of interest are often far away from the sensors. Several sources of interest may occur simultaneously. Many sound classes seem quite similar, yet are distinct in the noise code and so should be identified as such. Many other distractor sounds occur within urban sound recordings. And lastly, the acoustic environment changes by location and by time within seasonal cycles. Due to the complexity of this problem, it is important to evaluate machine listening systems for monitoring urban noise in realistic scenarios, using actual recordings from urban noise sensors and a label space that matches the needs of city agencies.

In this article, we present the SONYC Urban Sound Tagging (SONYC-UST) dataset[1], which contains 3068 annotated 10 s recordings from the SONYC acoustic sensor network and which served as the dataset for the DCASE 2019 Urban Sound Tagging Challenge[2]. Each recording has been annotated using a set of 23 "tags", which was developed in coordination with the New York City Department of Environmental Protection (DEP) and represents

*This work was partially funded by National Science Foundation awards 1633259 and 1544753

[1]Download the data at https://doi.org/10.5281/zenodo.3338310
[2]http://dcase.community/challenge2019/task-urban-sound-tagging

https://doi.org/10.33682/j5zw-2t88

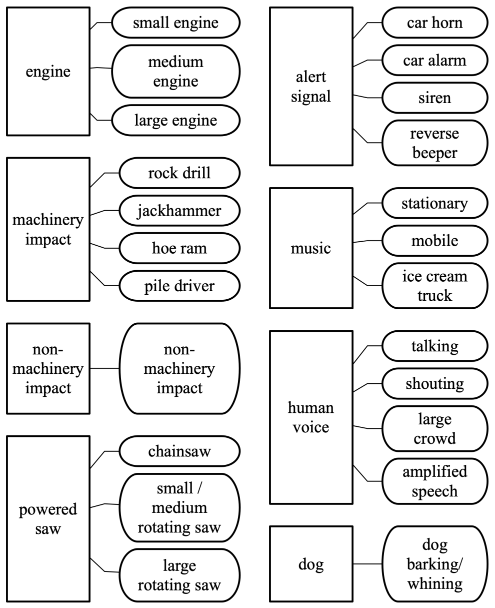


Figure 1: Hierarchical taxonomy of the SONYC Urban Sound Tagging (SONYC-UST) dataset. Rectangular and round boxes respectively denote coarse and fine urban sound tags.

many of the frequent causes of noise complaints in New York City.

Existing datasets for urban noise monitoring do not accurately represent the problem of urban noise monitoring. The freefield1010 [6], UrbanSound [7], UrbanSound8k, [7], and Urban-SED [8] datasets contain recordings curated from Freesound [9] rather than recorded in a realistic noise monitoring scenario. In addition, these datasets are multi-class, in which only the predominant sound class is labeled. The exception is Urban-SED [8], which does have strong, multi-label annotations, but it is a synthetic dataset that is not representative of actual urban soundscapes. The TUT Sound Events 2016 [10, 11, 12] and 2017 [13, 14] datsets consists of audio recordings in real urban environments as well as providing strong, multi-label annotations. However, these datasets have label sets limited to human presence and traffic, and their spatiotemporal context is limited to a handful of times and locations. SONYC-UST addresses these limitations by providing recordings from urban noise sensors across a variety of times and locations, and by more closely matching the label set to the needs of noise enforcement agencies.

2. SONYC-UST TAXONOMY

Through consultation with the New York Department of Environmental Protection (DEP) and the New York noise code, we constructed a small, two-level urban sound taxonomy (see Figure 1) consisting of 8 coarse-level and 23 fine-level sound categories, e.g., the coarse *alert signals* category contains four fine-level categories: *reverse beeper*, *car alarm*, *car horn*, *siren*. Unlike the Urban Sound Taxonomy [7], this taxonomy is not intended to provide a framework for exhaustive description of urban sounds. Instead, it was scoped to provide actionable information to the DEP, while also being understandable and manageable for novice annotators. The chosen sound categories map to categories of interest in the noise code; they were limited to those that seem likely discernible by novice annotators; and we kept the number of categories small enough so that they can all be visible at once in an annotation interface.

3. DATA COLLECTION

The SONYC acoustic sensor network consists of more than 50 acoustic sensors deployed around New York City and has recorded over 100M 10-second audio clips since its launch in 2016. The sensors are located in the Manhattan, Brooklyn, and Queens boroughs of New York, with the highest concentration around New York University's Manhattan campus. To maintain the privacy of bystanders' conversations, the network's sensors are positioned for far-field recording, 15–25 feet above the ground, and record audio clips at random intervals, rather than continuously.

To annotate the sensor recordings, we launched an annotation campaign on Zooniverse [15, 16], the largest citizen-science platform. In a previous study comparing multiple types of weak annotation tasks, we found that full multi-label annotation (i.e., an annotation task in which all classes are annotated at once by each annotator) with at least three annotators per recording resulted in high quality annotations and high throughput with citizen science volunteers [17]. In another previous study, we found that spectrogram visualizations aided annotators in producing high quality annotations [18]. Given these findings, we configured the annotation task as a multi-label, weak annotation (i.e., tagging) task in which the annotators were presented with a spectrogram visualization of the audio clip along with the audio playback.

After presenting volunteers with instructions explaining the task and a field guide describing the SONYC-UST classes, we asked them to annotate the presence of all of the fine-level classes in a recording. For every coarse-level class (e.g., *alert signal*) we also included a fine-level *other/unknown* class (e.g., *other/unknown alert signal*) with the goal of capturing an annotator's uncertainty in a fine-level tag while still annotating the coarse-level class. If an annotator marked a sound class as present in the recording, they were also asked to annotate the proximity of the sound event (*near*, *far*, *not sure*). Volunteers could annotate as many recordings as were available.

Manually annotating all 100M+ of the unlabeled sensor recordings is not feasible, but annotating a random sample is not efficient since many of them may not contain sound events of interest. To address this, we sample sensor recordings that are most similar to a small set of exemplary clips for each sound class in our taxonomy. The exemplary clips were curated from YouTube and selected based on the presence of the target class in the audio along with visual confirmation from the video. Similarity to the exemplary clips was computed using a distance function D, which compares a sensor recording to M exemplary clips for a particular class:

$$D(\boldsymbol{X}^{(c)}, \boldsymbol{y}_n) = \sum_{m} \frac{1}{K_m} \sum_{k}^{K_m} \min_{j} d(x_{m,k}^{(c)}, y_{n,j})^2 \tag{1}$$

where $x_{m,k}^{(c)}$ is the k^{th} VGGish [19] embedding frame of the m^{th} example clip (from class c) with K_m frames, $\boldsymbol{X}^{(c)}$ represents the M exemplary clips from class c, $y_{n,j}$ is the j^{th} VGGish embedding frame of the n^{th} sensor recording $\boldsymbol{y}_n$, and d is the Euclidean distance function.

The SONYC-UST dataset contains annotated train, validate, and test splits (2351 / 443 / 274 recordings respectively). We selected these splits such that recordings from the same sensors would

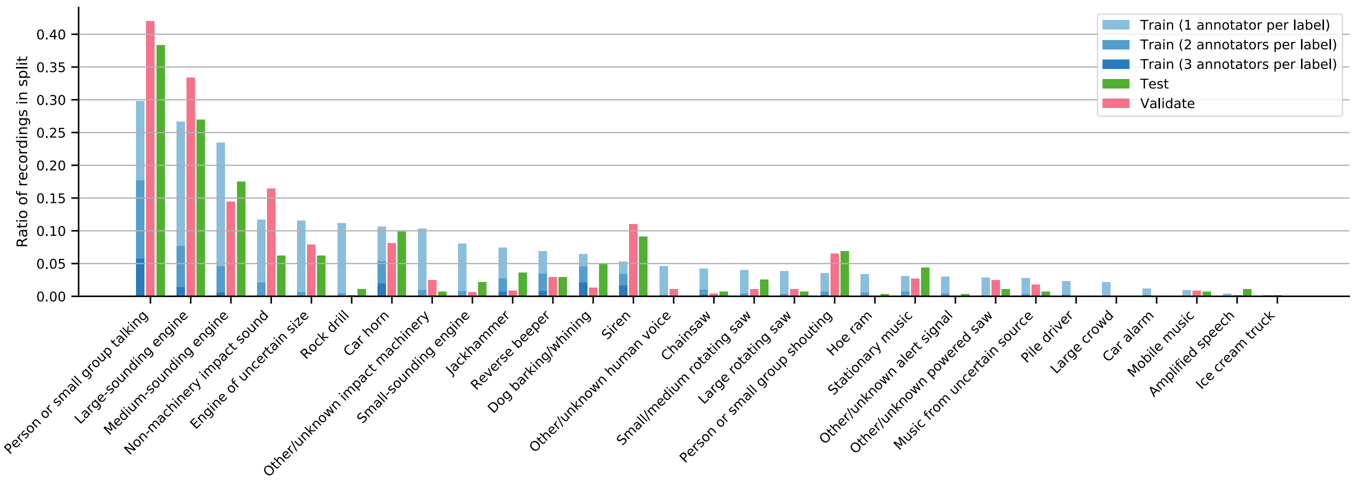


Figure 2: SONYC-UST tag distribution normalized for each split, in decreasing order of frequency in the train split. The shades of blue indicate how many annotators tagged the class in a training set recording, i.e. darker shades of blue indicate higher annotator agreement.

not appear in both the train and validate sets, and such that the distributions of citizen-science-provided labels were similar for both the train and validate sets (see Figure 2). While doing so, we considered a class present in a sample if at least one of the three volunteers annotated it as such. 35 sensors were assigned to the training set and 8 sensors assigned to the validate set. Unlike the train/validate sets, the test set is not disjoint in terms of sensors, but rather it is disjoint in time—all recordings in the test set are posterior to those in the train/validate sets. This allows us to evaluate model generalization to known locations at unseen times.

In addition to the crowdsourced annotations from Zooniverse, we include "SONYC-team-verified" labels for both the validation and test splits. To create verified labels, we first distributed recordings based on coarse-level sound category to members of the SONYC research team for labeling. To determine whether a recording belonged to a specific category for the validation process, we selected those that had been annotated by at least one volunteer. Next, two members of our team labeled each category independently. Once each member had finished labeling their assigned categories, the two annotators for each class discussed and resolved label disagreements that occurred during the independent annotation process. We use these agreed-upon "SONYC-team-verified" labels as the "ground truth" when evaluating models. We also use these labels to evaluate the annotations from Zooniverse, aggregated using minority vote, which we have previously shown to be an effective aggregation strategy in this context [17]. To aggregate with minority vote, we simply count a positive for a tag if at least one annotator has labeled the audio clip with that tag. In Table 1, we present annotation accuracy results using the metrics described Section 4. When examining the class-wise F1 scores, we see that crowdsourced annotations score well against the ground-truth for many classes, but it seems the Zooniverse annotators have difficulty identifying impact sounds and powered saws, especially when discriminating between fine-level classes.

In the SONYC-UST dataset, we include the Zooniverse volunteers' fine-level multi-label class-presence and proximity annotations for all the audio recordings in all three data splits. We also provide the SONYC-team-verified multi-label class-presence annotations and the agreed-upon ground-truth class-presence labels for the validate and test sets. All annotations also include identifiers for both the annotator and the sensor from which the clip was recorded. The coarse-level indicators of class presence are also included and are computed by logical disjunction over the fine-level class-presence tags associated with the coarse-level category.

Estimator:	Annotators				Baseline Model			
Split:	Validate		Test		Validate		Test	
Level:	F	C	F	C	F	C	F	C
Overall								
AUPRC	.73	.87	.75	.90	.67	.77	.62	.76
F1@0.5	.68	.83	.68	.84	.50	.70	.43	.67
Class F1@0.5								
Engine	.64	.94	.64	.94	.37	.79	.29	.76
Mech. imp.	.25	.24	.32	.35	.29	.36	.62	.58
Non-mech. imp.	.49	.49	.60	.60	.05	.02	.00	.11
Powered saw	.47	.70	.28	.62	.30	.66	.45	.66
Alert signal	.88	.95	.87	.92	.48	.67	.35	.48
Music	.59	.76	.52	.76	.07	.07	.00	.00
Human voice	.82	.95	.82	.96	.77	.84	.63	.77
Dog	.74	.74	.96	.96	.00	.00	.66	.66

Table 1: The performance of the Zooniverse annotations (using minority vote aggregation) and the baseline classifier as compared the the ground-truth annotations for both validate and test splits on the coarse (C) and fine (F) levels. AUPRC and F1 are both micro-averaged.

4. MULTILABEL CLASSIFICATION METRICS

Due to the presence of *other/unknown* tags, SONYC-UST has an incomplete ground truth at the fine taxonomical level. Such incompleteness poses a problem for evaluating multilabel classifiers. We propose a pragmatic solution to this problem; the guiding idea behind our solution is to evaluate the prediction at the fine level only when possible, and fall back to the coarse level if necessary.

Let a coarse-level category contain K fine-level tags. We denote by $t_1 \dots t_K$ the indicators of presence of these tags in the ground truth. For $k \in \{1 \dots K\}$, the integer t_k is equal to 1 if the fine-level tag k is present in the ground truth and equal to 0 otherwise. Furthermore, we denote by t_0 the indicator of presence of the *other/unknown* tag in the ground truth for the coarse category at hand. In the following, we adopt the bar notation $\overline{t_k}$ as a shorthand for the logical negation $(1 - t_k)$. Whereas the fine-level composition of the coarse category cannot be assessed with certainty, taking the product of all integers $\overline{t_0} \dots \overline{t_k}$ yields a coarse indicator of *certain absence*, equal to 1 if and only if none of the fine-level tags is present, even the uncertain one. This operation of tag coarsening allows to evaluate any prediction $\boldsymbol{y}$ against the ground truth $\boldsymbol{t}$. In each coarse category, the comparison of $\boldsymbol{y}$ and $\boldsymbol{t}$ results in, either, a true positive (TP), a false positive (FP), or a false negative (FN):

$$\begin{aligned}
\mathrm{TP}_{\text{coarse}} &= \left(1 - \prod_{k=0}^{K} \overline{t_k}\right) \times \left(1 - \prod_{k=0}^{K} \overline{y_k}\right) \\
\mathrm{FP}_{\text{coarse}} &= \left(\prod_{k=0}^{K} \overline{t_k}\right) \times \left(1 - \prod_{k=0}^{K} \overline{y_k}\right) \\
\mathrm{FN}_{\text{coarse}} &= \left(1 - \prod_{k=0}^{K} \overline{t_k}\right) \times \left(\prod_{k=0}^{K} \overline{y_k}\right). \qquad (2)
\end{aligned}$$

The three numbers above are equal to zero or one, and sum to one in each coarse category. Although they are resilient to the incompleteness of tags, this comes at the cost of them being insensitive to permutations of complete fine-level tags within the same coarse category. Therefore, we propose the alternative definitions below:

$$\begin{aligned}
\mathrm{TP}_{\text{fine}} &= \left(\sum_{k=1}^{K} t_k y_k\right) + t_0 \times \left(1 - \prod_{k=1}^{K} t_k y_k\right) \times \left(1 - \prod_{k=0}^{K} \overline{y_k}\right), \\
\mathrm{FP}_{\text{fine}} &= \overline{t_0} \times \left(\sum_{k=1}^{K} \overline{t_k} y_k\right) + \overline{t_0} y_0 \times \left(\prod_{k=1}^{K} \overline{t_k}\right) \times \left(1 - \prod_{k=1}^{K} y_k\right), \\
\mathrm{FN}_{\text{fine}} &= \left(\sum_{k=1}^{K} t_k \overline{y_k}\right) + t_0 \times \left(\prod_{k=1}^{K} \overline{t_k}\right) \times \left(\prod_{k=0}^{K} \overline{y_k}\right). \qquad (3)
\end{aligned}$$

In contrast to their coarse counterparts, these counters range from 0 to K. In the simple case where the ground truth is complete (i.e., $t_0 = 0$), they boil down to a one-to-one comparison of complete fine-level predicted tags y_k with the complete fine-level ground truth tags t_k, with the incomplete prediction y_0 being counted as a false positive if present. However, if the ground truth contains the incomplete tag (i.e., $t_0 = 1$), $\mathrm{FP}_{\text{fine}}$ falls to zero. If, in addition, no complete fine-level ground truth tag t_k matches a complete fine-level prediction (i.e., $t_k y_k = 0$ for all $k > 0$), then the number of true positives is set to one if the coarsened predicted tag is present (i.e., $y_k = 0$ for any $k \geq 0$) and zero otherwise. Lastly, if the coarsened predicted tag is absent (i.e., $y_k = 0$ for all $k \geq 0$) and if the ground truth does not contain any complete tag (i.e., $t_k = 0$ for all $k > 0$), then the number of false negatives is set to t_0.

For example, if a small engine is present in the ground truth and absent in the prediction but an *other/unknown engine* is predicted, then it is a true positive in the coarse-grained sense, but a false negative in the fine-grained sense. However, if a small engine is absent in the ground truth and present in the prediction, then the outcome of the evaluation will depend on the completeness of the ground truth for the coarse category of engines. If this coarse category is complete (i.e. if the tag "engine of uncertain size" is absent from the ground truth), then we may evaluate the small engine tag at the fine level, and count it as a false positive. Conversely, if the coarse category of engines is incomplete (i.e. the tag "engine of uncertain size" is present in the ground truth), then we fall back to coarse-level evaluation for the sample at hand, and count the small engine prediction as a true positive, in aggregation with potential predictions of medium engines and large engines.

In each coarse category, these integer counts can then be translated into well-known information retrieval metrics: precision, recall, F1 score, and area under the precision recall curve (AUPRC). Furthermore, they can be micro-averaged across coarse categories to yield an overall F1 score and an overall AUPRC. The repository of our baseline system contains an open-source implementation of these metrics, both for "coarse" and "fine" formulas[3].

5. BASELINE SYSTEM

For the baseline classifier (cf. footnote 3) we use a multi-label logistic regression model. The model uses VGGish embeddings [19] as its input representation, which are computed from non-overlapping 0.96-second windows, giving us ten frames of 128-dimensional embeddings for each clip in our dataset. We train the model at the frame-level and use the weak tags from each audio clip as the targets for each of the 10 frames in a clip. To aggregate the crowdsourced annotations for training, we count a positive for a tag if at least one annotator has labeled the audio clip with that tag, i.e. minority vote.

We trained the model using stochastic gradient descent to minimize binary cross-entropy loss. To train the model to predict fine-level tags, the loss is modified such that if "other/unknown" is annotated for a particular coarse tag, the loss for the fine tags corresponding to this coarse tag is masked out. We use early stopping based on the validation set loss to mitigate overfitting. We trained one model to only predict fine-level tags, and we trained another model to only predict coarse-level tags.

For clip-level inference, we predict tags at the frame level and take the average of output tag probabilities as the clip-level tag probabilities. The resulting summary and class-wise metrics are presented in Table 1. Overall the baseline models achieved an AUPRC of 0.62 and 0.76 on the test split's fine and coarse levels respectively, and performed poorly on *music* and *non-machinery impact* sounds, leaving considerable room for improvement.

6. CONCLUSION

SONYC-UST is a multi-label dataset for urban sound tagging, recorded from an urban acoustic sensor network and annotated by crowdsourced volunteers. This dataset addresses deficiencies in the current set of urban sound datasets by providing real-world recordings and a label set that more closely matches the needs of city agencies in charge of noise abatement. In this work, we present the process used to collect this data; a taxonomy of urban sound tags informed by the New York City noise code and consultation with noise enforcement agents; metrics to evaluate tagging systems with uncertain ground-truth data; and a baseline model demonstrating that this is a challenging task with considerable room for improvement. We hope this dataset will encourage researchers to focus on this problem and advance the state of the art in urban sound event detection, helping build tools to make cities quieter.

[3]https://github.com/sonyc-project/urban-sound-tagging-baseline

7. REFERENCES

[1] M. S. Hammer, T. K. Swinburn, and R. L. Neitzel, "Environmental noise pollution in the united states: developing an effective public health response," *Environmental Health Perspectives*, vol. 122, no. 2, pp. 115–119, 2013.

[2] A. Bronzaft, "Neighborhood noise and its consequences," *Survey Research Unit, School of Public Affairs, Baruch College, New York*, 2007.

[3] World Health Organization, "Burden of disease from environmental noise: Quantification of healthy life years lost in europe," in *Burden of disease from environmental noise: Quantification of healthy life years lost in Europe*, 2011, pp. 126–126.

[4] M. Basner, W. Babisch, A. Davis, M. Brink, C. Clark, S. Janssen, and S. Stansfeld, "Auditory and non-auditory effects of noise on health," *The Lancet*, vol. 383, no. 9925, pp. 1325–1332, 2014.

[5] C. Mydlarz, M. Sharma, Y. Lockerman, B. Steers, C. Silva, and J. P. Bello, "The life of a new york city noise sensor network," *Sensors*, vol. 19, no. 6, p. 1415, 2019.

[6] D. Stowell and M. Plumbley, "An open dataset for research on audio field recording archives: freefield1010," in *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*, Jan 2014. [Online]. Available: http://www.aes.org/e-lib/browse.cfm?elib=17095

[7] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 1041–1044.

[8] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 344–348.

[9] F. Font, G. Roma, and X. Serra, "Freesound technical demo," in *Proceedings of the 21st ACM international conference on Multimedia*. ACM, 2013, pp. 411–412.

[10] A. Mesaros, T. Heittola, and T. Virtanen, "Tut database for acoustic scene classification and sound event detection," in *2016 24th European Signal Processing Conference (EUSIPCO)*. IEEE, 2016, pp. 1128–1132.

[11] ——, "Tut sound events 2016, development dataset," 2016. [Online]. Available: https://zenodo.org/record/45759

[12] ——, "Tut sound events 2016, evaluation dataset," 2017. [Online]. Available: https://zenodo.org/record/996424

[13] ——, "Tut sound events 2017, development dataset," 2017. [Online]. Available: https://zenodo.org/record/814831

[14] ——, "Tut sound events 2017, evaluation dataset," 2017. [Online]. Available: https://zenodo.org/record/1040179

[15] R. Simpson, K. R. Page, and D. De Roure, "Zooniverse: Observing the world's largest citizen science platform," in *Proceedings of the 23rd International Conference on World Wide Web*, ser. WWW '14 Companion. New York, NY, USA: ACM, 2014, pp. 1049–1054.

[16] "Zooniverse." [Online]. Available: www.zooniverse.org

[17] M. Cartwright, G. Dove, A. E. Méndez Méndez, J. P. Bello, and O. Nov, "Crowdsourcing multi-label audio annotation tasks with citizen scientists," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, 2019, p. 292.

[18] M. Cartwright, A. Seals, J. Salamon, A. Williams, S. Mikloska, D. MacConnell, E. Law, J. P. Bello, and O. Nov, "Seeing sound: Investigating the effects of visualizations and complexity on crowdsourced audio annotations," *Proceedings of the ACM on Human-Computer Interaction*, vol. 1, no. CSCW, p. 29, 2017.

[19] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "Cnn architectures for large-scale audio classification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. [Online]. Available: https://arxiv.org/abs/1609.09430

NON-NEGATIVE MATRIX FACTORIZATION-CONVOLUTIONAL NEURAL NETWORK (NMF-CNN) FOR SOUND EVENT DETECTION

Teck Kai Chan,[1,2]*, *Cheng Siong Chin*[1]

[1]Newcastle University Singapore
Faculty of Science, Agriculture, and Engineering
Singapore 599493
{t.k.chan2, cheng.chin}@newcastle.ac.uk

Ye Li[2]

[2]Visenti Pte Ltd (A Brand of Xylem)
3A International Business Park
Singapore 609935
ye.li@xyleminc.com

ABSTRACT

The main scientific question of this year DCASE challenge, Task 4 - Sound Event Detection in Domestic Environments, is to investigate the types of data (strongly labeled synthetic data, weakly labeled data, unlabeled in domain data) required to achieve the best performing system. In this paper, we proposed a deep learning model that integrates Non-Negative Matrix Factorization (NMF) with Convolutional Neural Network (CNN). The key idea of such integration is to use NMF to provide an approximate strong label to the weakly labeled data. Such integration was able to achieve a higher event-based F1-score as compared to the baseline system (Evaluation Dataset: 30.39% vs. 23.7%, Validation Dataset: 31% vs. 25.8%). By comparing the validation results with other participants, the proposed system was ranked 8th among 19 teams (inclusive of the baseline system) in this year Task 4 challenge.

Index Terms— Non-negative matrix, convolutional neural network, DCASE 2019

1. INTRODUCTION

The primary objective of a Sound Event Detection (SED) system is to identify the type of sound source present in an audio clip or recording and return the onset and offset of the identified source. Such a system has great potential in several domains, such as activity monitoring, environmental context understanding, and multimedia event detection [1], [2]. However, there are several challenges associated with SED in real-life scenarios.

Firstly, in real-life scenarios, different sound events can coincide [2]. Secondly, the presence of background noise could complicate the identification of sound event within a particular time frame [3]. This problem is further aggravated when the noise is the prominent sound source resulting in a low Signal to Noise Ratio (SNR). Thirdly, each event class is made up of different sound sources, e.g., a dog bark sound event can be produced from several breeds of dogs with different acoustic characteristics [1]. Finally, to achieve the best results, SED detection algorithm may require strongly labeled data where the occurrence of each event with its onset and offset are known with certainty during the model development phase.

While such data are useful, collecting them is often time-consuming, and sizes of such dataset are often limited to minutes or a few hours [3], [4]. In certain scenarios such as an approaching vehicle, the onset and offset time is ambiguous due to the fade in and fade out effect [5] and is subjective to the person labeling the event.

On the other hand, there exist a substantial amount of data known as the weakly labeled data where only the occurrence of an event is known without any offset or onset annotations. While it seems like the core information is missing, previous implementations proposed in the annual Detection and Classification of Acoustic Scenes and Events (DCASE) challenge that utilized only weakly labeled data had achieved a certain level of success [6]-[8]. Although a large number of different SED systems were proposed in the past, a majority of them were mainly based on Gaussian Mixture Model (GMM) [9], Hidden Markov Model (HMM) [10] or the use of dictionaries constructed using NMF [11-13]. However, due to the rising success of deep learning in other domains [14-17], deep learning for SED development is now a norm and has been shown to perform slightly better than established methods [1]. Riding on the success of deep learning, this paper proposes a deep learning model that integrates NMF and CNN which can provide an approximate strong label to the weakly labeled data. Results have shown that the proposed system achieved a much higher event based F1-score as compared to the baseline system (Evaluation Dataset: 30.39% vs. 23.7%, Validation Dataset: 31% vs. 25.8%) and by comparing the validation results with other participants, the proposed system was ranked 8th among 19 teams (inclusive of the baseline system) in this year Task 4 challenge.

* This work was supported by the Economic Development Board-Industrial Postgraduate Programme (EDB-IPP) of Singapore under Grant BH180750 with Visenti Pte. Ltd.

https://doi.org/10.33682/50ef-dx29

2. RELATED WORK

In recent years, SED development has been overwhelmed with the use of deep learning algorithms particularly the use of CNN or Convolutional Recurrent Neural Network (CRNN). This phenomenon was also reflected in the 2018 and 2019 DCASE Task 4 challenge, where a large group of participants proposed the use of CRNN. As discussed in [1], CNN has the benefit of learning filters that are shifted in both time and frequency while Recurrent Neural Network (RNN) has a benefit of integrating information from the earlier time windows. Thus, a combined architecture has the potential to benefit from two different approaches that suggest its popularity.

The CRNN architecture proposed by Cakir et al. [1] first extracted features through multiple convolutional layers (with small filters spanning both time and frequency) and pooling in the frequency domain. The features were then fed to recurrent layers, whose features were used to obtain event activity probabilities through a feedforward fully connected layer. Evaluation over four different datasets had also shown that such a method has a better performance as compared to CNN, RNN and other established SED system. However, such a system would require a large amount of annotated data for training.

Lu [8] proposed the use of a Mean Teacher Convolution System that won the DCASE Task 4 challenge with an F1 score of 32.4%. In their system, context gating was used to emphasize the important parts of audio features in frames axis. Mean-Teacher semi-supervised method was then applied to exploit the availability of unlabeled data to average the model weights over training steps. Although this system won the 2018 challenge, there is still a large room for improvement.

3. SYSTEM OVERVIEW

3.1. Audio Processing

In this system, training inputs are mel-frequency scaled. This is because they can provide a reasonably good representation of the signal's spectral properties. At the same time, they also provide reasonably high inter-class variability to allow class discrimination by many different machine learning approaches [18].

In this paper, audio clips were first resampled to 32 kHz that were suggested to contain the most energies [19]. Moreover, segments containing higher frequency may not be useful for event detection in daily life [8].

A short-time fast Fourier transform with a Hanning window size of 1024 samples (32 ms) and a hop size of 500 samples (15.6 ms) was used to tabulate the spectrogram. After that, a mel filter bank of 64 and bandpass filter of 50 Hz to 14 kHz was applied to obtain the mel spectrogram. Finally, a logarithm operation was applied to obtain the log-mel spectrogram to use be used as input to the training model..

3.2. Non-Negative Matrix Factorization

The NMF popularized by Lee and Seung [20] is an effective method to decompose a non-negative $L \times N$ matrix M into two non-negative matrices, W and H of sizes $L \times R$ and $R \times N$ respectively where R is the number of components. The linear combination of W and H produces an approximated M and can be represented as

$$M \approx WH \tag{1}$$

W can be interpreted as the dictionary matrix and H can be interpreted as the activation matrix. These two matrices can be randomly initialized and updated through the multiplicative rule given as [20] to produce an optimized set of W and H. The updating procedure can be terminated when any further updating produces no improvement or when the difference of M and WH is below a user-defined threshold.

$$W \leftarrow W \otimes \frac{W^T \frac{M}{WH}}{W^T 1} \tag{2}$$

$$H \leftarrow H \otimes \frac{\frac{M}{WH} H^T}{1 H^T} \tag{3}$$

W is commonly extracted on isolated events to form a dictionary and SED is performed by applying a threshold on the activation matrix obtained from the decomposition of the test data [12]. Since NMF only works on non-negative matrix, it was applied on the mel spectrogram prior to the logarithm operation. Thus, M represents the mel spectrogram with L as the number of mel bins and N as the number of frames. In this paper, instead of consolidating W to form the dictionary, we find the H to indicate which frames of each audio clip are activated (above a pre-defined threshold) to label the weakly labeled data so that the weakly labeled data becomes an approximated strongly labeled data. If the clip contains multiple events, then those activated frames are deemed to contain all the sound events.

3.3. Convolutional Neural Network

The CNN used in this system is modified based on the one proposed in [19]. Kong et al. [19] proposed four different CNN with a different number of layers and pooling operators and found that the 9 layers CNN with max-pooling operator achieved the best performance. In this paper, we are interested in finding out whether with the inclusion of NMF, will a shallower CNN produce a comparable or even a better result. In this paper, a 5 layers CNN with max-pooling operator is proposed. In this architecture, the 5 layers consist of 1 input layer and 4 convolutional layers of kernel size 5 x 5 with a padding size of 2 x 2 and strides 1 x 1. This architecture is similar to Kong et al. [19] except for the kernel size and the number of layers as shown in Table 1.

Table 1. CNN architectures

Proposed	**Kong [19]**
Input : log-mel spectrogram	
$\begin{pmatrix} 5\times5@64 \\ BN, ReLU \end{pmatrix}$	$\begin{pmatrix} 3\times3@64 \\ BN, ReLU \end{pmatrix}\times2$
2 x 2 Max Pooling	
$\begin{pmatrix} 5\times5@128 \\ BN, ReLU \end{pmatrix}$	$\begin{pmatrix} 3\times3@128 \\ BN, ReLU \end{pmatrix}\times2$
2 x 2 Max Pooling	
$\begin{pmatrix} 5\times5@256 \\ BN, ReLU \end{pmatrix}$	$\begin{pmatrix} 3\times3@256 \\ BN, ReLU \end{pmatrix}\times2$
2 x 2 Max Pooling	
$\begin{pmatrix} 5\times5@512 \\ BN, ReLU \end{pmatrix}$	$\begin{pmatrix} 3\times3@512 \\ BN, ReLU \end{pmatrix}\times2$

For both architectures, Binary Cross Entropy is adopted as the loss function which is similar to the loss function adopted in [19] given as

$$l_{BCE(p,y)} = \sum_{k=1}^{K} [y_k \ln(p_k) + (1-y_k)\ln(1-p_k)] \quad (4)$$

3.4. System Flow

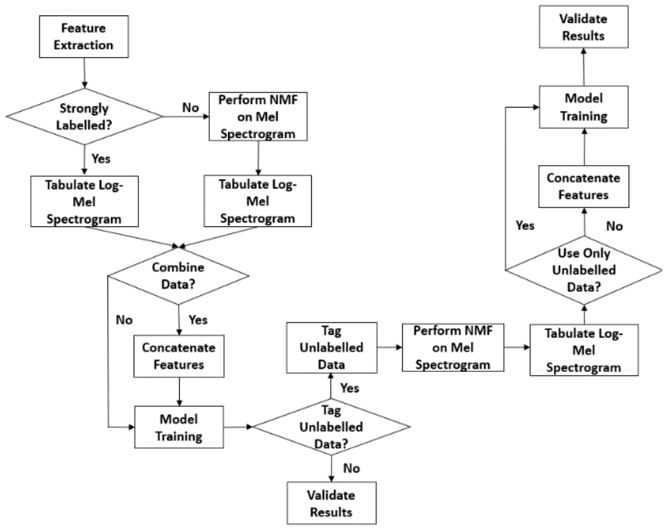


Fig. 1. Flowchart of proposed architecture

In this year DCASE challenge, Task 4 - Sound Event Detection In Domestic Environments, is specifically organized to investigate the types of data (strongly labeled synthetic data, weakly labeled data, unlabeled in domain data) required to achieve the best performing system. Therefore, the flow of the proposed system depends on the types of data used as seen in Fig. 1. While strongly labeled and weakly labeled data can be used readily, unlabeled data require a model to be trained in advance so that its content can be tagged and be used as training data. Example, if the user is keen to use all the data given, NMF will be applied to weakly labeled data to produce an approximated strongly labeled data. The log-mel spectrograms tabulated from both the approximated strongly labeled data and the actual strongly labeled data will be combined and used as input to the CNN. The model trained will be used to tag the events in the unlabeled data. Similar to weakly labeled data, NMF is applied to tagged unlabeled data prior to the calculation of log-mel spectrogram. These newly calculated log-mel spectrogram will be combined with previous calculated log-mel spectrograms and used as input to train a new model.

4. RESULTS AND DISCUSSION

Based on the proposed system flow, we tested the accuracy of our proposed architecture using the different combination of data on the given evaluation dataset that is a mixture of DCASE 2018 task 4 test set, and evaluation set consisting of 1168 audio clips with 4093 events.

Based on the results shown in Table 2, the model trained using both weakly labeled data and synthetic data achieved the highest accuracy as compared to using other combinations of data. It is surprising to find that strongly labeled synthetic data was not able to achieve higher accuracy than weakly labeled data. Whereas, a combination of data can increase the accuracy of the model.

On the other hand, results have shown that using only unlabeled in domain data or training a model with the inclusion of unlabeled in domain data labeled using different models, accuracy decreases. Furthermore, by comparing the proposed model results with Kong et al. [19] model and baseline model, it shows that although the proposed model can achieve a better event based F1 score, it has a lower segment based F1-score as compared to Kong et al. [19]. These two phenomenon could be due to the way how NMF was utilized. In this system, NMF was used to find H that indicates when the event was activated for the calculated H of certain frames were above a predefined threshold. However, if the clip contains multiple events, then NMF will indicate that those frames above a predefined threshold belong to all the events present in the audio. As such, it affected the quality of unlabeled data being labeled which resulted in a decrease in accuracy when unlabeled data is included and also resulted in a lower segment accuracy. Therefore, it may be worthwhile to investigate the use of source separation before the application of NMF.

The best four models (trained using C1, C3, C5, C7 as described in Table 2) were submitted to the challenge where

Table 2. F1-Score using different types of data, C1-Weakly Labeled Data, C2- Strongly Labeled Synthetic Data, C3- Weakly Labeled and Strongly Labeled Synthetic Data, C4- Unlabeled Data (labeled using model with F1 29.73%), C5- Weakly Labeled Data and Unlabeled Data (labeled using model with F1 29.73%), C6- Unlabeled Data (labeled using model with F1 30.39%), C7- Weakly Labeled, Strongly Labeled Synthetic Data and Unlabeled Data (labeled using model with F1 30.39%)

F1-Score	Type of Data	C1	C2	C3	C4	C5	C6	C7	Kong et al. [19]	Baseline [21]
Event Based F1	Evaluation Dataset	29.73%	15.27%	**30.39%**	25.47%	27.2%	26.64%	27.84%	24.1%	23.7%
Segment Based F1		55.79%	43.59%	57.66%	45.88%	48.52%	47.13%	50.92%	**63.0%**	55.2%
Event Based F1	Validation Dataset	29.7%	-	**31.0%**	-	26.9	-	27.7	22.3%	25.8%
Segment Based F1		55.6%	-	58.2%	-	48.7%	-	50.5%	**59.4%**	53.7%

the validation dataset is made up of audio clips extracted from YouTube and Vimeo videos. The best performing was the model trained using C3 which achieved an F1-score of 31%. By comparing the validation results with other participants, the proposed system was ranked 8th among 19 teams (inclusive of the baseline system) in this year Task 4 challenge.

The system proposed by Lin and Wang [22] come in first place which achieved an accuracy of 42.7% while the system proposed by Yang et al. [23] took the last place which achieved an accuracy of 6.7%. On the other hand, the median accuracy for this challenge was at 29.25%.

While this system can be considered as an above-average system, there is still a large room of improvement as compared to the top 3 models which achieved F1-score of above 40%. One of the common features adopted by the top 3 models [22], [24], [25] was the use of mean teacher model which was also part of the winning model in 2018 [8] ([22] used a variant of mean teacher model called the professional teacher model). The idea of the mean teacher model was to average the model weights over training steps instead of label predictions and at the same time bringing the benefits of improved accuracy with fewer training labels [26] and this has become the new frontier in SED as seen in DCASE 2018 and 2019 challenge. However, it should be noted that virtual adversarial training as proposed by Agnone and Altaf [27] can be promising as well where it achieved an accuracy of 59.57% on the evaluation dataset although it only achieved an accuracy of 25% on the validation dataset. It was mentioned in [26] that both methods are compatible and their combination may produce better outcomes.

5. CONCLUSION

In this paper, a five layers CNN with the use of NMF was proposed for DCASE 2019 task 4. The proposed system was able to achieve a higher event based F1-score as compared to the baseline model. However, there is still room for improvement, particularly in the aspect of source separation that may very well helps in the accuracy of sound event detection. Future work may also consider the integration of mean teacher model virtual adversarial training which may produce an even better outcome.

6. ACKNOWLEDGMENT

We would like to thank the organizers for their technical support especially Nicolas Turpault, Romain Serizel and also Kong Qiuqiang from Surrey University for his prompt replies in regards to our questions on his system.

7. REFERENCES

[1] E. Cakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, “Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection,” *IEEE/ACM Trans Audio, Speech, an Language Process.*, vol. 25, no. 6, pp. 1291-1303, Jun. 2017.

[2] T. Hayashi, S. Watanabe, T. Toda, T. Hori, J. L. Roux, and K. Takeda, “Duration-Controlled LSTM for Polyphonic Sound Event Detection,” *IEEE/ACM Trans Audio, Speech, an Language Process.*, vol. 25, no. 11, pp. 2059-2070, Nov. 2017.

[3] Q. Kong, Y. Xu, I. Sobieraj, W. Wang, and M. D. Plumbley “Sound Event Detection and Time–Frequency Segmentation from Weakly Labelled Data,” *IEEE/ACM Trans Audio, Speech, an Language Process.*, vol. 27, no. 4, pp. 777-787, Apr. 2019.

[4] B. McFee, J. Salamon, and J. P. Bello, “Adaptive Pooling Operators for Weakly Labeled Sound Event Detection,” *IEEE/ACM Trans Audio, Speech, an Language Process.*, vol. 26, no. 11, pp. 2180-2193, Apr. 2018.

[5] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, “A Joint Separation-Classification Model For Sound Event Detection of Weakly Labelled Data,” in *2018 IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP)*, Calgary, AB, Canada, Apr. 2018, pp. 321-325.

[6] S. Adavanne, G. Parascandolo, P. Pertila, T. Heittola, and T. Virtanen, “Sound Event Detection In Multichannel Audio Using Spatial and Harmonic Features,” in *Detection and Classification of Acoustics Scenes and Events 2016 Workshop*, Budapest, Hungary, Sept. 2016, pp. 1-5.

[7] S. Adavanne, P. Pertila, and T. Virtanen, “Sound Event Detection Using Spatial Features and Convolutional

Recurrent Neural Network," in *Detection and Classification of Acoustics Scenes and Events 2017 Workshop*, Munich, Germany, Nov. 2017, pp. 1-5.

[8] J. Lu, "Mean Teacher Convolution System For DCASE 2018 Task 4," *Detection and Classification of Acoustics Scenes and Events 2018 Challenge*, Jul. 2018. [Online]. Available: http://dcase.community/documents/challenge2018/technical_reports/DCASE2018_Lu_19.pdf

[9] D. Su, X. Wu, L. Xu, "GMM-HMM acoustic model training by a two level procedure with Gaussian components determined by automatic model selection," in *2010 IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP)*, Dallas, TX, USA, Mar. 2010, pp. 4890-4893.

[10] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen "Acoustic Event Detection In Real Life," in *18th European Signal Process. Conf.*, Aalborg, Denmark, Aug. 2010, pp. 1267-1271.

[11] O. Dikmen, and A. Mesaros, "Sound Event Detection Using Non-Negative Dictionaries Learned From Annotated Overlapping Events," in *2013 IEEE Workshop Applications Signal Process. Audio Acoustics*, New Paltz, New York, Oct. 2013, pp. 1-4.

[12] V. Bisot, S. Essid, and G. Richard, "Overlapping Sound Event Detection With Supervised Nonnegative Matrix Factorization," in *2017 IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP)*, New Orleans, LA, USA, Mar. 2017, pp. 31-35.

[13] T. Komatsu, Y. Senda, and R. Kondo, "Acoustics Event Detection Based on Non-Negative Matrix Factorization With Mixtures of Local Dictionaries and Activation Aggregation," in *2016 IEEE Int. Conf. Acoustics, Speech and Signal Process. (ICASSP)*, Shanghai, China, Mar. 2016, pp. 2259-2263.

[14] Z. Md. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-Art Deep Learning: Evolving Machine Intelligence Toward Tomorrow's Intelligent Network Traffic Control Systems," *IEEE Commun. Surveys Tutorials*, vol. 19, no. 4, pp. 2432-2455, 2017.

[15] Z. Liu, Z. Jia, C. Vong, S. Bu, J. Han, and X. Tang, "Capturing High-Discriminative Fault Features for Electronics-Rich Analog System via Deep Learning," *IEEE Trans. Indust. Inform.*, vol. 13, no. 3, pp. 1213-1226, Jun. 2017.

[16] M. He and D. He, "Deep Learning Based Approach for Bearing Fault Diagnosis," *IEEE Trans. Indust. Applications*, vol. 53, no. 3, pp. 3057-3065, Jun. 2017.

[17] T. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "PCANet: A Simple Deep Learning Baseline for Image Classification?," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5017-5032, Dec. 2015.

[18] A. Mesaros, T. Heittola , E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and Classification of Acoustic Scenes and Events: Outcome of the DCASE 2016 Challenge," *IEEE/ACM Trans Audio, Speech, an Language Process.*, vol. 26, no. 2, pp. 379-393, Feb. 2018.

[19] Q. Kong, Y. Cao, T. Iqbal, Yong Xu, W. Wang, and M. D. Plumbley, "Cross-task learning for audio-tagging, sound event detection spatial localization: DCASE 2019 baseline systems," *arXiv preprint arXiv: 1904.03476*, pp. 1-5.

[20] D. D. Lee, and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788-791, Oct. 1999.

[21] N. Turpault, R. Serizel, A. P. Shah, and J. Salamon, "*Sound event detection in domestic environments with weakly labeled data and soundscape synthesis,"* working paper or preprint, Jun. 2019. [Online]. Available: https://hal.inria.fr/hal-02160855.

[22] L. Lin and X. Wang, "Guided Learning Convolution System for DCASE 2019 Task 4," *Detection and Classification of Acoustics Scenes and Events 2019 Challenge*, Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Lin_25.pdf

[23] Q. Yang, J. Xia, and J. Wang, "Mean Teacher Model Based On CMRANN Network For Sound Event Detection," *Detection and Classification of Acoustics Scenes and Events 2019 Challenge*, Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Yang_18.pdf

[24] L. Delphin-Poulat and C. Plapous "Mean Teacher With Data Augmentation For DCASE 2019 Task 4," *Detection and Classification of Acoustics Scenes and Events 2019 Challenge*, Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Delphin_15.pdf

[25] Z. Shi, "Hodgepodge: Sound Event Detection Based On Ensemble of Semi-Supervised Learning Methods," *Detection and Classification of Acoustics Scenes and Events 2019 Challenge*, Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Shi_11.pdf

[26] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *31st Conf. Neural Information Processing Syst. (NIPS 2017)*, Long Beach, CA, USA, Dec. 2017, pp. 1-10.

[27] A. Agnone and U. Altaf, "Virtual Adversarial Training System For DCASE 2019 Task 4," *Detection and Classification of Acoustics Scenes and Events 2019 Challenge*, Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Agnone_78.pdf

ACOUSTIC SCENE CLASSIFICATION BASED ON A LARGE-MARGIN FACTORIZED CNN

Janghoon Cho, Sungrack Yun, Hyoungwoo Park, Jungyun Eum, Kyuwoong Hwang

Qualcomm AI Research*, Qualcomm Korea YH
343, Hakdong-ro, Gangnam-gu, Seoul, Korea
{janghoon, sungrack, c_hyoupa, c_jeum, kyuwoong}@qti.qualcomm.com

ABSTRACT

In this paper, we present an acoustic scene classification framework based on a large-margin factorized convolutional neural network (CNN). We adopt the factorized CNN to learn the patterns in the time-frequency domain by factorizing the 2D kernel into two separate 1D kernels. The factorized kernel leads to learn the main component of two patterns: the long-term ambient and short-term event sounds which are the key patterns of the audio scene classification. In training our model, we consider the loss function based on the triplet sampling such that the same audio scene samples from different environments are minimized, and simultaneously the different audio scene samples are maximized. With this loss function, the samples from the same audio scene are clustered independently of the environment, and thus we can get the classifier with better generalization ability in an unseen environment. We evaluated our audio scene classification framework using the dataset of the DCASE challenge 2019 task1A. Experimental results show that the proposed algorithm improves the performance of the baseline network and reduces the number of parameters to one third. Furthermore, the performance gain is higher on unseen data, and it shows that the proposed algorithm has better generalization ability.

Index Terms— Acoustic scene classification, Factorized convolutional neural network, Triplet sampling

1. INTRODUCTION

The interest of acoustic scene classification (ASC) has been continuously increasing in the last few years and is becoming an important research in the fields of acoustic signal processing. The ASC aims to identify different environments given the sounds they produce [1] and has various applications in context-awareness and surveillance [2, 3, 4]: e.g. the device which recognizes the environmental sound by analyzing the surrounded audio information. With the release of large scale datasets and challenge tasks by Detection and Classification of Acoustic Scenes and Events (DCASE) [5, 6], the ASC has become a very popular research topic in audio signal processing. We have an increasing number of research centers, companies, and universities participating in the DCASE challenge and workshop every year.

In the past decade, deep learning has accomplished many achievements in audio, image, and natural language processing. Especially, the algorithms based on the convolutional neural network (CNN) are dominant in ASC [7, 1, 8, 9] tasks. In [2], a simple CNN with 2 layers was adopted, and many attempts were tried to solve the overfitting problem in improving the ASC performance by increasing the model complexity: e.g. number of layers in CNN. In [7], SubSpectralNet method was introduced to capture more enhanced features with a convolutional layer by splitting the time-frequency features into sub-spectrograms. In [8], a simple preprocessing method was adopted to emphasize the different aspects of the acoustic scene. In [10, 11, 12], an ensemble of various acoustic features such as MFCC, HPSS, i-vector and the technique that independently learns the classifiers of each feature was proposed. However, the method is heuristic and requires a lot of computation in the front-end step before CNN inference step.

In order to solve the problems of the above-mentioned conventional methods, we propose an algorithm that uses only one feature and does not increase the model complexity of CNN. The pattern of each acoustic scene in the time-frequency domain can be represented as a low-rank matrix, thus we consider designing a 2D convolution layer as two consecutive convolution layers with 1D kernels. Also, we consider a loss function such that the samples from the same audio scene are clustered independently of the environment to be robust to unseen environment. In short, our proposed ASC framework is based on the Sub-Spectral Net [7], and the kernel factorization and loss function are applied to reduce the computational complexity and increase the generalization ability on unseen environment. We evaluated our ASC framework using the dataset of DCASE task1A, and all experimental results show that the proposed algorithm with the data augmentation techniques [13, 14] significantly improves the accuracy.

The rest of the paper is organized as follows. Section 2 formulates the problem of ASC. Section 3 describes the proposed algorithm including our factorized CNN structure and novel loss function. Section 4 presents the experimental results and analysis. Finally, Section 5 concludes.

2. ACOUSTIC SCENE CLASSIFICATION

Audio scene classification, the task1A of the DCASE 2019 challenge [6], is a process of predicting a label y^* given an input audio clip $\mathbf{x}$ as:

$$y^* = \arg\max_{y\in\mathcal{Y}} p(y|\mathbf{f}_\mathbf{x};\theta) \tag{1}$$

where $p(y|\mathbf{f}_\mathbf{x};\theta)$ is the audio scene posterior given the feature map $\mathbf{f}_\mathbf{x}$ with the network parameter θ, and $\mathcal{Y}$ is the entire set of scene labels. The input audio clip $\mathbf{x}$ contains only one audio scene, and the feature map of $\mathbf{x}$, $\mathbf{f}_\mathbf{x}$, can be obtained using various algorithms such as deep audio embeddings [15, 16], log-mel [17, 18, 10, 12], amplitude modulation filter bank [19, 20], and perceptual weighted power spectrogram [11]. In this paper, we use the 40 log-mel, $\mathbf{f}_\mathbf{x} \in \mathrm{R}^a$, since recently many approaches [10, 12] adopt the log-mel feature and show good performances in audio scene classification task. The

*Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

https://doi.org/10.33682/8xh4-jm46

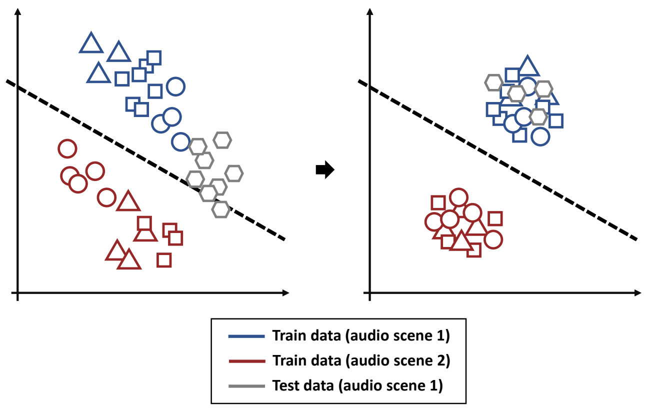

Train data (audio scene 1)
Train data (audio scene 2)
Test data (audio scene 1)

Figure 1: Examples of two audio scene samples (red, blue) from different cities (Triangle, rectangle, circle, hexagon). Given an audio scene, the audio scene from the same city are more clustered than from different cities. For the better generalization ability on unseen city, we apply a loss such that the samples are clustered independently of the cities.

posterior $p(y|\mathbf{f_x};\theta)$ is the probabilistic score of the audio scene label y using the softmax:

$$p(y|\mathbf{f_x};\theta) = \frac{\exp(M_y(\mathbf{f_x}))}{\sum_{v\in\mathcal{Y}}\exp(M_v(\mathbf{f_x}))} \tag{2}$$

where $M_{v\in\mathcal{Y}}(\mathbf{f_x})$ is the output of $\mathbf{f_x}$ obtained from the final layer of the audio scene classification network M.

The dataset of task1A consists of the audio scene samples recorded in a number of cities: i.e. samples of an audio scene (e.g. airport) were recorded in a number of locations (e.g. Amsterdam, London, Helsinki). In real applications, the audio scene classifier can be trained to classify N audio scenes using the dataset recorded in a limited number of cities, and the classifier may be deployed to the test environments which are unseen cities in the training dataset. In such cases, the test samples can be misclassified since the classification boundary may not accurately separate the audio scene samples from unseen cities. This is illustrated in Fig. 1. In this example, there are two audio scene feature points (red, blue) from three different cities (triangle, circle, rectangle). And, given an audio scene, it's highly probable that the samples from the same city are more clustered than from different cities. The black dashed line is the classification boundary trained with the audio scene data from three different cities. Here, we may have the test samples from unseen city (gray hexagon) which can be presented in the other region near the classification boundary. In this case, some of the test samples will be misclassified. In contrast to the left-side of the Fig. 1, if the audio scene features are clustered independently of the cities, and the within-class distances are minimized, then we can have the classifier with more generalization ability especially when there are audio scene samples from unseen cities as shown in the right-side of the Fig. 1.

3. PROPOSED ALGORITHM

The block diagram of our proposed algorithm is illustrated in Fig. 2. The overall structure is based on Sub-Spectral Net [7] which assumes that the key patterns of each class are concentrated on the specific frequency bins, and those key patterns are effectively learned by dividing the input mel-spectrogram into the multiple sub-bands and using the CNN classifier for each sub-band independently. By dividing the input mel-spectrogram into the multiple sub-bands and learning the CNN classifier of each sub-band independently, the key patterns of specific frequency bins are effectively learned. The CNN classifiers consist of two convolution layers and one fully connected network as in the baseline network of DCASE 2019. The outputs of all FCN layers are concatenated, and they are used as the input of the global classifier.

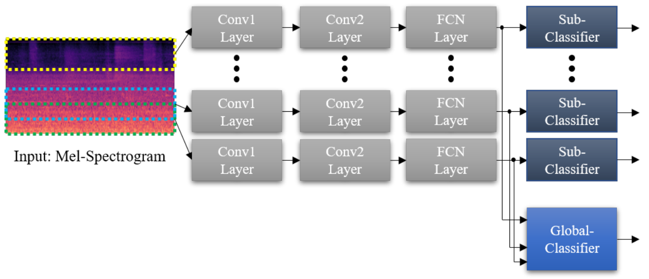


Figure 2: Block diagram of proposed acoustic scene classification algorithm.

3.1. Factorized CNN

Recently, the CNN-based algorithms have shown great performance and the state-of-the-art result in various areas such as image classification, audio/speech processing, speech recognition, and speaker verification [21, 22]. As CNN-based algorithms are also a popular trend in the ASC task, most of the algorithms submitted to the DCASE 2018 are based on CNN [10, 11, 12]. The mel-spectrogram feature of audio data can be regarded as an image, and the CNN-based algorithm can be used to recognize the audio characteristics such as the phoneme and human voices.

When we train a CNN-based model for acoustic scene classification, the over-fitting problem can be occurred due to the limitation of the data even when we use simple 2-layer CNN structure is used, while learned convolution filters had a noisy pattern that was difficult to analyze. The CNN model tends to memorize all training audio scenes including the noise components which do not help classification, and it may cause the poor generalization performance.

To resolve the above-mentioned problem, we propose the factorized CNN based on low-rank matrix factorization. Low-rank matrix factorization is also widely used technique in audio signal separation [23, 24], and it is based on that the mel-spectrogram of audio signal can be represented as the summation of a small number of rank 1 matrices. This leads to classify the acoustic scene with a small number of parameters for the ASC network.

In the classification of acoustic scenes, we can consider the following two audio elements: one is the ambient signal over a long period of time, and the other is an event signal with short period such as bird sound in a park and car horn in the road. As shown in Fig. 3, the ambient signals for each acoustic scene show faint stripes of horizontal lines in mel-spectrogram due to the statistical stationarity over time. Also, many audio examples for this kind of event signals show the pattern with the horizontal stripe, and this can be represented as a rank-1 matrix.

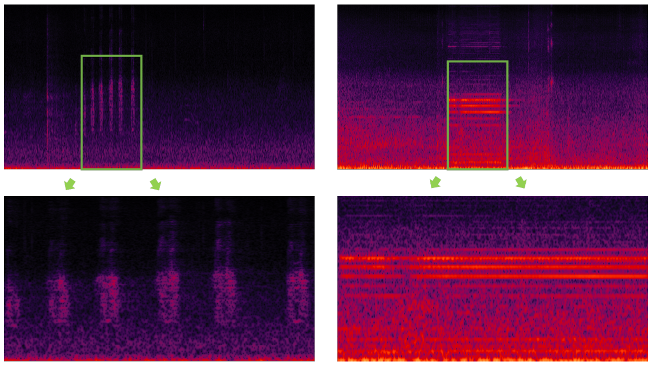


Figure 3: The mel-spectrograms of the sound of a bird in a park and the horn of a car. (Top left) Mel-spectrogram of ”park-stockholm-102-2895-a.wav” in the development dataset (Bottom left) Zoom-up of top left (Top right) Mel-spectrogram of ”street_traffic-london-271-8255-a.wav” (Bottom right) Zoom-up of top right

The detail specification of each layer is described in Figure 4. Rather than using a single conv2D layer which has rectangular (k, k) kernel, we use two consecutive conv2D layers. These two conv2D layers have 1 dimensional kernels $(k, 1)$ and $(1, k)$. When the rectangular (k, k) kernel is a rank-1 matrix and can be factorized into $(k, 1)$ and $(1, k)$ vectors, conv2D with (k, k) kernel becomes equivalent to conv2D with $(k, 1)$ and conv2D with $(1, k)$. So, this network is equivalent to conv2D layer with a rank-1 matrix kernel. With the convolution kernel of rank-1 matrix, we can reduce the over-fitting of learning noisy patterns from training data. Also, the number of model parameters can be reduced since two 1-dim kernels need $2k$ while the square kernel needs k^2 parameters.the square kernel needs k^2 parameters.

The factorized CNN originally proposed in this paper is different from the network in [25] where it factorized the 3D convolution layer as a single intra-channel convolution and a linear channel projection.

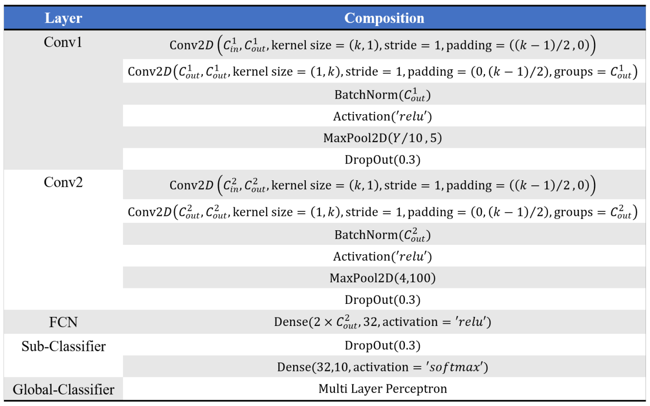
3.2. Large-margin loss function

As the acoustic scene classification is the multi-class classification problem, which maximizes the probability of the correct label and minimizes that of all the others, most of the algorithms are using the cross-entropy loss function. The cross entropy between the true label y and recognized output $\hat{y}$ is given as

$$L_{\text{CE}} = -\sum_{y} y \log(\hat{y}). \tag{3}$$

However, the cross-entropy loss only focuses on fitting or classifying the training data accurately; it does not explicitly encourage a large decision margin for classification [26]. Even when training the simple CNN classifier which has only 2 layers with the cross entropy loss, we can observe that the training loss converges to zero, while the test loss does not converge.

In this context, we consider a loss function similar to the triplet loss but slightly different. Triplet loss function is widely used loss function in many machine learning tasks such as person re-identification, face clustering, and speaker embedding [27, 22]. It enforces the positive pairs to be closer, and the negative pairs to be



Layer	Composition
Conv1	$\text{Conv2}D\left(C_{in}^1, C_{out}^1, \text{kernel size} = (k,1), \text{stride} = 1, \text{padding} = ((k-1)/2\,,0)\right)$
	$\text{Conv2}D(C_{out}^1, C_{out}^1, \text{kernel size} = (1,k), \text{stride} = 1, \text{padding} = (0,(k-1)/2), \text{groups} = C_{out}^1)$
	$\text{BatchNorm}(C_{out}^1)$
	$\text{Activation}('relu')$
	$\text{MaxPool2D}(Y/10\,,5)$
	DropOut(0.3)
Conv2	$\text{Conv2}D\left(C_{in}^2, C_{out}^2, \text{kernel size} = (k,1), \text{stride} = 1, \text{padding} = ((k-1)/2\,,0)\right)$
	$\text{Conv2}D(C_{out}^2, C_{out}^2, \text{kernel size} = (1,k), \text{stride} = 1, \text{padding} = (0,(k-1)/2), \text{groups} = C_{out}^2)$
	$\text{BatchNorm}(C_{out}^2)$
	$\text{Activation}('relu')$
	MaxPool2D(4,100)
	DropOut(0.3)
FCN	$\text{Dense}(2 \times C_{out}^2, 32, \text{activation} = 'relu')$
Sub-Classifier	DropOut(0.3)
	$\text{Dense}(32{,}10, \text{activation} = 'softmax')$
Global-Classifier	Multi Layer Perceptron

Figure 4: The detail specifications of each block.

further and can be expressed as

$$L_{\text{triplet}} = \max\left(\|\mathbf{x}_a - \mathbf{x}_p\|^2 - \|\mathbf{x}_a - \mathbf{x}_n\|^2 + \alpha, 0\right), \tag{4}$$

where $\mathbf{x}$ indicates the embedding vector which is the output of the final convolution layer before entering the fully connected layer, and $\mathbf{x}_a$, $\mathbf{x}_p$, and $\mathbf{x}_n$ are anchor, positive, and negative embedding vectors, respectively. α indicates the triplet loss margin parameter. The anchor and positive pair should come from the same class, and the anchor and negative pair should come from the different classes. In our case, we modified the sampling: we choose the positive sample from the same class but different environment (city) to cluster the samples independently of the environment as shown in Fig. 1. Finally, we combine the cross-entropy and triplet losses as

$$L_{\text{ASC}} = L_{\text{CE}} + \gamma L_{\text{triplet}} \tag{5}$$

where γ is the hyper-parameter which should be tuned.

To reduce the triplet loss effectively, we should choose a distant anchor-positive pair and a close anchor-negative pair, however, it is very inefficient to figure out the distance of all pairs for choosing efficient pairs. Fortunately, the additional label which describes the city of each acoustic scene is given in DCASE 2019 dataset. The sound signals of the same acoustic scene and city is more similar than that of the same acoustic scene and different city, therefore we choose all anchor-positive pairs from the same acoustic scene and different city and all anchor-negative pairs randomly.

3.3. Data augmentation

Since the number of training data is limited in the development dataset, it is necessary to perform data augmentation to increase the performance of unknown data. Most of the algorithms participating in the DCASE challenge are using a deep neural network-based algorithm with high model complexity, so they are using data augmentation and it improves the performance.

We used mix-up [13] and spec-augment [14] for the data augmentation. Mix-up is one of the most popular method in past DCASE 2018 challenge. It creates a new training sample by mixing a pair of two randomly chosen training samples. Spec-augment is an effective approach which shows significant performance improvement in acoustic speech recognition recently. It replaces values by zeros in randomly chosen time-frequency bands. It is also effective in acoustic scene classification task, and applied in most of the algorithms submitted in DCASE 2019 challenge.

Validation dataset	Overall		Unseen city	
Feature (Logmel number)	40	200	40	200
SubSpectralNet [7]	68.93	73.44	58.29	68.71
FCNN	71.15	75.97	59.89	70.32
FCNN-mixup	71.57	76.25	62.19	68.70
FCNN-spec	71.85	76.44	63.9	71.74
FCNN-mixup-spec	72.76	75.97	62.62	70.40
FCNN-triplet	72.67	76.61	63.07	70.24
FCNN-triplet-spec	**73.14**	**77.19**	**64.23**	**72.38**
DCASE 2019 Rank 1 [28]	85.10		-	

Table 1: Classification accuracies (%) of the proposed algorithm with different settings and the baseline CNN algorithm.

4. EXPERIMENT

4.1. Dataset

The dataset for this task is the TAU Urban Acoustic Scenes 2019 dataset, consisting of recordings from various acoustic scenes in ten large European cities. For each recording location, there are 5-6 minutes of audio. The original recordings were split into segments with a length of 10 seconds that are provided in individual files. The dataset includes 10 scenes such as ’airport’ and ’shopping mall’. TAU Urban Acoustic Scenes 2019 development dataset contains 40 hours of data with total of 14400 segments. Here, we used 9185 segments as a training dataset and 4185 segments as an evaluation dataset, and this split is given in the first fold of the validation set. For evaluation of unseen city, we used 1440 segments which is recorded in Milan and not appeared in the training dataset.

4.2. Setup

Our source code is implemented as python script using Torch library and our experiment is conducted on a GeForce GTX TITAN X GPU having 12Gb RAM. As the stereo audio segments whose length is 10 seconds are sampled as 48kHz in DCASE 2019 development dataset, we used 40 and 200 logmel features of the stereo channel without down-sampling as the input of CNN. Also, we set the sub-spectrogram size as 20 and overlap as 10 as the same setting in [7]. The input/output channel sizes of CNN structure is assumed to be $C^1_{in} = 2, C^1_{out} = 64, C^2_{in} = 64, C^2_{out} = 64$ for 40 logmel case and $C^1_{in} = 2, C^1_{out} = 32, C^2_{in} = 32, C^2_{out} = 64$ for 200 logmel case. The kernel size parameter is set to be $k = 7$ for all logmel cases. All of the convolution filters and weight matrices in dense layers are initialized by kaiming normal and xavier normal functions in pyTorch, respectively. The learning rate is set to 0.001 with Adam optimizer. Here, the hyper-parameters of triplet loss margin, and the balance coefficient between the cross-entropy and triplet losses are respectively given as $\alpha = 0.2$, and $\gamma = 10$.

4.3. Result

The DCASE 2019 development dataset includes recordings from ten cities, and the training subset contains only 9 cities excepting for Milan as unseen city. By checking the performance of unseen city, we can measure the generalization ability of the learned model. Therefore, we measured not only the accuracy of overall validation dataset but also the accuracy of the dataset only containing unseen city.

Input feature	Network	# of param
Logmel 40	DCASE baseline	117K
	SubSpectralNet [7]	331K
	Proposed FCNN	113K
Logmel 200	SubSpectralNet [7]	2,541K
	Proposed FCNN	871K
DCASE 2019 Rank 1 [28]		48M

Table 2: The number of parameters for the DCASE baseline, SubSpectralNet, and our proposed FCNN

Experimental results are enumerated in Table 1. Since the classification accuracy of the evaluation set is saturated before the 100-th epoch, we trained all models for 200 epochs and averaged the accuracies of last 10 epochs. For each evaluation of the algorithms, the model training is conducted twice and the accuracies are also averaged.

For the overall dataset, the factorized CNN (FCNN) improves the performance of the baseline network by 2.22% and 2.53% for 40 and 200 logmel cases, respectively. Further improvement is confirmed when using the mix-up and spec-augment data augmentation. By adopting the proposed loss function combined the triplet loss also improves the performance of the FCNN by 1.52% and 0.64%. When the spec-augment is applied to the FCNN with the proposed loss function, the best performance which improves the baseline network by 4.21% and 3.75% is obtained. Here, mix-up cannot be used with triplet loss since the labels of augmented data using mix-up are not discrete. For the only evaluation of unseen city, the performance trends are similar to the overall dataset, and the quantity of the performance enhancement is increased. The total increase in performance of unseen city is 5.94% and 3.67%. For 40 logmel case, the performance increase is relatively 41% more than the increase of the overall dataset, and it supports that the proposed algorithm is robust against unseen data. Note that the performance comparison with DCASE 2019 Challenge Rank 1 system [28] is unfair since the system characteristics such as the number of input features, model size, usage of ensemble and so on are different from ours, but we enumerated it as a reference.

As the number of parameters in CNN is one of the main criteria, we briefly compared the number of model parameters. As enumerated in Table 2, the number of proposed FCNN parameters is about one-third of SubSpectralNet, and it is almost the same as DCASE baseline network.

5. CONCLUSION

In this paper, an acoustic scene classification algorithm based on a large-margin factorized CNN is proposed. The motivation of a factorized CNN is that the most key patterns in the mel-spectrogram including the long-term ambient and short-term event sounds are low-rank, and the factorized CNN can effectively learn these key patterns with reducing the number of model parameters. To increase the generalization performance of the learned model, the triplet loss is combined with the cross-entropy loss function. Experimental results show that our proposed algorithm outperforms a conventional simple CNN-based algorithm with decreasing the model complexity. Further improvement on unseen data is also shown and it supports that the proposed algorithm has better generalization performance.

6. REFERENCES

[1] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 16–34, 2015.

[2] M. Valenti, A. Diment, G. Parascandolo, S. Squartini, and T. Virtanen, "DCASE 2016 acoustic scene classification using convolutional neural networks," in *Proc. Workshop Detection Classif. Acoust. Scenes Events*, 2016, pp. 95–99.

[3] R. Radhakrishnan, A. Divakaran, and A. Smaragdis, "Audio analysis for surveillance applications," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2005.* IEEE, 2005, pp. 158–161.

[4] S. Chu, S. Narayanan, and C.-C. J. Kuo, "Environmental sound recognition with time–frequency audio features," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 17, no. 6, pp. 1142–1158, 2009.

[5] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," *arXiv preprint arXiv:1807.09840*, 2018.

[6] "DCASE2019 task1a description," http://dcase.community/challenge2019/task-acoustic-scene-classification#subtask-a.

[7] S. S. R. Phaye, E. Benetos, and Y. Wang, "Subspectralnet–using sub-spectrogram based convolutional neural networks for acoustic scene classification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2019, pp. 825–829.

[8] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *the Detection and Classification of Acoustic Scenes and Events (DCASE)*, pp. 1–5, 2017.

[9] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[10] Y. Sakashita and M. Aono, "Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divisions," *Detection and Classification of Acoustic Scenes and Events (DCASE) Challenge*, 2018.

[11] M. Dorfer, B. Lehner, H. Eghbal-zadeh, H. Christop, P. Fabian, and W. Gerhard, "Acoustic scene classification with fully convolutional neural networks and i-vectors," *Tech. Rep., DCASE2018 Challenge*, 2018.

[12] H. Zeinali, L. Burget, and J. Cernocky, "Convolutional neural networks and x-vector embedding for DCASE2018 acoustic scene classification challenge," *arXiv preprint arXiv:1810.04273*, 2018.

[13] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[14] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[15] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2019, pp. 3852–3856.

[16] V. Arora, M. Sun, and C. Wang, "Deep embeddings for rare audio event detection with imbalanced data," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2019, pp. 3297–3301.

[17] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time–frequency weighted noisy speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2125–2136, 2011.

[18] Y. Hoshen, R. J. Weiss, and K. W. Wilson, "Speech acoustic modeling from raw multichannel waveforms," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2015, pp. 4624–4628.

[19] T. Dau, B. Kollmeier, and A. Kohlrausch, "Modeling auditory processing of amplitude modulation. i. detection and masking with narrow-band carriers," *The Journal of the Acoustical Society of America*, vol. 102, no. 5, pp. 2892–2905, 1997.

[20] N. Moritz, J. Anemüller, and B. Kollmeier, "An auditory inspired amplitude modulation filter bank for robust feature extraction in automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 23, no. 11, pp. 1926–1937, 2015.

[21] A. Torfi, J. Dawson, and N. M. Nasrabadi, "Text-independent speaker verification using 3d convolutional neural networks," in *2018 IEEE International Conference on Multimedia and Expo (ICME).* IEEE, 2018, pp. 1–6.

[22] L. Wan, Q. Wang, A. Papir, and I. L. Moreno, "Generalized end-to-end loss for speaker verification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2018, pp. 4879–4883.

[23] P. Sprechmann, A. M. Bronstein, and G. Sapiro, "Real-time online singing voice separation from monaural recordings using robust low-rank modeling." in *ISMIR*. Citeseer, 2012, pp. 67–72.

[24] A. Ozerov and C. Févotte, "Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 550–563, 2009.

[25] M. Wang, B. Liu, and H. Foroosh, "Factorized convolutional neural networks," in *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[26] X. Li, D. Chang, T. Tian, and J. Cao, "Large-margin regularized softmax cross-entropy loss," *IEEE Access*, vol. 7, pp. 19 572–19 578, 2019.

[27] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[28] H. Chen, Z. Liu, Z. Liu, P. Zhang, and Y. Yan, "Integrating the data augmentation scheme with various classifiers for acoustic scene modeling," DCASE2019 Challenge, Tech. Rep., June 2019.

HIERARCHICAL DETECTION OF SOUND EVENTS AND THEIR LOCALIZATION USING CONVOLUTIONAL NEURAL NETWORKS WITH ADAPTIVE THRESHOLDS

Sotirios Panagiotis Chytas, Gerasimos Potamianos

Department of Electrical and Computer Engineering, University of Thessaly, Volos, Greece

`schytas@uth.gr` `gpotam@ieee.org`

ABSTRACT

This paper details our approach to Task 3 of the DCASE'19 Challenge, namely sound event localization and detection (SELD). Our system is based on multi-channel convolutional neural networks (CNNs), combined with data augmentation and ensembling. Specifically, it follows a hierarchical approach that first determines adaptive thresholds for the multi-label sound event detection (SED) problem, based on a CNN operating on spectrograms over long-duration windows. It then exploits the derived thresholds in an ensemble of CNNs operating on raw waveforms over shorter-duration sliding windows to provide event segmentation and labeling. Finally, it employs event localization CNNs to yield direction-of-arrival (DOA) source estimates of the detected sound events. The system is developed and evaluated on the microphone-array set of Task 3. Compared to the baseline of the Challenge organizers, on the development set it achieves relative improvements of 12% in SED error, 2% in F-score, 36% in DOA error, and 3% in the combined SELD metric, but trails significantly in frame-recall, whereas on the evaluation set it achieves relative improvements of 3% in SED, 51% in DOA, and 4% in SELD errors. Overall though, the system lags significantly behind the best Task 3 submission, achieving a combined SELD error of 0.2033 against 0.044 of the latter.

Index Terms— Sound event detection and localization, convolutional neural networks, DCASE19

1. INTRODUCTION

Sound event detection (SED) constitutes an active research area with many applications, such as medical telemonitoring [1] and surveillance [2]. Not surprisingly, SED has been the subject of multiple evaluation campaigns in the literature, including the recent and well-established DCASE Challenges [3–5]. Moreover, alongside SED, in many applications [6, 7] it is also crucial to determine the location or, more coarsely, the direction of arrival (DOA) of each detected sound event source. Thus, in Task 3 of the 2019 DCASE Challenge [8], both problems are considered jointly (SED and DOA estimation of the detected events). The task is referred to as sound event localization and detection (SELD), and it is addressed in an indoors scenario given multi-channel audio.

In this paper, we present our developed SELD system for Task 3 of the 2019 DCASE Challenge [8]. As deep-learning based methods are well-established, outperforming traditional machine learning ones in both SED [9–12] and DOA estimation [13,14], we adopt a deep-learning approach. In particular, we employ convolutional neural networks (CNNs) to first address SED, i.e., determine the existence of each class at each time-frame, and to subsequently estimate the DOA for each of the audio segments predicted to exist. Notably, for SED we follow a hierarchical approach, where, first, a CNN operating over long-duration audio windows determines adaptive thresholds indicating how likely it is for each class to exist, and, subsequently, an ensemble of CNNs operating over shorter-duration windows determines the exact moments each class occurs.

The remainder of the paper is organized as follows: Section 2 overviews the Challenge dataset; Section 3 focuses on the developed SELD system; Section 4 details its evaluation on the Challenge data; and, finally, Section 5 concludes the paper.

2. CHALLENGE DATASET

Task 3 of the 2019 DCASE Challenge provides two datasets of the same indoors sound scene: “Microphone Array” and “Ambisonic” [15]. In this paper, the “Microphone Array” set is employed, containing four-channel directional microphone recordings from a tetrahedral array configuration. The development dataset consists of 400 1-min long recordings sampled at 48 kHz, divided into four cross-validation splits. For the purposes of the Challenge, the given cross-validation split should be used during system development, and the use of external data is not allowed. In addition, the evaluation dataset consists of 100 1-min long recordings. Note also that, in our system, all audio data are downsampled to 16 kHz.

There exist 11 sound event classes, taken from Task 2 of the 2016 DCASE Challenge [4]. The duration of each event segment in the development set ranges from 205ms to 3.335s, and there can be at most two overlapping sounds at any given time. The number of segments is almost the same for all classes, however there exists significant variation in their total durations (see also Fig. 1).

Each segment is associated with an elevation and an azimuth value. Elevation values lie within the [-40°, 40°] range, while azimuth values are within [-180°, 170°], both at a resolution of 10°.

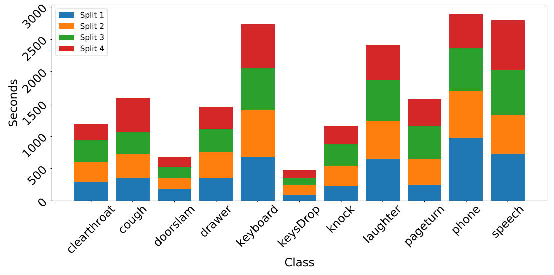


Figure 1: Class total durations in the four development set splits.

3. SYSTEM DESCRIPTION

In our method, we first address the SED sub-task and then the DOA one. Specifically, we develop a hierarchical approach to the for-

https://doi.org/10.33682/c6q0-wv87

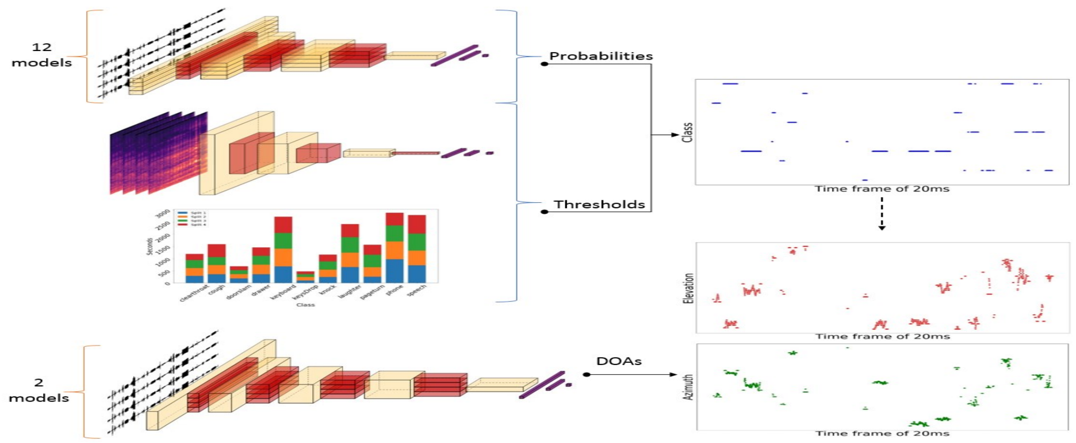


Figure 2: Overview diagram of the developed system for Task 3 of DCASE'19 (CNNs are drawn using the PlotNeuralNet software [16]).

mer, determining the existence of each sound event class at each time-frame. For this purpose, first a "*long SED model*" estimates adaptive thresholds for each class, also taking into account the class prior probabilities. Then, an ensemble of "*short SED models*" determines the exact time-frames each class exists, exploiting the aforementioned thresholds. Following SED, we utilize a *DOA model* to localize the source of each detected event, estimating its elevation and azimuth values. All models are multi-channel CNNs, operating on raw waveforms or spectrograms over sliding windows of different durations. A schematic of the system is provided in Fig. 2.

3.1. Short SED models

We create an ensemble of multi-channel CNNs (12 in total, as explained in the next paragraph), all with the architecture of Table 1. These operate on raw audio waveforms over short-duration windows of 100ms or 200ms, with these values determined after experimenting with various window lengths on the development set. We do not apply any preprocessing to the four channels (other than their downsampling to 16 kHz), and we use all four microphone data streams as input to the CNNs. The output layers of the models have 11 neurons (same as the number of sound event classes), each providing the probability of its corresponding class, following sigmoid activation. Note that, during training, windows with no sound events are kept, and windows with overlapping events are assigned to all occurring events inside them (maximum of two), while they slide in steps equal to half their duration, i.e. by 50 or 100ms. All CNNs are trained with a binary cross-entropy objective using the Adam optimizer and early stopping to prevent overfitting, employing the Keras API for development [17].

In order to have more segments with overlapping sounds, we employ data augmentation as follows: we add segments, each belonging to only one class, two at a time. Concerning the Challenge evaluation metrics, we observed that datasets with more overlapping segments tend to yield better frame-recall results, while data with less overlapping segments tend to perform better in terms of SED error and F-score. As we wish to improve all three metrics simultaneously, we choose to create different models, trained on data with various degrees of artificial overlap, and then ensemble them. Thus, we create six datasets, having 0%, 5%, 10%, 20%, 30%, and 40% extra overlapping segments, and we train two different CNNs on each (i.e. with input window sizes of 100ms and 200ms length), thus resulting to 12 models. The process is repeated for each of the four given development data splits.

3.2. Long SED model

A major issue in multi-label problems concerns the choice of class thresholds, used to decide if a class exists or not. A simple approach is to set all thresholds to 0.5, as in the Challenge baseline system [18], however their careful tuning may yield significant improvements. For example, in [9] exhaustive search is utilized to yield a single optimal threshold for all classes, whereas in [10, 11] separate thresholds are employed for each class, found by exhaustive search. Nevertheless, both approaches may be prone to overfitting due to the exhaustive search used.

To prevent overfitting, we opt to create a SED model operating on longer-duration data windows. Our motivation stems from the expectation that such a model will provide a "bigger picture" concerning class existence, and thus can help in determining class thresholds adaptively. These can then be utilized in conjunction with the outputs of the short SED models to predict the exact time-frames in which each sound event occurs.

For this purpose, we create a multi-channel CNN that operates on power spectrograms over signal windows of one-second duration (sliding in 100ms steps during training), with 128×32-dimensional spectrograms generated by libROSA [19] under its default parameters. We use all available channels, ending up with four spectrograms as input. For data augmentation, we consider all permutations of the four channels, resulting in 24 times more training data. Details of the long SED model architecture are provided in Table 2.

3.3. Adaptive thresholds and SED predictions

To determine the class thresholds, we work with a time-resolution of 20ms, exploiting the long SED model predictions. These fine-resolution predictions are obtained by averaging the coarser-resolution probabilities of each class over all 1s-long windows that contain the given 20ms time-frame, while sliding by 200ms.

A first approach is to simply set the desired thresholds to

$$\theta_c^t = 1 - \mathrm{lp}_c^t \,, \tag{1}$$

where lp_c^t denotes the long SED model prediction (probability) of class c at time-frame t, and θ_c^t is the corresponding threshold. In general, however, we do not wish the thresholds to be too close to 1, in order to guard against false negatives of the long SED model. Thus, we choose to smooth (1) by multiplying the thresholds with a number within the [0.6, 0.9] range. This number is different for each class, and it is based on its total duration in the training data

Input (4 x segment size)
100 filters, Conv 1x10, ReLU MaxPool 1x5
200 filters, Conv 1x10, ReLU MaxPool 1x6
300 filters, Conv 1x10, ReLU MaxPool 1x7
500 filters, Conv 4x1, ReLU
Flatten Dropout 0.6
1000 neurons, Dense, ReLU Dropout 0.3
11 neurons, Dense, sigmoid

Table 1: Architecture of the short SED models. Segment sizes are 1600 for 100ms windows and 3200 for 200ms ones.

Input (4x128x32)
40 filters, Conv 1x6x1, ReLU MaxPool 1x3x1
60 filters, Conv 1x1x6, ReLU MaxPool 1x1x3
80 filters, Conv 1x6x6, ReLU MaxPool 1x3x3
Flatten Dropout 0.5
500 neurons, Dense, ReLU Dropout 0.3
11 neurons, Dense, sigmoid

Table 2: Long SED model architecture.

Input (4 x segment size)
100 filters, Conv 4x10 (same padding), ReLU MaxPool 1x3
200 filters, Conv 4x10 (same padding), ReLU MaxPool 1x5
300 filters, Conv 4x10 (same padding), ReLU MaxPool 1x5
400 filters, Conv 4x10 (same padding), ReLU MaxPool 1x5
500 filters, Conv 4x1 (same padding), ReLU
Flatten Dropout 0.5
1000 neurons, Dense, ReLU Dropout 0.3
22 neurons, Dense, linear

Table 3: Architecture of the DOA models.

(class prior), meaning that less frequent classes tend to have lower thresholds. The resulting thresholds are given by

$$\theta_c^t = \left(1 - \mathrm{lp}_c^t\right)\left(0.6 + 0.3\,\frac{p_c - p_{\min}}{p_{\max} - p_{\min}}\right), \tag{2}$$

where p_c denotes the prior of class c (based on duration), while $p_{\min}$ and $p_{\max}$ are the minimum and maximum of all class priors.

The desired SED results are finally derived at a time-resolution of 20ms, by employing the ensemble of the 12 short SED models of Section 3.2 and the adaptive thresholds of (2). Specifically, let sp_c^t denote the combined short model prediction of class c at time-frame t. This is estimated for each of the 12 CNNs by averaging the class probabilities over all windows (of length 100 or 200ms, depending on the model) that contain the given 20ms time-frame, while sliding by a 20ms step. The resulting estimates are then averaged over all 12 models of the CNN ensemble to yield sp_c^t. As a final step, class c is detected at time-frame t, whenever $\mathrm{sp}_c^t \geq \theta_c^t$.

3.4. DOA models

Following SED, we proceed to the DOA sub-task. For this purpose, and similarly to the short SED models, we create short models for DOA estimation that provide 22 numbers at their output layer, i.e. the elevation and azimuth for each of the 11 classes. The goal is, given a raw multi-channel audio segment of short duration, to predict the DOA of each class, no matter if it exists or not (SED results will determine what to keep). Specifically, we create two CNNs, with their architecture detailed in Table 3. The CNNs operate on four channels of raw audio over windows of 100ms or 200ms in duration that, during model training, slide at steps of 50ms or 100ms, respectively. For training the two networks, we use the same data as in the SED sub-task, but exclude audio with no sound events, as such data are not associated with DOA values. We employ the mean squared error loss as training objective, but slightly modified, as we calculate it only in the 2 (in the case of one class) or the 4 (for two overlapping classes) output neurons of interest. As before, we use the Adam optimizer and early stopping to prevent overfitting.

DOA estimation occurs at a time resolution of 20ms, first by averaging the elevation and azimuth predictions for the 20ms time-frame of interest within the model sliding windows, and subsequently averaging the predictions across the two models. A problem arises in this approach towards the boundaries of each segment. To prevent noisy DOA estimates there, these are smoothed by setting predictions for the first and last 300ms of each segment to the minimum or maximum of that sub-segment (depending on the relative position to the zero), thus preventing steep DOA ascents or descents. An example of this process is depicted in Fig. 3.

3.5. Submitted systems

Following the above process, we create a total of four SELD systems, each trained on one of the four given cross-validation development data splits. We then combine these four systems in two ways, thus providing two submissions to the Challenge, resulting from: (a) their average; and (b) their weighted average. In both cases, averaging occurs at the sub-component level across the four systems (e.g., each short SED CNN is averaged across the four systems first, before model ensembling). Particularly in the weighted averaging

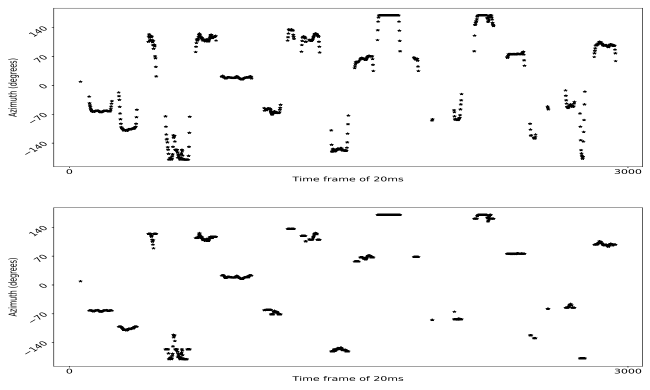


Figure 3: Example of DOA (here, azimuth) estimate smoothing at segment edges: (top) before smoothing; (bottom) after smoothing.

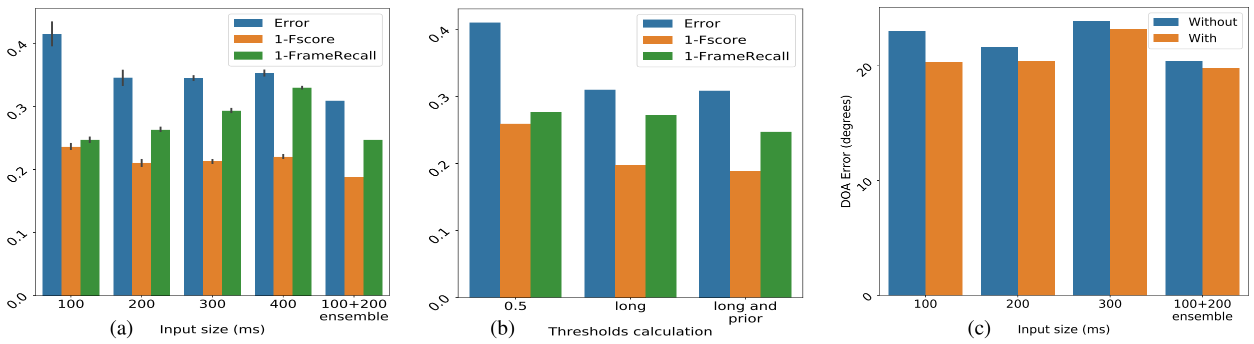


Figure 4: Evaluation of design choices of the proposed system components on the Challenge development set, namely of various: (a) short SED model window durations; (b) class threshold estimation approaches; (c) DOA model window durations with or without smoothing.

case, the performance of the four systems on the corresponding fold test-set is taken into account, based on the appropriate metrics (i.e., the average of the three SED metrics or the DOA error).

4. RESULTS

4.1. Development set results

We first present in Table 4 (top) our system performance on the development set (over its four data splits, thus there is a single result according to the evaluation paradigm), in terms of the four Challenge metrics and their combination (SELD score). We can readily observe that, compared to the Challenge baseline, our system achieves a 3% relative reduction in the SELD score (from 0.22 to 0.213). In terms of the individual metrics, the system yields relative improvements of 12% in SED error, 2% in F-score, 36% in DOA error (from 30.8°to 19.8°), but trails significantly in the frame-recall metric, where it achieves only 75.3% vs. 84.0% for the baseline.

In Fig. 4 we present results to highlight performance differences between the various design choices of our developed system components. First, in Fig. 4(a) we depict performance of the short SED models of Section 3.1 and their ensembles in terms of SED error, F-score, and frame-recall (difference from 1 is shown for the latter two). We also depict results for additional window sizes, namely 300ms and 400ms. Each bar shows results of the ensemble of six models, trained on various data augmented sets (from 0% to 40%, as discussed in Section 3.1), with the error bars indicating the range of the individual model results. Note that the 12-model ensemble results are also shown ("100+200 ensemble"). We observe that shorter window sizes (100ms) yield the best results in frame-recall, mainly because two sounds may overlap for very short periods of time, but have much worse results in SED error and F-score, because short windows may not carry adequate class information. On the other hand, medium window sizes (200ms) yield the best results in SED error and F-score, but worse frame-recall as they may fail to detect very short segments. Combining the two window sizes by model ensembling exploits the relative advantages of both, improving SED error and F-score significantly, but at minor detriment in frame-recall. Longer windows (e.g. 300ms or 400ms sizes) significantly degrade frame-recall, thus are not used in our system. Next, in Fig. 4(b) we examine the effect of class thresholds to SED performance. Thresholds fixed to 0.5 for all classes perform the worst, whereas adaptive thresholds estimated by means of (1) – labeled as "long" in the graph, perform better in all three metrics (SED error, F-score, and frame-recall). Results further improve when adaptive thresholds are computed by (2) – labeled as "long and prior" in the bar-plot. Finally, in Fig. 4(c) we consider the DOA estimation component. There, we can readily observe the importance of DOA estimate smoothing, as systems "without" smoothing perform significantly worse than systems "with" it. Also DOA models operating on windows of 100ms or 200ms in duration outperform systems built on 300ms windows. The ensemble of both 100ms and 200ms systems performs even better in terms of the DOA error metric.

set	system	SED error	F-score	frame-recall	DOA error	SELD score
dev	proposed	**0.309**	**81.2%**	75.3%	**19.8°**	**0.213**
	baseline *	0.350	80.1%	**84.0%**	30.8°	0.220
eval	proposed (a)	**0.29**	82.4%	75.6%	**18.6°**	**0.2033**
	proposed (b)	**0.29**	82.3%	75.7%	18.7°	0.2034
	baseline *	0.30	**83.2%**	**83.4%**	38.1°	0.2114
	best	0.08	94.7%	96.8%	3.7°	0.044

Table 4: Comparison of our system on the development (dev) and evaluation set (eval) of DCASE'19 Task 3 against the Challenge baseline (*: on Microphone Array data), in terms of the five task metrics. Performance of the best-scoring submission is also shown.

4.2. Evaluation set results

Finally, in Table 4 (bottom) we present our system performance on the Challenge evaluation set. Both system variants of Section 4.2 are shown: (a) averaging; and (b) weighted averaging. They perform similarly, with variant (a) being slightly superior. Compared to the baseline, it yields a slight 4% relative reduction in the SELD metric (from 0.2114 to 0.2033), with the greatest improvement in the DOA error metric (51% relative reduction, from 38.1°to 18.6°). It should be noted however that the proposed system lags significantly behind the best overall submission in Task 3 in all metrics.

5. CONCLUSIONS

We presented a SELD system for Task 3 of the DCASE'19 Challenge using CNNs only, separately addressing SED and DOA estimation, while making no explicit assumptions about the maximum possible number of overlapping segments. We followed a hierarchical approach to SED, first determining adaptive class thresholds based on a CNN operating over longer windows, which we then utilized in an ensemble of CNNs operating on shorter waveforms, also exploiting data augmentation in their training. Our system outperformed the baseline, particularly in DOA error, exhibiting consistent performance across development and evaluation sets, but trailed the best Challenge submission considerably.

6. REFERENCES

[1] N. C. Phuong and T. D. Dat, "Sound classification for event detection: Application into medical telemonitoring," in *Proc. International Conference on Computing, Management and Telecommunications (ComManTel)*, 2013, pp. 330–333.

[2] C. Clavel, T. Ehrette, and G. Richard, "Events detection for an audio-based surveillance system," in *Proc. IEEE International Conference on Multimedia and Expo (ICME)*, 2005, pp. 1306–1309.

[3] http://www.cs.tut.fi/sgn/arg/dcase2017/challenge/task-sound-event-detection-in-real-life-audio.

[4] http://www.cs.tut.fi/sgn/arg/dcase2016/task-sound-event-detection-in-synthetic-audio.

[5] http://www.cs.tut.fi/sgn/arg/dcase2016/task-sound-event-detection-in-real-life-audio.

[6] M. Crocco, M. Cristani, A. Trucco, and V. Murino, "Audio surveillance: A systematic review," *ACM Computing Surveys*, vol. 48, no. 4, pp. 52:1–52:46, 2016.

[7] C. J. Grobler, C. P. Kruger, B. J. Silva, and G. P. Hancke, "Sound based localization and identification in industrial environments," in *Proc. Annual Conference of the IEEE Industrial Electronics Society (IECON)*, 2017, pp. 6119–6124.

[8] http://dcase.community/challenge2019/task-sound-event-localization-and-detection.

[9] Y. Chen, Y. Zhang, and Z. Duan, "DCASE2017 sound event detection using convolutional neural network," DCASE2017 Challenge, Tech. Rep., 2017.

[10] I.-Y. Jeong, S. Lee, Y. Han, and K. Lee, "Audio event detection using multiple-input convolutional neural network," DCASE2017 Challenge, Tech. Rep., 2017.

[11] C.-H. Wang, J.-K. You, and Y.-W. Liu, "Sound event detection from real-life audio by training a long short-term memory network with mono and stereo features," DCASE2017 Challenge, Tech. Rep., 2017.

[12] R. Lu and Z. Duan, "Bidirectional GRU for sound event detection," DCASE2017 Challenge, Tech. Rep., 2017.

[13] X. Zhang and D. Wang, "Deep learning based binaural speech separation in reverberant environments," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 5, pp. 1075–1084, 2017.

[14] R. Takeda and K. Komatani, "Sound source localization based on deep neural networks with directional activate function exploiting phase information," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 405–409.

[15] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and detection," in *Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019. [Online]. Available: https://arxiv.org/abs/1905.08546

[16] https://github.com/HarisIqbal88/PlotNeuralNet.

[17] https://keras.io/.

[18] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, 2019.

[19] https://librosa.github.io/librosa/.

GCC-PHAT CROSS-CORRELATION AUDIO FEATURES FOR SIMULTANEOUS SOUND EVENT LOCALIZATION AND DETECTION (SELD) IN MULTIPLE ROOMS

Héctor A. Cordourier Maruri[1], *Paulo López Meyer*[1], *Jonathan Huang*[2],
Juan Antonio del Hoyo Ontiveros[1], *Hong Lu*[2],

[1] Intel Corporation, Intel Labs, Zapopan, Jal., Mexico,
{hector.a.cordourier.maruri, paulo.lopez.meyer, juan.antonio.del.hoyo.ontiveros}@intel.com
[2] Intel Corporation, Intel Labs, Santa Clara, CA, USA,
{jonathan.huang, hong.lu}@intel.com

ABSTRACT

In this work, we show a simultaneous sound event localization and detection (SELD) system, with enhanced acoustic features, in which we propose using the well-known Generalized Cross Correlation (GCC) PATH algorithm, to augment the magnitude and phase regular Fourier spectra features at each frame. GCC-PHAT has already been used for some time to calculate the Time Difference of Arrival (TDoA) in simultaneous audio signals, in moderately reverberant environments, using classic signal processing techniques, and can assist audio source localization in current deep learning machines. The neural net architecture we used is a Convolutional Recurrent Neural Network (CRNN), and is tested using the sound database prepared for the Task 3 of the 2019 DCASE Challenge. In the challenge results, our proposed system was able to achieve 20.8 of direction of arrival error, 85.6% frame recall, 86.5% F-score and 0.22 error rate detection in evaluation samples.

Index Terms— GCC-PHAT, SELD, Polyphonic event detection, Sound source localization, CRNN, Sound event detection.

1. INTRODUCTION

Sound event detection (SED) or Audio Classification, refers to the task of automatically recognizing the type of sound that is being detected into some previously specified classes (like human voice, vehicle moving, music, etc.). Meanwhile, Sound Source Localization, or Sound Direction of Arrival (DoA) detection determines the location of the sound in some coordinate system, and in this task we use elevation and azimuth angles as its proxy. In most current published work, these two tasks have been approached as separate problems. However, there are many applications in which the simultaneous location and identification of the sound can be very useful, like detection of an intended user, observation and understanding of human activities, audio surveillance, autonomous agent navigation, among others[1]. In most of this real-life applications, it is reasonable to assume that sources sometimes will overlap in time. A detection pipeline of this kind of audio events is proposed in [1], and named as polyphonic SED. Such work proposes an interesting CRNN architecture to perform the task, and in this work, we present a system based in that architecture, but with additional features based in the Generalized Cross Correlation (GCC), and provide measurements of the benefits obtained.

2. SOUND EVENT DETECTION

In current literature, Convolutional Neural Networks (CNNs) have been proven to be very effective for image classification tasks. A natural next step was to use CNNs or similar systems for audio classification, providing audio features that resemble images, usually Fourier-based spectrograms, or similar representations. This approach has also been met with relative success[2], and in recent DCASE acoustic scene classification tasks, top submissions are mostly CNN-based or related[3].

2.1. Sound source localization

Estimating the location of a sound source is definitely not a new engineering problem. In classical signal-processing systems, source location is calculated from the Time Difference of Arrival (TDoA) of the signal in each element of a microphone array. Then, an analytic, regression formula, or a machine learning (ML) technique can be used to produce the source location. For the first step, regular cross-correlation can detect the time delay of two signals that contain little auto-correlation (i.e. low reverberation, rich frequency content sounds). In that sense, generalized cross-correlation with phase transform (GCC-PHAT) algorithm, developed in 1976 by Knapp and Carter [4], can reduce the effects of the auto-correlation of a signal, and make the system more robust to reverberation.

However, new machine learning techniques usually do not rely on mapping TDoA to spatial location. Instead, the trend is to directly relate some features of the audio signals to the source location[5, 6, 7]. In our work, we aim to get the best of both worlds: the tractability of a classical signal processing as feature, and the high accuracy and noise robustness of neural nets.

3. AUDIO FEATURE EXTRACTION

3.1. Development data set

For all this work, training and testing was performed using the development data set made available by the 2019 DCASE Challenge for the task 3[8], consisting of 400 recordings of roughly 60 seconds each. In order to identify the acoustic characteristics of the recordings, the average spectrum of all the development audio samples was calculated, and the result can be seen in Fig. 1. It can be noticed that there is practically no audio information above the 15.8 kHz frequency, even when the sample frequency (48 kHz) allows up

https://doi.org/10.33682/3re4-nd65

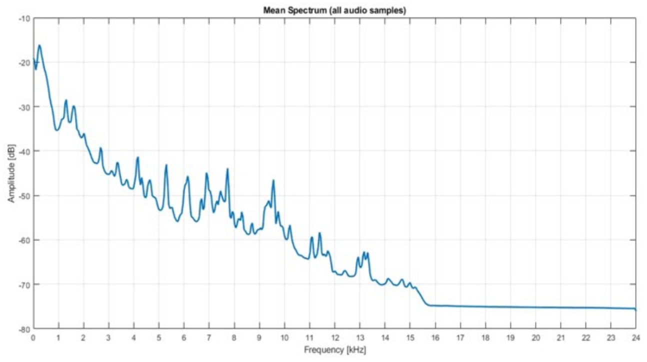


Figure 1: Average magnitude spectrum of all the database audio samples. Notice that there is practically no frequency content above 15.8 kHz.

to 24 kHz. Therefore, in all our feature extraction, we only include data information of frequency bins of up to 15.8 kHz.

The inputs of the feature extraction routine are 4-channel, 48kHz sample frequency, audio recordings. The signals are segmented in time frames, for which a time hop of 20ms is used, producing 3000 samples per each 60-second audio sample. The development data set consists of pre-defined four cross-validation splits of 200 one-minute samples for training, 100 samples for validation, and 100 for testing. The metrics shown in the results section refer to the average of all folds models in their corresponding testing set. The recordings include audio of 11 different classes, that can be located in azimuth angles of -180 to 170 degrees in 10 intervals, and elevations of -40 to 40 in 10 intervals. In preparation for the training routine, all the audio samples were padded or clipped to even out their run time to 60 seconds.

The first feature obtained from each time frame is the GCC-PHAT vector from each signal combination. Usually, the vector obtained from two signals from a microphone array shows a delta-like response close to the vector center, in which the maximum value of the vector has an offset from the center numerically equal to the amount of delay samples between the two time domain signals[4]. Therefore, it is the middle part of the vector which contains most of the useful information. We proposed to segment it, to generate time matrices we call GCC-grams, as an analogous name to spectrograms. An scheme of this process can be seen in Fig. 2.

We propose GCC-grams as an additional input feature, in which all the GCC-grams obtained from all channel combinations (6 in total, for the case of 4 input channels) are concatenated in one single feature matrix in which all the time frames are aligned. Our hypothesis is that this additional pre-processing of audio data can directly improve DoA detection performance. However, in order to keep all input information available for the system, we also feed the magnitude and phase spectra per each channel, properly synchronized. An scheme of this process can be seen in Fig. 3.

For the magnitude and phase spectrograms, the discrete Fourier transform was applied to time frames of 2048 samples, using a Hann window (these numbers were used also for the GCC vector calculation). From the positive frequencies of the output spectra (1024 samples), only the samples corresponding to frequencies below 15.8 kHz were extracted, which produced a vector of $1024 \times (15.8kHz/24kHz) = 672$ samples per each time frame.

In order to keep the same dimensions, the concatenated GCC-gram was fixed to have 672 samples wide too. Therefore, the middle section size of each of the six individual GCC-grams was set up to $672/6 = 112$ samples wide, which was large enough to contain the maximum value of the GCC vector in all recordings. Therefore, per each 1-minute recording, the feature extraction routine produces a 3-D tensor of 3000 time frames, by 672 frequency bins, by 9 components: 1 concatenated GCC-gram, 4 phase spectrograms, and 4 magnitude spectrograms. As a final step, all this 2D matrices were individually normalized.

To gather additional insight in the benefits of our proposed feature extraction technique, additional tests were performed in which only the GGC-grams, or only the magnitude and phase spectrograms, were fed into the proposed CRNN.

4. PROPOSED SYSTEM

4.1. CRNN Architecture

The CRNN architecture we used is based directly in the work proposed by S. Adavanne et al in [1], with some minimal modifications, as our approach was mainly focused on enhancing feature extraction. This architecture is characterized by taking a sequence of features in consecutive frames as input and predicting the sound event classes that are active for each of the input frames along with their respective spatial location (defined as a couple of output angles, azimuth and elevation), producing the temporal activity and DOA related information for each sound event class in parallel. An illustration of the final architecture proposed can be seen in Fig. 4.

The input of the neural network is composed of multiple 2D CNN layers. Each CNN layer has 64 filters of $3 \times 3 \times 9$ receptive fields with a ReLU activation function. After each CNN layer, the outputs are normalized using batch normalization, and the dimensions a reduced with average-pooling (MP_i) along the frequency axis. We preferred average-pooling over max-pooling on the hypothesis that average pooling carries more information from the whole kernel, in this case, the spectrograms and GCC-grams. The output after the final CNN is of dimension $T \times 2 \times 64$, in which T is the number of input time frames.

The output is reshaped to a T frame sequence of 128 feature vectors, fed to two GRU bidirectional layers of 128 nodes, followed by three identical branches of fully connected (FC) layers in parallel, one for SED, one for azimuth and other for elevation detection. The first FC layer consists of 256 input nodes with linear activation, followed by a Dropout layer, a SELU layer, and and finally a linear layer with 12 outputs, one per each audio class, plus an additional "garbage" class for the frames in which there is no audio event present.

4.2. Training parameters

In each frame, one-hot encoding target values were used for each of the active sound events in the event detection branch output. Since sound events can be overlapping in time, it is possible to have multiple ones at each time step. Similarly, for the azimuth and elevation branches, a 12 element vector is produced as output in which the active class (according with the event detection output) contains the numeric value of the angle in degrees, and the rest of the vector contains the garbage value of -181. A multi-classification hinge (margin-based) loss is used between the event detection predictions of our system and the reference sound class activities, while a mean square error (MSE) loss is used for both the azimuth and elevation

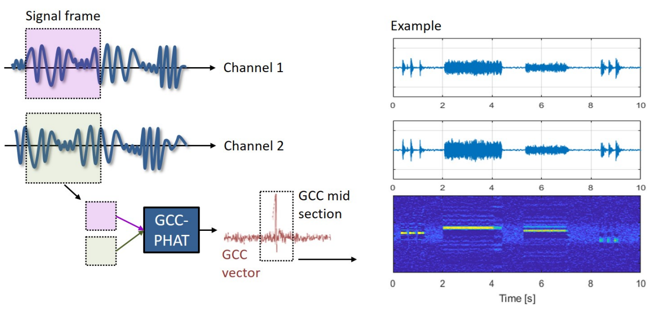


Figure 2: Scheme of the process in which a GCC-gram is produced from the audio signals.

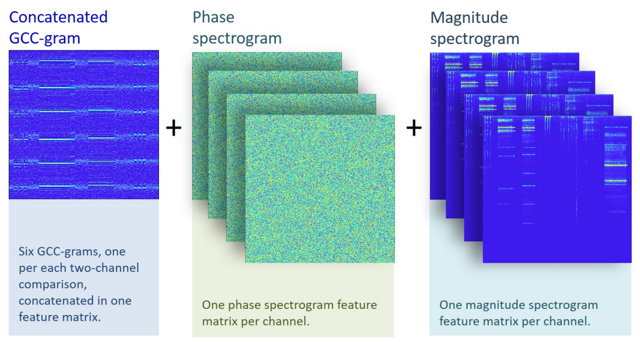

Concatenated GCC-gram
Phase spectrogram
Magnitude spectrogram
Six GCC-grams, one per each two-channel comparison, concatenated in one feature matrix.
One phase spectrogram feature matrix per channel.
One magnitude spectrogram feature matrix per channel.

Figure 3: Scheme of the complete contents of the audio features extracted from the 4-channel signal.

outputs, for the whole matrices (including the garbage values). An additional MSE loss value was calculated for both angles, in which the garbage values were ”masked out” of the MSE difference, using the target matrices. After some tests, the best result were obtained with a weighted sum of these three loss values. Additionally, we used spherical coordinates in all our calculations.

Training was performed by 150 epochs using Adam optimizer with batch size of 8 training samples, drop out rate of 0.5, and initial learning rate of 0.0001, using a cosine annealing scheduler. Early stopping is used to control the network from over-fitting to training split. The network was implemented using Pytorch.

5. RESULTS

5.1. In house tests on DCASE development data

As established in the DCASE Task 3 web page, four different metrics are taken onto account to assess the SELD system performance. Two are directly related with event detection: F-score and error rate (ER). The other two are related with DOA detection: DOA error and frame recall[9]. It must be highlighted that an ideal SELD method will have an error rate of 0, F-score of 100%, DOA error of 0 and frame recall of 100%. Also, as suggested in the page, the four cross-validation folds are to be treated as a single experiment, meaning that metrics are calculated only after training and testing all folds. The results are then compared with those from the baseline SELD system proposed by S. Adavanne et al[1], and the results of using only a part of the input features (GCC-gram only and magnitude-and-phase spectrums only). Such comparison of system results can be seen in Table 1.

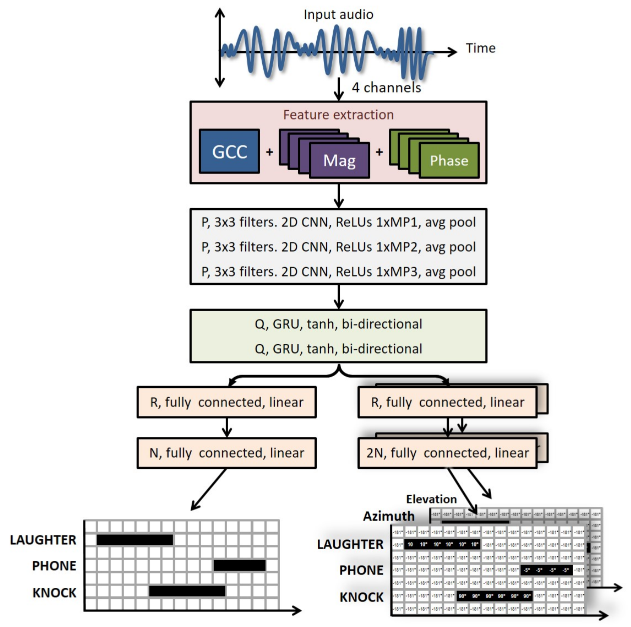

Input audio
Time
4 channels
Feature extraction
GCC
Mag
Phase
P, 3x3 filters. 2D CNN, ReLUs 1xMP1, avg pool
P, 3x3 filters. 2D CNN, ReLUs 1xMP2, avg pool
P, 3x3 filters. 2D CNN, ReLUs 1xMP3, avg pool
Q, GRU, tanh, bi-directional
Q, GRU, tanh, bi-directional
R, fully connected, linear
R, fully connected, linear
N, fully connected, linear
2N, fully connected, linear
Elevation
Azimuth
LAUGHTER
PHONE
KNOCK
LAUGHTER
PHONE
KNOCK

Figure 4: Overview of the architecture of the proposed sound event localization and detection (SELD) system

Model	ER	F-score	DOA error	Frame recall
Baseline (mic.)	0.35	80.0%	30.8	84.0%
GCC-Gram only	0.31	78.6%	29.9	80.9%
Spectrums only	0.25	83.8%	30.8	84.8%
Our system	**0.20**	**87.1%**	**20.4**	**86.4%**

Table 1: Results from the in-house tests.

As can be seen, using only GCC-grams or magnitude-and-phase spectrums as input features produce results in which the four metrics are very close to the baseline (with some slight improvement in the ER metric, that could be attributed to the small changes in the network architecture). However, it is only when the two feature types are fused in the input that the best results are obtained in all metrics. The best improvements over the baseline are present in the error rate (-0.15) and DOA error (-10.4). Then, we can attribute part of this performance jump to the additional prepossessing of audio events provided by GCC-grams.

5.2. Submission to the Task 3 of the DCASE Challenge with evaluation data

For the task 3 DCASE challenge, two sets were submitted, one with a CRNN trained with the fold with the best results in development data, and another with a fusion of the four folds. Table 2 shows the official results obtained in the four metrics considered, and the absolute ranking of each submission, in comparison with the two baseline options, and the best performing system.

Model	ER	F-score	DOA error	Frame recall	Rank
Baseline (mic.)	0.30	83.2%	38.1	83.4	58
Baseline (FOA)	0.28	85.4%	24.6	85.7	48
Our system (1-fold)	0.22	86.3%	19.9	85.6	46
Our system (fusion)	0.22	86.5%	20.8	85.7	45
Best system	**0.08**	**94.7%**	**3.7**	**96.8%**	**1**

Table 2: Results from the DCASE Challenge (Task 3).

These results ranked us in the 16th best out of 24 participating teams.

6. CONCLUSIONS

In this work we describe a system for the simultaneous audio event classification and location, based in the use of regular Fourier spectrograms and our proposed GCC-grams, in order to improve detection and localization performance over a previous baseline. Some additional changes in the CRNN architecture are also included, with the hypothesis of improved robustness over the baseline system. However, the main differentiation of our approach is clearly on the feature extraction side. The results obtained from the cross validation results show that our system performs better than the baseline in all the metrics proposed by the DCASE Challenge coordination team, which suggests that the additional processing at the feature extraction stage we proposed can produce significant additional benefits over an already properly functioning NN architecture.

7. REFERENCES

[1] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, March 2019.

[2] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "Cnn architectures for large-scale audio classification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. [Online]. Available: https://arxiv.org/abs/1609.09430

[3] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 9–13. [Online]. Available: https://arxiv.org/abs/1807.09840

[4] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, August 1976.

[5] X. Xiao, S. Zhao, X. Zhong, D. L. Jones, E. S. Chng, and H. Li, "A learning-based approach to direction of arrival estimation in noisy and reverberant environments," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 2814–2818.

[6] R. Takeda and K. Komatani, "Discriminative multiple sound source localization based on deep neural networks using independent location model," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2016, pp. 603–609.

[7] S. Chakrabarty and E. A. P. Habets, "Broadband doa estimation using convolutional neural networks trained with noise signals," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, Oct 2017, pp. 136–140.

[8] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and detection," *Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019. [Online]. Available: https://arxiv.org/abs/1905.08546

[9] http://dcase.community/challenge2019/task-sound-event-localization-and-detection/.

LANGUAGE MODELLING FOR SOUND EVENT DETECTION WITH TEACHER FORCING AND SCHEDULED SAMPLING

Konstantinos Drossos, Shayan Gharib, Paul Magron, and Tuomas Virtanen

Audio Research Group, Tampere University, Tampere, Finland
{firstname.lastname}@tuni.fi

ABSTRACT

A sound event detection (SED) method typically takes as an input a sequence of audio frames and predicts the activities of sound events in each frame. In real-life recordings, the sound events exhibit some temporal structure: for instance, a "car horn" will likely be followed by a "car passing by". While this temporal structure is widely exploited in sequence prediction tasks (e.g., in machine translation), where language models (LM) are exploited, it is not satisfactorily modeled in SED. In this work we propose a method which allows a recurrent neural network (RNN) to learn an LM for the SED task. The method conditions the input of the RNN with the activities of classes at the previous time step. We evaluate our method using F_1 score and error rate (ER) over three different and publicly available datasets; the TUT-SED Synthetic 2016 and the TUT Sound Events 2016 and 2017 datasets. The obtained results show an increase of 6% and 3% at the F_1 (higher is better) and a decrease of 3% and 2% at ER (lower is better) for the TUT Sound Events 2016 and 2017 datasets, respectively, when using our method. On the contrary, with our method there is a decrease of 10% at F_1 score and an increase of 11% at ER for the TUT-SED Synthetic 2016 dataset.

Index Terms— sound event detection, language modelling, sequence modelling, teacher forcing, scheduled sampling

1. INTRODUCTION

Sound event detection (SED) consists in detecting the activity of classes (onset and offset times) in an audio signal, where the classes correspond to different sound events. (e.g., "baby cry", "glass shatter"). This task finds applications in many areas related to machine listening, such as audio surveillance for smart industries and cities [1, 2], smart meeting room devices for enhanced telecommunications [3, 4], or bio-diversity monitoring in natural environments [5, 6]. SED is a challenging research task since the sound events are of very diverse nature, which might be unknown a priori in real-life recordings. Besides, they often overlap in time, a problem termed as polyphonic SED. Significant advances in SED were made recently thanks to the advent of deep learning [7]. The recurrent neural network (RNN) have proven particularly promising [8, 9] as they are able to model the temporal discriminant representations for sound events. More recently, these have been stacked with convolutional layers, resulting in convolutional recurrent neural networks (CRNN) which yield state-of-the-art results [10, 11].

In real-life recordings, the various sound events likely temporal structures within and across events. For instance, a "footsteps" event might be repeated with pauses in between (intra-event structure). On the other hand, "car horn" is likely to follow or precede the "car passing by" sound event (inter-events structure). Although these temporal structures vary with the acoustic scene and the actual sound events classes, they exist and can be exploited in the SED task. Some previous studies focus on exploiting these temporal structures. For example, in [9], the authors propose to use hidden Markov models (HMMs) to control the duration of each sound event predicted with a deep neural network (DNN). Although the results show some improvement with the usage of HMMs, the approach is a hybrid one and it requires a post processing step, which might be limited compared to an non-hybrid, DNN-based approach. In [12] and [13], the connectionist temporal classification (CTC) [14] loss function is used for SED: the output of the DNN is modified in order to be used with the CTC. Although the usage of CTC seems to be promising, CTC needs modification in order to be used for SED, it is a complicated criterion to employ, and it was developed to solve the problem where there is no frame-to-frame alignment between the input and output sequences [14]. Thus, there might be the case that using a different method for SED language modelling could provide better results than CTC. Finally, in [13], the authors also employ N-grams, which require pre and post processing stages, and use the class activities as extra input features. However, the latter approach did not perform better than a baseline which did not employ any language model.

In this paper we propose an RNN-based method for SED that exploit the temporal structures within and across events of audio scenes without the aforementioned drawbacks of the previous approaches. This method is based on established practices from other scientific disciplines that deal with sequential data (e.g., machine translation, natural language processing, speech recognition). It consists in using the output of the classifier as an extra input to the RNN in order to learn a model of the temporal structures of the output sequence (referred to as language model), a technique called *teacher forcing* [15]. Besides, this extra input of the RNN is chosen as a combination of the ground truth and predicted classes. This strategy, known as *schedule sampling* [16], consists in first using the ground truth activities and further replacing them by the predictions. This allows the RNN to learn a robust language model from clean labels, without introducing any mismatch between the training and inference processes.

The rest of the paper is organized as follows. In Section 2 we present our method. Section 3 details the experimental protocol and Section 4 presents the results. Section 5 concludes the paper.

2. PROPOSED METHOD

We propose a system that consists of a DNN acting as a feature extractor, an RNN that learns the temporal structures withing and across events (i.e. a language model), and a feed-forward neural network (FNN) acting as a classifier. Since we focus on designing

https://doi.org/10.33682/1dze-8739

an RNN that is able to learn a language model over the sound events, the RNN takes as inputs the outputs of both the DNN and the FNN. The code for our method can be found online[1].

2.1. Baseline system

The DNN takes as an input a time-frequency representation of an audio signal denoted $\mathbf{X} \in \mathbb{R}_{\geq 0}^{T \times F}$, where T and F respectively denote the number of time frames and features. It outputs a latent representation:

$$\mathbf{H} = \text{DNN}(\mathbf{X}), \tag{1}$$

where $\mathbf{H} \in \mathbb{R}^{T \times F'}$ is the learned representation with F' features. Then, the RNN operates over the rows of $\mathbf{H}$ as

$$\mathbf{h}'_t = \text{RNN}(\mathbf{h}_t, \mathbf{h}'_{t-1}), \tag{2}$$

where $t = 1, 2, \ldots, T$, $\mathbf{h}'_0 = \{0\}^{F''}$, $\mathbf{h}'_t \in [-1, 1]^{F''}$, and F'' is the amount of features that the RNN outputs at each time-step. Finally, the FNN takes $\mathbf{h}'_t$ as an input and outputs the prediction $\hat{\mathbf{y}}_t$ for the time-step t as:

$$\hat{\mathbf{y}}_t = \sigma(\text{FNN}(\mathbf{h}'_t)), \tag{3}$$

where σ is the sigmoid function, and $\hat{\mathbf{y}}_t \in [0, 1]^C$ is the predicted activity of each of the C classes.

The DNN, the RNN, and the FNN are simultaneously optimized by minimizing the loss $\mathcal{L}(\hat{\mathbf{Y}}, \mathbf{Y}) = \sum_t \mathcal{L}_t(\hat{\mathbf{y}}_t, \mathbf{y}_t)$ with:

$$\mathcal{L}_t(\hat{\mathbf{y}}_t, \mathbf{y}_t) = \sum_{c=1}^{C} y_{t,c} \log(\hat{y}_{t,c}) + (1 - y_{t,c}) \log(1 - \hat{y}_{t,c}), \tag{4}$$

where $y_{t,c}$ and $\hat{y}_{t,c}$ are the ground truth and predicted activities, respectively, of the c-th class at the t-th time-step.

2.2. Teacher forcing

The modeling in Eq. (2) shows that the RNN learns according to its input and its previous state [15, 16]. In order to allow the RNN to learn a language model over the output (i.e. the sound events), we propose to inform the RNN of the activities of the classes of the sound events at the time step $t - 1$. That is, we condition the input to the RNN as:

$$\mathbf{h}'_t = \text{RNN}(\mathbf{h}_t, \mathbf{h}'_{t-1}, \mathbf{y}'_{t-1}), \tag{5}$$

where $\mathbf{y}'_{t-1}$ is the vector with the activities of the classes of the sound events at time step $t - 1$, and $\mathbf{y}'_0 = \{0\}^C$. This technique is termed as teacher forcing [15], and is widely employed in sequence prediction/generation tasks where the output sequence has an inherent temporal model/structure (e.g., machine translation, image captioning, speech recognition) [17, 18, 19]. By using this conditioning of the RNN, the RNN can learn a language model over the output tokens of the classifier [15, 16]. In SED, this results in letting the RNN learn a language model over the sound events, e.g., which sound events are more likely to happen together and/or in sequence, or how likely is a sound event to keep being active, given the previous activity of the sound events. Teacher forcing is different from what was proposed in [13], as the latter approach conditioned the DNN (not the RNN) with the class activities: such an approach yielded poor results, intuitively explained by having $\mathbf{y}'_{t-1}$ dominated by the information in $\mathbf{X}$ through the sequence of the CNN blocks.

[1]`https://github.com/dr-costas/SEDLM`

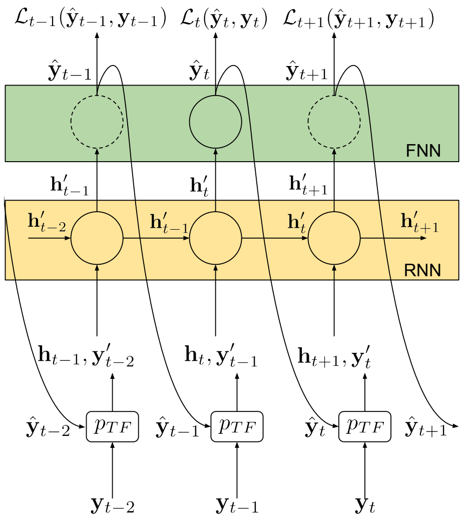


Figure 1: Proposed method of teacher forcing with scheduled sampling.

2.3. Scheduled sampling

The activity vector $\mathbf{y}'_{t-1}$ can be either the ground truth data (i.e., $\mathbf{y}_{t-1}$), or the predictions of the classifier (i.e., $\hat{\mathbf{y}}_{t-1}$). In the former case, the RNN is likely to start learning the desired language model from the first updates of the weights. At each time step t, the RNN will take as input the ground truth activities of the classes, thus being able to exploit this information from the very first weight updates. However, these ground truth values are not available at the inference stage: these would be replaced by the estimates $\hat{\mathbf{y}}_{t-1}$, which would create a mismatch between the training and testing processes. Besides, an RNN trained using only the true class activities is very likely to be sensitive to the prediction errors in $\hat{\mathbf{y}}_{t-1}$. Finally, we empirically observed that using $\mathbf{y}_{t-1}$ with the SED datasets, which are of relatively small size, results in a very poor generalization of the SED method.

A countermeasure to the above is to use the predictions $\hat{\mathbf{y}}_{t-1}$ as $\mathbf{y}'_{t-1}$, which allows the RNN to compensate for the prediction errors. However, during the first weight updates, the predicted $\hat{\mathbf{y}}_{t-1}$ is very noisy and any error created at a time step t is propagated over time, which results in accumulating more errors down the line of the output sequence. This makes the training process very unstable and is likely to yield a poor SED performance.

To exploit the best of both approaches, we propose to use the scheduled sampling strategy [16]: the ground truth class activities are used during the initial epochs, and they are further gradually replaced by the predicted class activities. This gradual replacement is based on a probability p_{TF} of picking $\mathbf{y}_{t-1}$ over $\hat{\mathbf{y}}_{t-1}$ as $\mathbf{y}'_{t-1}$ that decreases over epochs. Different functions can be used for the calculation of p_{TF} (e.g., exponential, sigmoid, linear). Here, we employ a model of exponential decrease of p_{TF}:

$$p_{\text{TF}} = \min(p_{\text{max}}, 1 - \min(1 - p_{\text{min}}, \frac{2}{1 + e^{\beta}} - 1)), \tag{6}$$

where $\beta = -i\gamma N_b^{-1}$, i is the index of the weight update (i.e., how many weight updates have been performed), N_b is the amount of batches in one epoch, and p_{max}, p_{min}, and γ are hyper-parameters to be tuned. p_{max} and p_{min} are the maximum and minimum probabilities of selecting $\hat{\mathbf{y}}_t$, and γ controls the slope of the curve of p_{TF} for a given N_b and as i increases. We use a minimum probability $p_{\min}$ because we experimentally observed that if we solely use $\mathbf{y}_{t-1}$ as $\mathbf{y}'_{t-1}$ even in the first initial weight updates, then the SED method overfits. The usage of $p_{\min}$ counters this fact. On the other hand, we use a maximum probability $p_{\max}$ in order to allow the usage of $\mathbf{y}_{t-1}$ as $\mathbf{y}'_{t-1}$ at the later stages of the learning process. We do this because the length of a sequence in SED is usually over 1000 time-steps and any error in $\hat{\mathbf{y}}_t$ is accumulated in this very long sequence, resulting in hampering the learning process. The usage of $p_{\max}$ offers a counter measure to this, by allowing the usage of ground truth values $\mathbf{y}_t$ instead of predicted and noisy values. This method is illustrated in Figure 1.

3. EVALUATION

We evaluate our method using the CRNN from [10], and we employ synthetic and real-life recordings datasets to illustrate the impact of the language model learned by our method.

3.1. Data and feature extraction

The synthetic dataset is the TUT-SED Synthetic 2016, used in [10], and consisting of 100 audio files which are synthetically created out of isolated sound events of 16 different classes. These classes are: *alarms and sirens*, *baby crying*, *bird singing*, *bus*, *cat meowing*, *crowd applause*, *crowd cheering*, *dog barking*, *footsteps*, *glass smash*, *gun shot*, *horse walk*, *mixer*, *motorcycle*, *rain*, and *thunder*. Each audio file contains a maximum of N number of randomly selected target classes, where N is sampled from the discrete uniform distribution $U(4, 9)$, and the maximum number of simultaneously active (polyphony) sound events is 5. The audio files do not contain any background noise. The audio files amount to a total of 566 minutes of audio material, and according to the splits introduced by [10], roughly 60% of the data are dedicated to training, 20% to validation, and 20% to testing split. More information about the dataset can be found online[2].

We employ two real-life recording datasets, which were part of the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge datasets for SED in real life audio task: the TUT Sound Events 2016 and the TUT Sound Events 2017 [20]. The TUT Sound Events 2016 dataset contains sound events recorded in two environments: home (indoor), which contains 11 classes, and residential area (outdoor), which contains 7 classes. The classes for the home environment are: *(object) rustling*, *(object) snapping*, *cupboard*, *cutlery*, *dishes*, *drawer*, *glass jingling*, *object impact*, *people walking*, *washing dishes*, and *water tap running*. The classes for the residential area environment are: *(object) banging*, *bird singing*, *car passing by*, *children shouting*, *people speaking*, *people walking*, and *wind blowing*. The TUT Sound Events 2017 dataset contains recordings in a street environment and contains 6 different classes. These classes are: *brakes squeaking*, *car*, *children*, *large vehicle*, *people speaking*, and *people walking*. For both datasets, we use the cross-fold validation split proposed in the DCASE 2016 and 2017 challenges. More information about the classes, the cross-fold setting, and the recordings of the datasets can be found online[3,4].

[2]`http://www.cs.tut.fi/sgn/arg/taslp2017-crnn-sed/tut-sed-synthetic-2016`

[3]`http://www.cs.tut.fi/sgn/arg/dcase2016`

[4]`http://www.cs.tut.fi/sgn/arg/dcase2017/challenge`

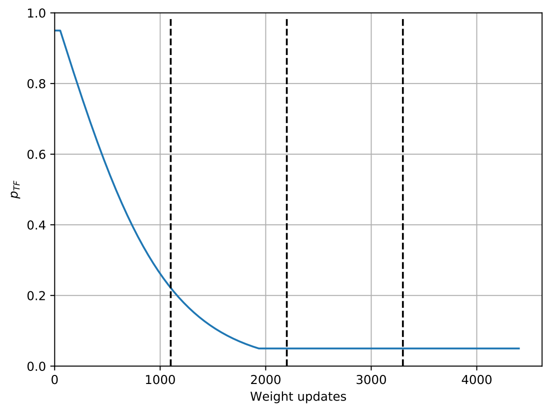


Figure 2: The value of p_{TF} with consecutive weight updates with $p_{\min} = 0.05$, $p_{\max} = 0.95$, and $N_b = 44$. The vertical dashed lines indicate steps of 25 epochs (i.e. 25, 50, 75 epochs).

The synthetic dataset has randomly selected and placed sound events, therefore not exhibiting any underlying temporal structure of sound events. We thus expect the performance of our method to be similar to a method without language modeling on the synthetic dataset. Contrarily, the real-life recording datasets exhibit some underlying temporal structures in the sound events, therefore we expect our method to perform better than a method without language modelling on these datasets.

As input features $\mathbf{X}$ we use non-overlapping sequences of $T = 1024$ feature vectors. These consist of $F = 40$ log mel-bands, extracted using a short-time Fourier transform using a 22 ms Hamming window, 50% overlap and no zero padding. We normalize the extracted feature vectors from each dataset to have zero mean and unit variance, employing statistics calculated on the training split of each corresponding dataset.

3.2. System and hyper-parameters

As our DNN we use the three convolutional neural network (CNN) blocks from the system in [10], each consisting of a CNN, a batch normalization function, a max-pooling operation, a dropout function, and a rectified linear unit (ReLU). The kernels of the CNNs are square with a width of 5, a stride of 1, and a padding of 2 in both directions. There are 128 filters for each CNN. The kernel and the stride for the first max-pooling operation are $\{1, 5\}$, for the second $\{1, 4\}$, and for the third $\{1, 2\}$. These result in $F' = 128$ for $\mathbf{H}$. All CNN blocks use a dropout of 25% at their input, and the last CNN block also uses a dropout of 25% at its output. As our RNN we use a gated recurrent unit (GRU) with $F'' = 128$ and our FNN is a single-layer feed-forward network with the output size defined according to the amount of classes in each dataset: $C = 16$ for TUT-SED Synthetic 2016, $C = 11$ and $C = 7$ for the home and residential area scenes of the TUT Sound Events 2016, and $C = 6$ for the TUT Sound Events 2017. To optimize the weights we employed the Adam optimizer [21] with default values. We employ a batch size of 8 and we stop the training when the loss for the validation data is not decreasing for 50 consecutive epochs. Finally, we set the hyper-parameters for p_{TF} at $\gamma = 12^{-1}$, $p_{\min} = 0.05$, and $p_{\max} = 0.9$. In Figure 2 is the value of p_{TF} for consecutive weight updates of $N_b = 44$ and for 100 epochs.

Empirically we observed that when using the TUT Sound

Events 2017, there are some irregular spikes of relatively high gradients in different batches during training. To alleviate this issue, we clipped the ℓ_2-norm of the gradient of all weights in each layer of our system to a value of 0.5. Additionally, we also observed that for the TUT Sound Events 2017 and TUT-SED Synthetic 2016 datasets, our method performed significantly better when we decreased the learning rate of the optimizer to $5e-4$. Therefore, we employed the above mentioned gradient clipping and modified learning rate for our method, when using the aforementioned datasets. Finally, for the TUT Sound Events 2016, we employed a binarized version of $\mathbf{y}'$ denoted $\mathbf{y}''$, such that $y''_{t,c}=1$ if $y'_{t,c}\geq 0.5$, and $y''_{t,c}=0$ otherwise.

All the above hyper-parameters were tuned using the cross validation set up for the TUT Sound Events 2016 and 2017 datasets provided by DCASE challenges, and the validation split provided in [10] for the TUT-SED Synthetic 2016 dataset.

3.3. Metrics

We measure the performance of our method using the frame based F_1 score and the error rate (ER), according to previous studies and the DCASE Challenge directions [10, 13]. For the real-life datasets, the F_1 and ER are the averages among the provided folds (and among the different acoustic scenes for the 2016 dataset), while for the synthetic dataset the F_1 and ER are obtained on the testing split. Finally, we repeat four times the training and testing process for all datasets, in order to obtain a mean and standard deviation (STD) for F_1 and ER.

3.4. Baseline

As a baseline we employ the system presented in [10], that does not exploit any language model. We do not apply any data augmentation technique during training and we use the hyper-parameters presented in the corresponding paper. This system is referred to as "Baseline".

When using our method with the TUT Sound Events 2017 and TUT-SED Synthetic 2016 datasets, we employ a modified learning rate for the optimizer and we clip the ℓ_2-norm of the gradient for all weights. To obtain a thorough and fair assessment of the performance of our method, we utilize a second baseline for this dataset: we use again the system presented in [10], but we employ the above-mentioned gradient clipping and modified learning rate. We denote this modified baseline as "modBaseline".

Finally, we compare our method to the best results presented in [13] which are obtained by employing N-grams as a post-processing to learn a language model. We report the results of this method on the TUT Sound Events 2016 datasets, as these are the only ones in the corresponding paper that are based on a publicly available dataset. It must be noted that in [13] was proposed the usage of $\mathbf{y}'_{t-1}$ as extra input features and the usage of CTC, but the results were inferior to the N-grams approach. Specifically, the per frame F_1 score was 0.02 and 0.04 lower and ER was 0.02 and 0.15 higher with the usage of $\mathbf{y}'_{t-1}$ as an extra input feature and the usage of CTC, respectively, compared to the N-grams approach.

4. RESULTS & DISCUSSION

In Table 1 are the obtained results for all the employed datasets. We remark that using the proposed language model improves the performance of SED in the real-life datasets. Specifically, for the TUT Sound Events 2016 dataset there is an improvement of 0.09 in the F_1 score and 0.07 for the ER. For the TUT Sound Events 2017, there is a 0.02 improvement in F_1 and 0.02 improvement in ER. These results clearly show that the employment of language modelling was beneficial for the SED method, when a real life datset was used. This is expected, since in a real life scenario the sound events exhibit temporal relationships. For example, "people speaking" and "people walking" or "washing dishes" and "water tap running" are likely to happen together or one after the other.

Table 1: Mean and STD (Mean/STD) of F_1 (higher is better) and ER (lower is better). For the method [13] only the mean is available.

	Baseline	**modBaseline**	**[13]**	**Proposed**
	TUT Sound Events 2016 dataset			
$\mathbf{F_1}$	0.28/0.01	–	0.29	0.37/0.02
ER	0.86/0.02	–	0.94	0.79/0.01
	TUT Sound Events 2017 dataset			
$\mathbf{F_1}$	0.48/0.01	0.49/0.01	–	0.50/0.02
ER	0.72/0.01	0.70/0.01	–	0.70/0.01
	TUT-SED Synthetic 2016 dataset			
$\mathbf{F_1}$	0.58/0.01	0.62/0.01	–	0.54/0.01
ER	0.54/0.01	0.49/0.01	–	0.61/0.02

On the contrary, from Table 1 we observe that there is a decrease in performance with our method on the synthetic data. Specifically, there is a 0.04 (or 0.08 when compared to modBaseline) decrease in F_1 and 0.07 (or 0.12 when compared to modBaseline) increase in ER. This clearly indicates that using a language model has a negative impact when the synthetic dataset is used. The sound events in the synthetic dataset do not exhibit any temporal relationships and, thus, the language model cannot provide any benefit to the SED method. We suggest that in such a scenario, the network focuses on learning a language model that does not exist in the data instead of solely trying to accurately predict the events on a frame-wise basis: this explains the drop in performance compared to the baseline method. Overall, this difference in performance between the two types of datasets strongly suggests that our method learns a language model over the activities of the sound events.

Finally, our system significantly outperforms the previous method [13] on the TUT Sound Events 2016 dataset. This shows that learning a language model is more powerful than crafting it as a post-processing.

5. CONCLUSIONS

In this paper we presented a method for learning a language model for SED. Our method focuses on systems that utilize an RNN before the the last layer of the SED system, and consists of conditioning the RNN at a time step t with the activities of sound events at the time step $t-1$. As activities for $t-1$ we select the ground truth early on the training process, and we gradually switch to the prediction of the classifier as the training proceeds over time. We evaluate our method with three different and publicly available datasets, two from real life recordings and one synthetic dataset. The obtained results indicate that with our method, the utilized SED system learned a language model over the activities of the sound events, which is beneficial when used on real life datasets.

In future work, we will conduct a more in-depth analysis of the learned language model and of the SED performance per class.

6. REFERENCES

[1] M. Crocco, M. Cristani, A. Trucco, and V. Murino, "Audio surveillance: A systematic review," *ACM Comput. Surv.*, vol. 48, no. 4, pp. 52:1–52:46, Feb. 2016.

[2] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Audio surveillance of roads: A system for detecting anomalous sounds," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 1, pp. 279–288, Jan 2016.

[3] T. Butko, F. G. Pla, C. Segura, C. Nadeu, and J. Hernando, "Two-source acoustic event detection and localization: Online implementation in a smart-room," in *Proc. European Signal Processing Conference*, Aug 2011.

[4] C. Busso, S. Hernanz, Chi-Wei Chu, Soon-il Kwon, Sung Lee, P. G. Georgiou, I. Cohen, and S. Narayanan, "Smart room: participant and speaker localization and identification," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, March 2005.

[5] B. J. Furnas and R. L. Callas, "Using automated recorders and occupancy models to monitor common forest birds across a large geographic region," *The Journal of Wildlife Management*, vol. 79, no. 2, pp. 325–337, 2015.

[6] T. A. Marques, L. Thomas, S. W. Martin, D. K. Mellinger, J. A. Ward, D. J. Moretti, D. Harris, and P. L. Tyack, "Estimating animal population density using passive acoustics," *Biological Reviews*, vol. 88, no. 2, pp. 287–309, 2013.

[7] E. Benetos, D. Stowell, and M. D. Plumbley, *Approaches to Complex Sound Scene Analysis*. Springer International Publishing, 2018, pp. 215–242.

[8] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016.

[9] T. Hayashi, S. Watanabe, T. Toda, T. Hori, J. Le Roux, and K. Takeda, "Duration-controlled LSTM for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 11, pp. 2059–2070, November 2017.

[10] E. Çakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, June 2017.

[11] S. Adavanne, P. Pertilä, and T. Virtanen, "Sound event detection using spatial features and convolutional recurrent neural network," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017.

[12] Y. Wang and F. Metze, "A first attempt at polyphonic sound event detection using connectionist temporal classification," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017.

[13] G. Huang, T. Heittola, and T. Virtanen, "Using sequential information in polyphonic sound event detection," in *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, Sep. 2018, pp. 291–295.

[14] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: ACM, 2006, pp. 369–376. [Online]. Available: http://doi.acm.org/10.1145/1143844.1143891

[15] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, June 1989.

[16] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 1171–1179. [Online]. Available: http://dl.acm.org/citation.cfm?id=2969239.2969370

[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR)*, 2015.

[18] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 3104–3112. [Online]. Available: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

[19] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3156–3164.

[20] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," in *24th European Signal Processing Conference 2016 (EUSIPCO 2016)*, Budapest, Hungary, 2016.

[21] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations*, May 2015.

CONVOLUTIONAL RECURRENT NEURAL NETWORK AND DATA AUGMENTATION FOR AUDIO TAGGING WITH NOISY LABELS AND MINIMAL SUPERVISION

Janek Ebbers, Reinhold Haeb-Umbach

Paderborn University, Department of Communications Engineering, Paderborn, Germany
{ebbers, haeb}@nt.upb.de

ABSTRACT

In this paper we present our audio tagging system for the DCASE 2019 Challenge Task 2. We propose a model consisting of a convolutional front end using log-mel-energies as input features, a recurrent neural network sequence encoder and a fully connected classifier network outputting an activity probability for each of the 80 considered event classes. Due to the recurrent neural network, which encodes a whole sequence into a single vector, our model is able to process sequences of varying lengths. The model is trained with only little manually labeled training data and a larger amount of automatically labeled web data, which hence suffers from label noise. To efficiently train the model with the provided data we use various data augmentation to prevent overfitting and improve generalization. Our best submitted system achieves a label-weighted label-ranking average precision (lwlrap) of 75.5% on the private test set which is an absolute improvement of 21.7% over the baseline. This system scored the second place in the teams ranking of the DCASE 2019 Challenge Task 2 and the fifth place in the Kaggle competition “Freesound Audio Tagging 2019” with more than 400 participants. After the challenge ended we further improved performance to 76.5% lwlrap setting a new state-of-the-art on this dataset.

Index Terms— audio tagging, label noise, data augmentation

1. INTRODUCTION

Environmental sound recognition has recently attracted increased interest, not only in academia. Numerous commercial applications can benefit from a reliable acoustic scene analysis, such as ambient assisted living, autonomous driving and various monitoring and diarization tasks. Within environmental sound recognition the tasks of Sound Event Detection (SED), Audio Tagging and Acoustic Scene Classification (ASC) [1] can be distinguished, which differ in the level of detail obtained about the acoustic environment. While SED makes predictions at frame level, Audio Tagging and ASC make predictions at sequence level.

Because frame level annotations (so-called *strong labels*) are difficult and time-consuming to obtain, current large-scale datasets, such as Google’s AudioSet [2], only provide sequence level labels (*weak labels*). Further, for many applications frame level predictions are not required, which, together with the availability of weakly labeled large-scale datasets, results in an increased popularity of Audio Tagging.

Although weak annotations are easier to obtain than strong annotations, they still require human annotators. To avoid this necessity completely, one line of research is devoted to develop semi-supervised approaches which directly benefit from unlabeled data [3, 4], which is abundant. However, there are also tremendous amounts of data coming with other modalities and meta-data, which can potentially be exploited to derive labels automatically. In the dataset considered in this contribution, labels had been generated for the web data using video-level predictions. Note, that those automatically generated labels certainly contain errors, which is why they are called *noisy* labels. Until now there are only few works addressing the impact of noisy labels for sound recognition [5].

Data augmentation is another common approach to increase the amount of labeled training data and improve generalization. It has been shown to improve classifier performance on many tasks, including speech recognition [6] and audio classification [7, 8, 9].

In this contribution we tackle the DCASE 2019 Challenge Task 2 [10], where the main research objective is the following: How can we train a high performance Audio Tagging system given relatively little data with reliable labels but a larger amount of mismatched data with noisy labels? We primarily tackle this by exploring different methods for data augmentation, while we spend less time on neural network architecture tuning. We show that frequency warping and time and frequency masking for data augmentation significantly improve performance. Further, our experiments tend to suggest, that our model is robust against label noise, as it achieves state-of-the-art performance without any particular treatment of label noise. While the methods investigated for dealing with label noise did individually not result in classifier improvements, they nevertheless increased the diversity of the models trained, resulting eventually in performance improvement through system combination. Please note that our system is publicly available on github.[1]

The rest of the paper is structured as follows. After briefly describing the considered task in Section 2, we outline our neural network architecture in Section 3. After presenting the data augmentation methods in Section 4, our training procedure and experiments are given in Sections 5 and 6, respectively. Conclusions are drawn in Section 7.

2. TASK DESCRIPTION

The DCASE 2019 Challenge Task 2 “Audio Tagging with noisy labels and minimal supervision” [10] is a follow-up of the DCASE 2018 Challenge Task 2 “General-purpose audio tagging of Freesound content with AudioSet labels” [11]. As in the 2018 edition the challenge was hosted on Kaggle[2] with more than 400 participants. The provided dataset “FSDKaggle2019” consists of two subsets: a small *curated* set of 4970 manually labeled audio clips and a *noisy* set of 19815 audio clips where labels were automatically derived from video-level predictions from a variety of

This work has been supported by Deutsche Forschungsgemeinschaft under contract no. HA 3455/15-1 within the Research Unit FOR 2457 (acoustic sensor networks).

[1] `https://github.com/fgnt/upb_audio_tagging_2019`
[2] `https://www.kaggle.com/c/freesound-audio-tagging-2019`

https://doi.org/10.33682/57xx-t679

pre-trained audio models. A vocabulary of 80 sound events is used where multiple events can be active at a time resulting in a multi-label classification problem.

3. MODEL

3.1. Feature Extraction

First, we perform an STFT with a frame length of $40\,\mathrm{ms}$ (1764 samples) and a hop size of $20\,\mathrm{ms}$ (882 samples) on the provided $44.1\,\mathrm{kHz}$ audio signals without resampling. For each frame we then extract 128 log-mel-band-energy features with $f_{\text{min}}{=}50\,\text{Hz}$ and $f_{\text{max}}{=}16\,\text{kHz}$. Lastly, we substract the global mean of each feature and then divide by the global standard deviation over all features.

3.2. Neural Network Architecture

Our proposed deep learning based model is outlined in Table 1. The neural network consists of a convolutional (conv.), a recurrent, and a fully connected module. We expect a four dimensional input to our model of shape $B{\times}C{\times}F{\times}N_m$ with B, C, F, N_m being the mini-batch size, number of channels, number of features and number of frames in the m-th input signal, respectively, where $C{=}1$ and $F{=}128$ are fix. In the following the signal index m is neglected.

The convolutional module combines a 2d CNN and a 1d CNN. The 2d CNN consists of five conv. blocks, with each block comprising one or two conv. layers and a max pooling layer. While the first four blocks have two conv. layers the last block only has a single one. In each block the number of channels is doubled starting from 16 while the number of features are halfed by max pooling. The number of time steps are also halfed in the first three blocks while being unchanged in the last two blocks. This results in an output of shape $B{\times}C'{\times}F'{\times}N'$ with $C'{=}256$, $F'{=}4$ and $N' = \lceil \frac{N}{8} \rceil$. Each 2d conv. layer uses a kernel size of $3{\times}3$ and is followed by batch normalization and ReLU activation.

While the 2d CNN is meant to extract high-level feature maps from the log-mel-band-energy spectrogram, the 1d CNN (or TDNN) is meant to provide holistic representations by jointly processing all frequencies and channels of adjacent frames. Therefore it takes the reshaped output ($B \times C'{\cdot}F' \times N'$) of the 2d CNN as input and applies three 1d conv. layers with 256 hidden channels each. Each 1d conv. layer uses a kernel size of 3 and is followed by batch normalization and ReLU activation.

The output of the CNN is then fed into a recurrent sequence encoder. We use two layers of Gated Recurrent Units (GRUs) with 256 units per layer. Only the last output vector of each sequence in a batch is forwarded to the classification network.

The fully connected classification network conists of one hidden layer with 256 hidden units and ReLU activation function and the final classification layer with Sigmoid activation outputting scores between 0 and 1 for each of the 80 target event classes.

4. DATA AUGMENTATION

Because there is only little data available, efficient data augmentation is crucial to prevent overfitting and improve generalization capabilities of the system. In the following we outline the data augmentation methods that we combined during model training. All augmentation methods are performed on the fly during training yielding an extremely large number of possible training samples.

4.1. Mixup

Mixup [12] is a data augmentation technique originating from classification tasks where a new training sample is generated as a weighted average of two samples from the dataset:

$$\tilde{\mathbf{x}}_i = \lambda \mathbf{x}_i + (1-\lambda)\mathbf{x}_j,\ \lambda \in [0, 1].$$

Similarly their one-hot encoded targets are combined to a soft target vector:

$$\tilde{\mathbf{y}}_i = \lambda \mathbf{y}_i + (1-\lambda)\mathbf{y}_j.$$

Although for classification tasks mixup results in ambiguous samples (which probably wouldn't make a lot of sense to a human either) it has shown to improve generalization and robustness of the trained classifier networks. Mixup has successfully been used for general purpose audio tagging, e.g., in [9] in the DCASE 2018 Challenge.

For audio tagging (as opposed to classification) the input audio may already be a superposition of multiple sources. Thus, mixing two or more audio signals together yields a new valid audio signal with an increased number of active events. Therefore, instead of building a weighted average we superpose two waveforms as follows:

$$\tilde{x}_i(t) = \lambda_0 x_i(t) + \gamma\lambda_1 \frac{\max(|x_i|)}{\max(|x_j|)} x_j(t-\tau)$$

with

$$\begin{aligned}
&\gamma \sim \mathcal{B}(2/3),\\
&\tau \sim \mathcal{U}(\max(-T_j, T_i - 30\,\mathrm{s}), \min(T_i, 30\,\mathrm{s} - T_j)),\\
&T_j \le 1.1 \cdot T_i,\\
&\lambda_m = a_m^{2b_m - 1};\ a_m \sim \mathcal{U}(1, 2);\ b_m \sim \mathcal{B}(1/2);\ m{\in}\{0, 1\}
\end{aligned}$$

where $\mathcal{B}$ denotes the Bernoulli distribution. Putting this equation into words we

- perform mixup only with a probability of $2/3$,
- only mixup signals which are shorter than 1.1 times the base signal x_i,
- allow mixup to lengthen the signal as long as it does not exceed the maximum length of $30\,\mathrm{s}$,
- normalize signals to the maximum value of the base signal x_i,
- attenuate or amplify each normalized signal by a random factor.

Table 1: Convolutional Recurrent Neural Network for Audio Tagging with output shapes of each block. Each ConvXd uses a kernel size of three and a stride of one and includes BatchNorm and ReLU.

Block	output shape
LogMel(128)	$B{\times}1{\times}128{\times}N$
GlobalNorm	$B{\times}1{\times}128{\times}N$
2×Conv2d(16)	$B{\times}16{\times}128{\times}N$
Pool2d(2×2)	$B{\times}16{\times}64{\times}\lceil N/2\rceil$
2×Conv2d(32)	$B{\times}32{\times}64{\times}\lceil N/2\rceil$
Pool2d(2×2)	$B{\times}32{\times}32{\times}\lceil N/4\rceil$
2×Conv2d(64)	$B{\times}64{\times}32{\times}\lceil N/4\rceil$
Pool2d(2×2)	$B{\times}64{\times}16{\times}\lceil N/8\rceil$
2×Conv2d(128)	$B{\times}128{\times}16{\times}\lceil N/8\rceil$
Pool2d(2×1)	$B{\times}128{\times}8{\times}\lceil N/8\rceil$
Conv2d(256)	$B{\times}256{\times}8{\times}\lceil N/8\rceil$
Pool2d(2×1)	$B{\times}256{\times}4{\times}\lceil N/8\rceil$
Reshape	$B{\times}1024{\times}\lceil N/8\rceil$
3×Conv1d(256)	$B{\times}256{\times}\lceil N/8\rceil$
2×GRU(256)	$B{\times}256$
$\text{fc}_{\text{ReLU}}(256)$	$B{\times}256$
$\text{fc}_{\text{Sigmoid}}(80)$	$B{\times}80$

We also do not build a weighted average of the individual targets, but simply combine all tags into a single n-hot encoded target $\tilde{\mathbf{y}}_i$.

4.2. Frequency Warping

Recently, SpecAugment [13] was introduced as a simple yet efficient augmentation method of log-mel spectrograms for automatic speech recognition. It uses three different distortions namely time warping, frequency masking, and time masking. In our experiments, however, we found that for audio tagging warping the spectrogram on the frequency axis yielded better performance than time warping. Hence, we exchange the time warping by frequency warping in our version of SpecAugment which is explained in this and the following two sections.

We consider the log-mel spectrogram as an image here with width T and height F. Warping the vertical (frequency) axis of the image is controlled by the cutoff frequency

$$f_\text{c} \sim \mathcal{E}(0.5 \cdot F),$$

where $\mathcal{E}$ denotes the exponential distribution parameterized by the scale $\beta = 0.5 \cdot F$, and by the warping factor

$$\alpha = (1+u)^{2s-1}; \quad u \sim \mathcal{E}(0.07);\ s \sim \mathcal{B}(1/2).$$

Fig. 1 shows the resulting piece-wise linear warping function. Note that f_c can be larger than F, which results in pure stretching/compressing the whole spectrogram.

It is worth noting that the frequency warping performed here is very similar to Vocal Tract Length Pertubation [14].

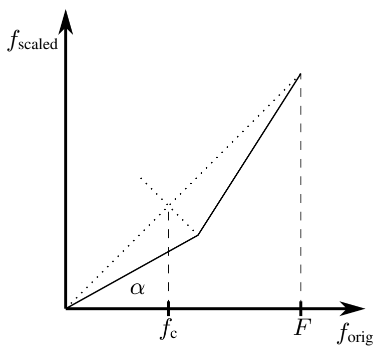


Figure 1: piece-wise linear frequency warping function.

4.3. Frequency Masking

We randomly mask H consecutive mel frequencies in the range $[f_0, f_0 + H]$, where H and f_0 are drawn from uniform distributions

$$H \sim \mathcal{U}(0, H_\text{max})$$
$$f_0 \sim \mathcal{U}(0, F - H)$$

with $H_\text{max} = 16$.

4.4. Time Masking

We randomly mask W consecutive time frames in the range $[n_0, n_0 + W]$, where W and n_0 are drawn from uniform distributions

$$W \sim \mathcal{U}(0, \min(W_\text{max}, p \cdot N))$$
$$n_0 \sim \mathcal{U}(0, T - W)$$

with $W_\text{max} = 70$ and $p = 0.2$.

5. TRAINING

5.1. Data

For training we use both the data with curated labels as well as the data with noisy labels.

As the provided train portion with noisy labels primarily contains audio signals of length 15 s and our model processes whole sequences, there is a bias towards sequences with a length of 15 s. To overcome this bias, we generate, before starting training, new audio excerpts of varying lengths by randomly splitting each of the audio signals with noisy labels into two at length $r_m \cdot T_m$ with T_m being the length of the m-th signal and $r_m \sim \mathcal{U}(0.1, 0.9)$. This results in audio excerpt lengths which are approximately uniformly distributed between 1 s and 13.5 s. As mixup data augmentation may lengthen audio signals, mixup of the noisy excerpts yields signals distributed between 1 s and 27 s. Each audio excerpt copies the event tags of the original audio, which results in some additional label noise. The random splitting is performed three times resulting in three different datasets which we refer to as splits 0-2 in the following[3].

We mixup curated only data which we refer to as the curated portion in the following as well as we mixup combined curated and noisy data which we refer to as the noisy portion in the following. In each training epoch the curated portion is repeated an integer number of times to prevent training from being dominated by noisy labels. The ratio of noisy labels to the total number of labels in one epoch is referred to as $R = \frac{M_\text{noisy}}{M_\text{total}}$.

5.2. Optimization

The training criterion is the binary cross entropy between the model predictions $\hat{\mathbf{y}}$ and the n-hot target vector $\tilde{\mathbf{y}}$:

$$L(\hat{\mathbf{y}}, \tilde{\mathbf{y}}) = -\sum_{k=0}^{K-1} \Big(\tilde{y}_k \log(\hat{y}_k) + (1-\tilde{y}_k)\log(1-\hat{y}_k)\Big)$$

with $K = 80$ denoting the number of target event classes.

We randomly sample mini batches of size 16 from the training data such that

1. no signal in the mini batch is padded by more than 20%,
2. no example in the mini batch includes the same events as another example in the same minibatch,

and compute gradients of the average loss in the mini batch. We clip gradients at a threshold of 15. Adam [15] was used for optimization. Training is performed for 200K iterations with a learning rate of $3 \cdot 10^{-4}$.

We perform Stochastic Weight Averaging (SWA) [16] for the last 50K iterations with a frequency of 1K iterations. At the end of training we exchange the model weights for the averaged weights. Finally we update the statistics of all batch normalization layers by making a forward pass on our (unaugmented) training data using the SWA model. SWA has shown to improve generalization and hence performance on unseen data [16]. Another advantage is that with SWA there is no need for held-out data to determine the best performing checkpoint.

6. EXPERIMENTS

In the following we evaluate the usefulness of the proposed data augmentation techniques and different methods for label noise handling. While Section 6.1 and Section 6.2 only report the performance of single model systems (which were trained on split 0),

[3]The generated splits are available at `https://github.com/fgnt/upb_audio_tagging_2019`

Section 6.3 presents performance of ensembles combining models trained on different splits.

System performance is measured in terms of label-weighted label-ranking average precision (lwlrap) evaluated on the private test set. The lwlrap metric measures the capablity of the model to rank active events over non-active events (see [10] for details). The reported scores are one-shot results and were obtained from (late) submissions on the Kaggle competition website.

6.1. Data Augmentation

To evaluate the different data augmentation methods we trained single model systems using curated and noisy labeled data using the provided labels. In this section, all models were trained with a noisy label rate of R=0.56. Table 2 shows the performance gains due to adding the proposed data augmentation methods. It can be seen that each augmentation method significantly increases the model performance in terms of lwlrap on the private test set.

Table 2: Performance when gradually adding data augmentation methods.

Data Augmentation	lwlrap
-	0.611
Mixup	0.665
+ Freq. Warp.	0.701
+ Freq. & Time Mask.	0.721

6.2. Label Noise Handling

In this Section we evaluate different methods for dealing with label noise when using all proposed data augmentation techniques. While for our challenge submission we used relabeling, which is explained first, we adopted Multi Task Learning [17] from the winning team after the challenge has ended, which is explained second.

6.2.1. Relabeling

For each of the three splits (as explained in Section 5.1) we trained models on five different folds using a noisy label rate of $R = 0.5$. This results in a total of 15 different models which were all combined into an ensemble to make predictions for the noisy excerpts from all three splits. Here the event scores of the individual models were averaged to obtain the ensemble output scores.

For each event we then determined the decision threshold yielding the best error rate jointly evaluated on the set of all noisy excerpts from all splits. These decision thresholds were used to relabel the noisy excerpts, where excerpts without any active event were discarded.

We then used the whole relabeled data of a split for training a new model as we do not need held-out data to determine the best checkpoint due to SWA.

6.2.2. Multi Task Learning

The winning team of the DCASE 2019 Challenge Task 2 proposed Multi Task Learning (MTL) to deal with noisy labels [17]. Here different classifier layers are used during training to predict curated and noisy labels, respectively. After the challenge ended we adopted this approach for our model, i.e., we trained different fully connected layers (the last two layers in Table 1) for curated and noisy labels. At test time only the classifier layers trained on the curated labels were used to make predictions.

6.2.3. Results

Results for the different methods of label noise handling are shown in Table 3. It can be seen that using the noisy labels results in a significant performance gain. However, using relabeling and MTL yield only small improvement of the lwlrap. In the next section these methods are further evaluated when performing ensembling.

Table 3: Performance for different treatments of label noise.

Method	R	lwlrap
Curated only	0.00	0.687
Provided labels	0.56	0.721
Relabeled	0.62	0.722
MTL	0.56	0.724

6.3. Ensembling

Finally, we evaluate ensembles of different models. In particular, we combine models trained on different splits and with different noisy label rates. System combination is achieved by averaging the output scores of the contributing systems. Each row in Table 4-6 adds three models which were trained on the three different splits. The results marked by an asterisk in Table 5 represent our submissions to the challenge. It can be seen that neither relabeling nor multi-task learning individually improve performance over the provided labels. Combining all the models from Tables 4-6 into a large ensemble of 27 models, however, raises performance to 76.5% lwlrap, setting a new state-of-the-art for this task as shown in Table 7. Do note that this ensemble has a total excecution time < $2\,\mathrm{h}$ using CPU, which meets the challenge constraints.

Table 4: Provided labels

#models	R	lwlrap
3	0.56	-
+3	0.64	-
+3	0.75	0.759

Table 5: Relabeled

#models	R	lwlrap
3	0.62	0.746*
+3	0.77	0.755*
+3	0.53	0.757

Table 6: MTL

#models	R	lwlrap
3	0.56	-
+3	0.64	-
+3	0.75	0.759

Table 7: System Comparison

System	lwlrap
Baseline [10]	0.537
DCASE winner [17]	0.758
Kaggle winner[4]	0.760
Our best subm. [18]	0.755
Our best late subm.	**0.765**

7. CONCLUSIONS

In this paper we presented our system for the DCASE 2019 Challenge Task 2. Our experiments carried out on the "FSDKaggle2019" data highlight the importance of data augmentation techniques for achieving high classification performance. On the other hand, particular consideration of the label noise did not prove effective in our case. Furthermore, performance was boosted by system combination raising the performance from 0.724 lwlrap of our best single model system to 0.765 by combining a total of 27 models, setting a new state of the art for this task.

[4]`https://www.kaggle.com/c/freesound-audio-tagging-2019/leaderboard`

8. REFERENCES

[1] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational analysis of sound scenes and events*. Springer, 2018.

[2] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[3] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," *arXiv preprint arXiv:1807.10501*, 2018.

[4] N. Turpault, R. Serizel, A. Parag Shah, and J. Salamon, "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis," June 2019, working paper or preprint. [Online]. Available: https://hal.inria.fr/hal-02160855

[5] E. Fonseca, M. Plakal, D. P. Ellis, F. Font, X. Favory, and X. Serra, "Learning sound event classifiers from web audio with noisy labels," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 21–25.

[6] C. Kim, A. Misra, K. Chin, T. Hughes, A. Narayanan, T. Sainath, and M. Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home," in *Proc. INTERSPEECH*, 2017, pp. 379–383.

[7] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[8] T. Inoue, P. Vinayavekhin, S. Wang, D. Wood, N. Greco, and R. Tachibana, "Domestic activities classification based on cnn using shuffling and mixing data augmentation," *DCASE 2018 Challenge*, 2018.

[9] I.-Y. Jeong and H. Lim, "Audio tagging system using densely connected convolutional networks," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018.

[10] E. Fonseca, M. Plakal, F. Font, D. P. Ellis, and X. Serra, "Audio tagging with noisy labels and minimal supervision," *arXiv preprint arXiv:1906.02975*, 2019.

[11] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, X. Favory, J. Pons, and X. Serra, "General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline," *CoRR*, vol. abs/1807.09902, 2018. [Online]. Available: http://arxiv.org/abs/1807.09902

[12] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[13] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[14] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in *Proc. ICML Workshop on Deep Learning for Audio, Speech and Language*, vol. 117, 2013.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[16] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson, "Averaging weights leads to wider optima and better generalization," *arXiv preprint arXiv:1803.05407*, 2018.

[17] O. Akiyama and J. Sato, "Multitask learning and semi-supervised learning with noisy data for audio tagging," DCASE2019 Challenge, Tech. Rep., June 2019.

[18] J. Ebbers and R. Haeb-Umbach, "Convolutional recurrent neural network and data augmentation for audio tagging with noisy labels and minimal supervision," DCASE2019 Challenge, Tech. Rep., June 2019.

AUDIO TAGGING WITH NOISY LABELS AND MINIMAL SUPERVISION

Eduardo Fonseca[1*†], *Manoj Plakal*[2†], *Frederic Font*[1], *Daniel P. W. Ellis*[2], *Xavier Serra*[1]

[1]Music Technology Group, Universitat Pompeu Fabra, Barcelona {name.surname}@upf.edu
[2] Google, Inc., New York, NY, USA {plakal,dpwe}@google.com

ABSTRACT

This paper introduces Task 2 of the DCASE2019 Challenge, titled "Audio tagging with noisy labels and minimal supervision". This task was hosted on the Kaggle platform as "Freesound Audio Tagging 2019". The task evaluates systems for multi-label audio tagging using a large set of noisy-labeled data, and a much smaller set of manually-labeled data, under a large vocabulary setting of 80 everyday sound classes. In addition, the proposed dataset poses an acoustic mismatch problem between the noisy train set and the test set due to the fact that they come from different web audio sources. This can correspond to a realistic scenario given the difficulty of gathering large amounts of manually labeled data. We present the task setup, the FSDKaggle2019 dataset prepared for this scientific evaluation, and a baseline system consisting of a convolutional neural network. All these resources are freely available.

***Index Terms*—** Audio tagging, sound event classification, audio dataset, label noise, minimal supervision

1. INTRODUCTION

Environmental sound recognition has gained attention in recent years, encompassing tasks such as acoustic scene classification, sound event detection or audio tagging [1]. The latter is becoming a popular task partly due to the various audio tagging tasks in the DCASE Challenge editions, and its impact on applications such as automatic description of multimedia, or acoustic monitoring. This paper describes the characteristics, dataset and baseline system of DCASE2019 Task 2 "Audio tagging with noisy labels and minimal supervision".

Everyday sound tagging consists of identifying the sound events present in an audio recording. The most common approach to create sound event taggers relies on supervised learning through labeled audio datasets. New released datasets tend to be of increasing size in order to allow exploitation of data-driven approaches, e.g., deep learning. However, manual labeling of large datasets is expensive and typically a limiting factor in machine listening; hence, creators are often forced to compromise between dataset size and label quality. Thus, most recent datasets feature larger sizes [2, 3, 4], but their labeling is less precise than that of conventional small and exhaustively labeled datasets [5, 6, 7]. We are, therefore, witnessing a transition towards larger datasets that inevitably include some degree of label noise. Likewise, the current trend is moving towards general-purpose sound event recognizers, able to recognize a broad range of everyday sounds. This is favoured by the appearance of the AudioSet Ontology; a hierarchical tree with 627 classes encompassing the most common everyday sounds [2]. This implies that not only are we interested in recognizing typical sound sources (e.g., *Bark*), but also less usual sound classes (e.g., production mechanisms such as *Fill (with liquid)*). Further, not only are these classes less frequent, but also some can be semantically or acoustically similar to others (e.g. *Trickle, dribble* and *Fill (with liquid)*). Manual annotation of these more ambiguous categories becomes more difficult, which makes them more prone to labeling errors. An alternative to gathering data for training general-purpose audio taggers is to retrieve audio and metadata from websites such as Freesound or Flickr. Labels can be inferred automatically by using automated heuristics applied to the metadata, or applying pre-trained classifiers on the audio material. This approach supports rapid collection of large amounts of data, but at the cost of a considerable amount of label noise.

In this context, label noise arises as a challenge in general-purpose sound event recognition, including adverse effects such as performance drop or increased complexity of models [8], and also hindering proper learning of deep networks [9, 10]. Consequently, coping with label noise could open the door to better sound event classifiers, and could allow the exploitation of large amounts of web audio for training, while reducing manual annotation needs. The topic of *learning with noisy labels* is a consolidated research area in computer vision [11, 12, 13, 14, 15, 16, 17]. However, in sound recognition it has received little attention, probably due to the conventional paradigm of learning from small and clean datasets; only a few works directly address the analysis and mitigation of label noise [4, 18, 19].

In this paper, we propose a task, a dataset and a baseline system to foster label noise research in general-purpose sound event tagging. We follow up on DCASE2018 Task 2 [3], and propose to investigate the scenario where a small set of manually-labeled data is available, along with a larger set of noisy-labeled data, in a multi-label audio tagging setting, and using a vocabulary of 80 classes of everyday sounds. The proposed task addresses two main research problems. The first problem is how to adequately exploit a large quantity of noisy labels, many of which are incorrect and/or incomplete, and how to complement it with the supervision provided by a much smaller amount of reliable manually-labeled data (*minimal* supervision). The second problem is given by the acoustic mismatch between the noisy train set and the test set. Distribution shifts between data have been shown to cause substantial performance drops in machine learning, both for vision [20] and audio [21]. In our case, the noisy train set comes from a different web audio source than the test set, which is sometimes a real-world constraint. This paper is organized as follows. Section 2 provides more details about the task and its experimental setup. Section 3 presents the dataset prepared for the task, and Section 4 describes a baseline system. Final remarks are given in Section 5.

*This work is partially supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 688382 AudioCommons and a Google Faculty Research Award 2018.

†Equal contribution.

https://doi.org/10.33682/w13e-5v06

2. TASK SETUP

The goal of this task is to predict appropriate labels for each audio clip in a test set. The predictions are to be done at the clip level, i.e., no start/end timestamps for the sound events are required. Some test clips bear one ground truth label while others bear several labels. Hence, the task setup is a *multi-label* classification problem and the systems to be developed can be denoted as multi-label audio tagging systems, as illustrated in Fig. 1. This task was hosted on the Kaggle platform from April 4th to June 10th 2019. The resources associated to this task (dataset download, submission, and leaderboard) can be found on the Kaggle competition page.[1]

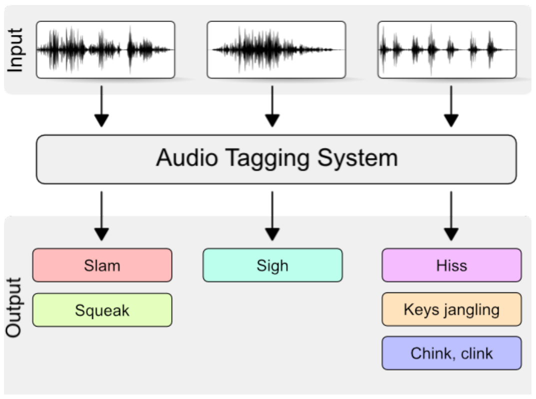


Figure 1: Overview of a multi-label tagging system.

As described in Section 3, the audio data for this task consists of a test set and two train sets: a *curated* train set and a *noisy* train set, that allow to experiment with training data of different levels of reliability and coming from different sources. System evaluation was carried out on Kaggle servers (see Section 2.1) using the test set, which is further split into two divisions, for the *public* and *private* leaderboards. During the competition, the test subset corresponding to the public leaderboard was used to provide live ranking of all participants. To compute the final private leaderboard, at the end of the competition, systems were re-evaluated using the unseen private test set, of which neither the audio nor the labels were accessible to participants.

2.1. Evaluation Metric and Competition Rules

The task was evaluated via label-weighted label-ranking average precision (abbreviated as *lωlrap* and pronounced "lol wrap"). This generalizes the mean reciprocal rank (MRR) used in the 2018 challenge [3] to the case of multiple true labels per test item. Let $Lab(s,r)$ be the class label at rank r (starting from 1) in test sample s, and $Rank(s,c)$ be the rank of class label c in that list, i.e. $Lab(s, Rank(s,c)) = c$. Then, if the set of ground-truth classes for sample s is $C(s)$, the label-ranking precision for the list of labels up to class c (assumed to be in $C(s)$) is:

$$Prec(s,c) = \frac{1}{Rank(s,c)} \sum_{r=1}^{Rank(s,c)} \mathbf{1}[Lab(s,r) \in C(s)] \quad (1)$$

where $\mathbf{1}[\cdot]$ evaluates to 1 if the argument is true, else zero. $Prec(s,c)$ is equal to 1 if all the top-ranked labels down to c are part of $C(s)$, and at worst case equals $1/Rank(s,c)$ if none of the higher-ranked labels are correct. In contrast to plain *lrap*, which averages precisions within a sample then across samples, thereby downweighting labels that occur on samples with many labels, *lωlrap* calculates the precision for each *label* in the test set, and gives them all equal contribution to the final metric:

$$l\omega lrap = \frac{1}{\sum_s |C(s)|} \sum_s \sum_{c \in C(s)} Prec(s,c) \quad (2)$$

where $|C(s)|$ is the number of true class labels for sample s. We use label weighting because it allows per-class values to be calculated, while keeping the overall metric as a simple average of the per-class metrics (weighted by each label's prior in the test set). A Python implementation of *lωlrap* is provided in [2].

This scientific evaluation was set up as a Kaggle *Kernels-only* competition. This means that all participants had to submit their systems as inference models in Kaggle Kernels (similar to *Jupyter Notebooks*), to be evaluated on remote servers. In addition, inference run-time was limited to a maximum of one hour in a Kernel with one GPU, and memory constraints were also imposed. These constraints aim to discourage the usage of large model ensembles. Participants could submit a maximum of two submissions per day, and select two *final* submissions to be considered for the private leaderboard ranking. A detailed description of the task rules can be found in the Rules section of the competition page;[1] the most important points are summarized in the DCASE Challenge page.[3]

To complement the leaderboard results of the *lωlrap* ranking, the task organizers introduced a complementary Judges' Award to promote submissions using novel, problem-specific and efficient approaches. Details about the Judges' Award rules can be found in the Judges' Award section of [1].

3. DATASET

The dataset used is called *FSDKaggle2019*, and it employs audio clips from the following sources:

- Freesound Dataset (FSD): a dataset under development based on Freesound content organized with the AudioSet Ontology [2]. Freesound is a sound sharing site hosting over 400,000 clips uploaded by a community of users, who additionally provide some basic metadata, e.g., tags. [22]. These data are used to create the curated train set and the test set.
- The soundtracks of a pool of Flickr videos taken from the Yahoo Flickr Creative Commons 100M (YFCC100M) dataset [23]. These data are used to create the noisy train set.

FSDKaggle2019 is freely available from the Data section of [1], all clips are provided as uncompressed PCM 16 bit 44.1 kHz mono audio files, its ground truth labels are provided at the clip-level (i.e., weak labels), and its partitioning is depicted in Fig. 2.

3.1. Curated train set and test set

The first step carried out in the creation of FSDKaggle2019 was the definition of a vocabulary of 80 classes drawn from the AudioSet

[1]`https://www.kaggle.com/c/freesound-audio-tagging-2019`
Note that the competition name on Kaggle is abbreviated from the full DCASE2019 Challenge task name to "Freesound Audio Tagging 2019".

[2]`https://colab.research.google.com/drive/1AgPdhSp7ttY18O3fEoHOQKlt_3HJDLi8`

[3]`http://dcase.community/challenge2019/task-audio-tagging#task-rules`

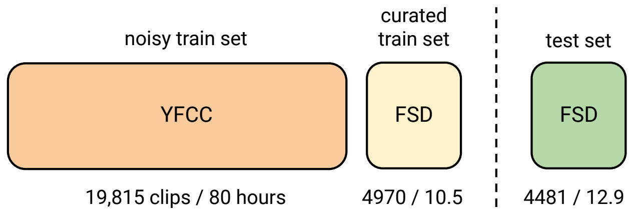


Figure 2: Data split in FSDKaggle2019, including number of clips / duration in hours, and data origin. Colors depict quality of labels: orange, yellow and green correspond to noisy labels, correct but potentially incomplete labels, and exhaustive labels, respectively.

Ontology [2]. This vocabulary was chosen based on the following criteria: *i)* we consider leaf nodes of the AudioSet hierarchy for which there is enough data available in FSD, *ii)* we aim to encompass a diverse range of everyday sounds, and *iii)* we remove few clearly isolated classes (those with the weakest semantic relations with any of the rest), thus promoting confounds between semantically/acoustically similar classes to some extent. The main sound *families* (i.e., groups of sound classes) in the resulting vocabulary are, in descending order of prevalence, human sounds, domestic sounds, musical instruments, vehicles, animal sounds, natural sounds, materials, and mechanisms. The full list of 80 classes is available in the Data section of [1].

In a second step, we did a mapping of Freesound clips to the selected 80 class labels. To this end, a set of keywords was defined connecting the user-provided Freesound tags with the AudioSet labels. Using this mapping, for every class, we retrieved the audio clips that feature at least one of the defined keywords among their tags. This process led to a number of automatically-generated *candidate annotations* indicating the potential presence of a sound class in an audio clip (i.e., weak labels, as timing information is not included). Nonetheless, in some audio clips the target signal fills the clip length almost completely, which can be considered as a strong label. Subsequently, the candidate annotations were human-validated using a *validation* task deployed in *Freesound Annotator*,[4] an online platform for the collaborative creation of open audio datasets [24]. In this task, users verify the presence/absence of a candidate sound class in an audio clip with a *rating* mechanism. The vast majority of provided labels have inter-annotator agreement but not all of them. The outcome is a set of clips where the corresponding label(s) are correct; nevertheless, it can happen that a few of these audio clips present additional acoustic material beyond the provided label(s).

The resulting data were split into a train set and a test set. We refer to this train set as *curated* in order to distinguish it from the noisy set described in Section 3.2. To mitigate train-test contamination, the split was carried out considering the clip uploaders in Freesound. We allocated all audio clips uploaded from the same user into either the curated train set or the test set, so that the sets are disjoint at the Freesound user level. The partition proportion was defined to limit the supervision provided in the curated train set, thus promoting approaches to deal with label noise.

Finally, the test set was further annotated using a label *generation* tool [25], in which *i)* pre-existent labels can be re-validated, and *ii)* potentially missing labels can be added through exploration of the AudioSet Ontology. The outcome is a set of exhaustively labeled clips where the label(s) are correct and complete considering the target vocabulary; nonetheless, few clips could still present additional (unlabeled) acoustic content out of the vocabulary.

[4]https://annotator.freesound.org

Table 1: Main stats of the sets in FSDKaggle2019. *A few classes have slightly less than 75 clips.

Aspect	**curated train**	**noisy train**	**test**
Clips/class	∼75*	300	∼ 50 - 150
Total clips	4970	19,815	4481
Labels/clip	1.2	1.2	1.4
Clip length	∼0.3 - 30s	∼15s	∼0.3 - 30s
Total duration	∼10.5h	∼80h	∼12.9h
Labelling	correct (inexhaustive)	noisy	exhaustive

The main characteristics of the curated train set, noisy train set and test set are listed in Table 1. The curated train set consists of 4970 clips with a total of 5752 labels. Labels per clip ranges from 1 to 6 with a mean of 1.2. The test set consists of 4481 clips with a total of 6250 labels. Labels per clip ranges from 1 to 6 with a mean of 1.4. Note the increased number of labels per clip with respect to the curated train set, due to the process of exhaustive labelling. In both cases, clip length ranges from 0.3s to 30 due to the diversity of the sound classes and the preferences of Freesound users when recording/uploading sounds.

3.2. Noisy train set

The noisy train set was prepared using the YFCC100M dataset [23], which has the advantages of *i)* being a very large and diverse dataset that is not correlated with Freesound in acoustics or domain, and *ii)* offering permissive Creative Commons licenses that allow ease of use, modification, and redistribution. The original dataset contained ∼99M photos and ∼793k videos from ∼581K Flickr users. We dropped videos with licenses that disallowed making derivatives or commercial use, videos that were no longer available, and videos with audio decode errors that we could not transcode, leaving us with ∼201K 44.1 kHz mono WAV files. Video length varied with a maximum of 20 minutes, and a mean of ∼37s and median of ∼20s.

The Flickr video metadata (title, description, tags) proved to be too sparse to meaningfully map to our class vocabulary. Therefore, we used a content-based approach where we generated video-level predictions from a variety of pre-trained audio models: a shallow fully-connected network as well as variants of VGG and ResNet [26], all of which were trained on a large collection of YouTube videos using the AudioSet class vocabulary. We generated sliding windows of ∼1s containing log mel spectrogram patches and aggregated the per-window predictions (using either maximum or average pooling) to produce a video-level vector of class scores. For each of our 80 classes, we kept the top 300 videos by predicted score for that class. We browsed the video labels and selected the maximum-pooled VGG-like model as producing a balance between reasonable predictions and a substantial amount of label noise. As a further source of noise, each final clip was produced by taking a random slice of a video of length up to 15 seconds (videos shorter than 15 seconds would be taken in their entirety). Hence, the label noise can vary widely in amount and type depending on the class, including in- and out-of-vocabulary noises [4].

As listed in Table 1, the noisy train set consists of 19,815 clips with a total of 24,000 labels (300 * 80). Labels per clip ranges from 1 to 7 with a mean of 1.2. Clip length ranges from 1s to

15s (by construction), with a mean of 14.5s. Therefore, the per-class training data distribution in FSDKaggle2019 is, for most of the classes, 300 clips from the noisy set and 75 clips from the curated set. This means 80% noisy / 20% curated at the clip level, while at the duration level the proportion is more extreme considering the variable-length clips. Since most of the train data come from YFCC, acoustic domain mismatch between the train and test set can be expected. We conjecture this mismatch comes from a variety of reasons. For example, through acoustic inspection of a small sample of both data sources, we find a higher percentage of high quality recordings in Freesound. In addition, audio clips in Freesound are typically recorded with the purpose of capturing audio, which is not necessarily the case in YFCC.

4. BASELINE SYSTEM

4.1. Model Architecture and Training

The baseline system uses a convolutional neural network that takes log mel spectrogram patches as input and produces predicted scores for 80 classes. We use an efficient MobileNet v1 [27] architecture which lets us fit comfortably within the inference time limits of the challenge. Incoming audio (always 44.1 kHz mono) is divided into overlapping windows of size 1s with a hop of 0.5s. These windows are decomposed with a short-time Fourier transform using 25ms windows every 10ms. The resulting spectrogram is mapped into 96 mel-spaced frequency bins covering 20 Hz to 20 kHz, and the magnitude of each bin is log-transformed after adding a small offset to avoid numerical issues. The model consists of 1 convolutional layer followed by 13 separable convolution layers (which give MobileNets their compactness) followed by either max or average pooling, and an 80-way logistic classifier layer. The model contains ∼3.3M weights and, by comparison, is ∼8x smaller than a ResNet-50 while using ∼4x less compute. Detailed documentation is available in the public release of the baseline system code.[5]

To combat label noise, we use dropout (inserted before the classifier layer) as well as label smoothing [28] which replaces each label's target value with a hyperparameter-controlled blend of the original value and 0.5 (representing a uniform probability distribution). To combat the domain mismatch between the test set and noisy train set, we use transfer learning by training a model on the noisy set first to learn a representation, and then use a checkpoint from that run to warm-start training on the curated train set. In addition, we used batch normalization, exponential learning rate decay, and the Adam optimizer. We trained models on the curated data alone, the noisy data alone, the curated and noisy combined, noisy first followed by warm-started curated, as well as a weighted version of warm-started training that we describe next.

4.2. Results

Table 2 shows the *lωlrap* values produced by our various baseline models when evaluated on the entire test set (i.e., including both the public and private splits). Each row lists the best *lωlrap* obtained from a small grid search (using the public test set for evaluation) that varied maximum vs average pooling, learning rate, label smoothing, dropout, learning rate decay, and whether or not we used batch normalization. The baseline system that we settled on was "Warm-started curated", which achieved a *lωlrap* of 0.537 on the public test set (see the publicly released baseline code for hyperparameter choices).

Table 2: Baseline system results on the *entire* test set.

Training approach	***lωlrap***
Curated only	0.542
Noisy only	0.312
Curated + Noisy	0.522
Warm-started curated	**0.546**
Weighted warm-started curated	0.561

Comparing "Noisy only" and "Curated + Noisy" to "Curated only" shows a considerable domain mismatch where we hurt our performance when we blindly add more data. A transfer learning approach of warm-starting the curated training with a noisily trained model gives us a small boost in performance.

We conducted a class-based analysis of the results in a public Colab notebook[6] where we look at the best and worst classes of each model. The baseline system attains highest *lωlrap* values for *Bicycle bell* (0.894) and *Purr* (0.873); lowest values occur for *Cupboard open or close* (0.219) and *Chirp, tweet* (0.127).

We also analyse the correlations between various pairs of models. It becomes evident that, at least for our baseline models, there are classes where using curated data alone is better while there are other classes where the noisy data is better. One simple way to incorporate this in training is to use the ratio of noisy to curated *lωlrap*s as a per-class weight during noisy training to boost classes that have value and suppress classes that do not. When we warm-start with this weighted noisy model, we get a further boost in performance. This optimization is not included in the released baseline.

5. CONCLUSION

In this paper, we have described the task setup, dataset, and baseline system of DCASE2019 Task 2 "Audio tagging with noisy labels and minimal supervision". This task was hosted on the Kaggle platform as "Freesound Audio Tagging 2019". The goal is to adequately exploit a large set of noisy-labeled data and a small quantity of manually-labeled data, in a multi-label audio tagging setting with a vocabulary of 80 everyday sound classes. In addition, the dataset poses an acoustic mismatch problem between the noisy train set and the test set due to the fact that they come from different web audio sources. We believe this can correspond to a realistic scenario given the difficulty in gathering large amounts of manually labeled data. *lωlrap* is proposed as evaluation metric. Baseline experiments indicate that leveraging noisy-labeled data with a distribution shift for sound event tagging can be challenging. The FSDKaggle2019 dataset and the baseline system proposed are freely available and not limited for use within the competition.

6. ACKNOWLEDGMENT

We thank Addison Howard and Walter Reade of Kaggle for their invaluable assistance with the task design and Kaggle platform, and everyone who contributed to FSDKaggle2019 with annotations, especially Jordi Pons and Xavier Favory. Eduardo Fonseca is also grateful for the GPU donated by NVidia.

[5]https://github.com/DCASE-REPO/dcase2019_task2_baseline

[6]https://colab.research.google.com/drive/1vIpAWCGSHyYYLblbtMzAKLNTkHcr1SMK

7. REFERENCES

[1] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational Analysis of Sound Scenes and Events*. Springer, 2018.

[2] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[3] E. Fonseca, M. Plakal, F. Font, D. P. Ellis, X. Favory, J. Pons, and X. Serra, "General-purpose tagging of Freesound audio with AudioSet labels: task description, dataset, and baseline," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018.

[4] E. Fonseca, M. Plakal, D. P. W. Ellis, F. Font, X. Favory, and X. Serra, "Learning Sound Event Classifiers from Web Audio with Noisy Labels," in *Proc. IEEE ICASSP 2019*, Brighton, UK, 2019.

[5] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2014, pp. 1041–1044.

[6] K. J. Piczak, "ESC: Dataset for environmental sound classification," in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2015, pp. 1015–1018.

[7] P. Foster, S. Sigtia, S. Krstulovic, J. Barker, and M. D. Plumbley, "CHiME-home: A dataset for sound source recognition in a domestic environment," in *Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, 2015.

[8] B. Frénay and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 5, 2014.

[9] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, *et al.*, "A closer look at memorization in deep networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 233–242.

[10] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[11] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Advances in Neural Information Processing Systems*, 2018, pp. 8527–8537.

[12] E. Malach and S. Shalev-Shwartz, "Decoupling" when to update" from "how to update"," in *Advances in Neural Information Processing Systems*, 2017, pp. 960–970.

[13] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," *arXiv preprint arXiv:1712.05055*, 2017.

[14] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," in *Proceedings of the International Conference on Learning Representations*, 2017.

[15] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. J. Belongie, "Learning from noisy large-scale datasets with minimal supervision." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6575–6583.

[16] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in Neural Information Processing Systems*, 2018.

[17] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1944–1952.

[18] A. Kumar, A. Shah, A. Hauptmann, and B. Raj, "Learning sound events from webly labeled data," *arXiv preprint arXiv:1811.09967*, 2018.

[19] E. Fonseca, F. Font, and X. Serra, "Model-agnostic approaches to handling noisy labels when training sound event classifiers," in *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New York, US, 2019.

[20] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do ImageNet Classifiers Generalize to ImageNet?" *arXiv preprint arXiv:1902.10811*, 2019.

[21] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification: an overview of DCASE 2017 challenge entries," in *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*. IEEE, 2018, pp. 411–415.

[22] F. Font, G. Roma, and X. Serra, "Freesound technical demo," in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2013, pp. 411–412.

[23] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li, "YFCC100M: The New Data in Multimedia Research," *Communications of the ACM*, vol. 59, no. 2, pp. 64–73, Feb. 2016.

[24] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound Datasets: a platform for the creation of open audio datasets," in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, Suzhou, China, 2017, pp. 486–493.

[25] X. Favory, E. Fonseca, F. Font, and X. Serra, "Facilitating the manual annotation of sounds when using large taxonomies," in *Proceedings of the 23rd Conference of Open Innovations Association FRUCT*. FRUCT Oy, 2018, p. 60.

[26] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "CNN Architectures for Large-Scale Audio Classification," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[27] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv preprint arXiv:1704.04861*, 2017.

[28] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition,*, 2016. [Online]. Available: http://arxiv.org/abs/1512.00567

ROBUST NON-NEGATIVE BLOCK SPARSE CODING FOR ACOUSTIC NOVELTY DETECTION

Ritwik Giri, Arvindh Krishnaswamy, and Karim Helwani

Amazon Web Services, Inc., Palo Alto, CA
{ritwikg, arvindhk, helwk}@amazon.com

ABSTRACT

In this paper we address the problem of detecting previously unseen novel audio events in the presence of real-life acoustic backgrounds. Specifically, during training, we learn subspaces corresponding to each acoustic background, and during testing the audio frame in question is decomposed into a component that lies on the mixture of subspaces and a supergaussian outlier component. Based on the energy in the estimated outlier component a decision is made, whether or not the current frame is an acoustic novelty. We compare our proposed method with state of the art auto-encoder based approaches and also with a traditional supervised Nonnegative Matrix Factorization (NMF) based method using a publicly available dataset - A3Novelty. We also present results using our own dataset created by mixing novel/rare sounds such as gunshots, glass-breaking and sirens, with normal background sounds for various event to background ratios (in dB).

1. INTRODUCTION

1.1. Background

Novel audio event detection has a number of important applications. For example, in surveillance systems, detecting unusual events using audio can nicely complement video based approaches. This is especially true in cases where there is not sufficient illumination, or in the presence of visual occlusions where the performance of video surveillance is impaired. But novelty detection poses interesting challenges as well.

One difficulty is that not all potential audio events can be pre-determined, pre-recorded and labeled. We need to deal with unseen novel sounds and transients. But most of the current literature we have seen on audio surveillance systems [1, 2, 3] propose fully supervised learning methods: along with acoustic background data they also require labeled audio examples corresponding to audio events. We have also seen applications of this supervised audio event detection in consumer products, for example Alexa Guard [1]: this feature enables Echo devices to detect specific sounds that the user selects such as smoke alarm, glass breaking sound, carbon monoxide alarms etc. But these fully-supervised systems can not detect unseen novel audio events. Therefore, researchers are working on unsupervised techniques as well to detect novel audio events [4, 5, 6, 7, 8].

1.2. Related Work

Even though novelty detection is a relatively new problem in the audio signal processing community, this topic has been well-researched in other data modalities and fields such as medical diagnosis [9, 10], damage inspection [11, 12] electronic IT security [13], and video surveillance systems [14]. In [15, 16] authors grouped several novelty detection techniques in two major categories - statistical approaches and neural network based approaches.

Statistical approaches depend on properties of the normal background audio data, and, during training, either fit a model or a probability distribution function over the data. During testing they exploit this pre-trained model to determine if a test sample belongs to the learned distribution or not. These methods have been well researched and have been applied to several novelty detection applications successfully such as in handwriting detection, the recognition of cancer, failure detection in jet engines, and fMRI analysis. In the context of acoustic novelty detection, in [7], the authors have introduced Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM) based methods for detecting audio novelties in realistic acoustic backgrounds such as in 1) smart-home environments, 2) ATM settings, and 3) general-purpose security settings. In [4] authors proposed a one-class Support Vector Machine (OC-SVM) based unsupervised method for real-time detection of novel events in the context of audio surveillance. Recently in [17], the authors proposed a non-negative matrix under-approximation (NMU) method to perform novel-sound detection for unhealthy machineries.

Neural network based approaches, specifically autoencoder based approaches, have recently gained attention for novelty detection in both audio and in other modalities. The main working principle of these approaches lies in training an autoencoder using normal/expected data, and during testing checking if the network is struggling to encode and decode the test data accurately. I.e., if the system produces a high reconstruction error compared to some threshold, it is then considered novel input. In [18], the authors proposed a denoising autoencoder structure using both feedforward units and LSTM units for acoustic novelty detection task and showed significant improvement of performance over both GMM and OC-SVM based methods.

1.3. Contribution

In this article, we propose a robust non-negative block sparse coding based technique to detect novel sounds, such as gunshots, glass-breaking, sirens etc, in different real life acoustic backgrounds, such as in a bus, cafe, beach, city center, metro station and grocery store. During training, we learn subspaces corresponding to each acoustic background using supervised Nonnegative Matrix Factorization (S-NMF). During testing the audio frame in question is decomposed into a component that lies on the mixture of subspaces and a supergaussian outlier component. Because we

[1]https://www.cnet.com/news/alexa-guard-goes-live-lets-your-echo-listen-for-trouble-amazon-home-security/

https://doi.org/10.33682/pkcj-5s72

use a separate estimator for the outlier modeled as a supergaussian random variable, our approach is robust to minor deviations in learned acoustic backgrounds and also actual backgrounds during testing unlike in [17], where no such constraint has been imposed on novel sound estimate. We also create a challenging dataset by mixing novel sounds with real life acoustic scenes for different event to background ratios. Finally, we compare the results of our method, with state-of-the-art methods using a publicly available dataset: A3Novelty. We also compare them using our own dataset and show that our proposed method either matches the performance of the state-of-the-art methods or outperforms them in different cases.

The rest of the article is organized as follows: In Section 2 the proposed method is presented in detail, in Section 3 a brief description of both the datasets, that have been used in this article is given, in Section 4 we present evaluation results of our proposed method and other competing methods over previously mentioned two datasets and finally Section 5 concludes the paper and talks about some future research directions.

2. PROPOSED METHOD

2.1. Training Stage: Learning Background Subspaces

First, an offline training stage is needed to learn the corresponding dictionaries as subspaces, for D different types of acoustic backgrounds such as in a bus, in a cafe, in a city center etc., by solving the following optimization problem D times, for $d = 1...D$.

$$\mathbf{W}_{N_d}, \mathbf{H}_{N_d} = \arg \min_{\mathbf{W}\geq 0, \mathbf{H}\geq 0} \mathrm{KL}(\mathbf{N}_d|\mathbf{WH}) + \mu|\mathbf{H}|_1, \quad (1)$$

where $\mathbf{N}_d \in K(\text{number of bins}) \times L(\text{number of frames})$ is the magnitude of the d^{th} training background sound STFT representation, and $KL(\cdot|\cdot)$ denotes the KL divergence. This optimization problem can be solved in an iterative manner using multiplicative updates [19]. To avoid scaling indeterminacies, normalization constraint is applied, such that columns of $\mathbf{W}_{N_d}$ have unit norm. After the training stage, the mixture of subspaces corresponding to D different types of acoustic backgrounds can be represented by the concatenated dictionary matrix, $\mathbf{W} = [\mathbf{W}_{N_1}, ..., \mathbf{W}_{N_d}, ..., \mathbf{W}_{N_D}]$.

2.2. Testing Stage: Robust Non-negative Sparse Coding

2.2.1. Basic Model

During testing stage, we will decompose the i^{th} test frame STFT magnitude in the following manner,

$$\mathbf{v_i} = \mathbf{Wh_i} + \mathbf{r_i}, \quad (2)$$

where, $\mathbf{v_i} \in K(\text{number of bins}) \times 1$ is the magnitude of the i^{th} frame STFT representation, $\mathbf{r_i} \in K(\text{number of bins}) \times 1$ is the outlier term and $\mathbf{h_i} \in M(\text{number of bases in } \mathbf{W}) \times 1$ is the i^{th} activation vector.

2.2.2. Structured Sparsity in Activation

Since $\mathbf{W}$ is an overcomplete dictionary, a sparseness constraint over the activation vector $\mathbf{h_i}$ is typically employed, leading to a standard sparse coding framework. Since we are operating on STFT magnitude feature space, non-negativity constraint over $\mathbf{h_i}$ will also be employed. During testing, it is reasonable to assume that the acoustic background is not changing drastically, hence instead of solving the above mentioned sparse coding problem for every frame, we will solve for $L = 5$ frames simultaneously. Hence the model will become,

$$\mathbf{V} = \mathbf{WH} + \mathbf{R} \quad (3)$$

where $\mathbf{V}, \mathbf{H}, \mathbf{R}$ are matrices now with $L = 5$ columns/ frames.

For our problem in hand, a more structured sparsity constraint has been identified as useful. Since our acoustic background dictionary $\mathbf{W}$ is essentially a concatenated version of D subdictionaries, we argue that similar block structure can also be expected in each frame of the activation matrix, i.e., $\mathbf{H}_{(:,l)}$. Intuitively, this represents the fact that during testing if a basis vector of a specific subdictionary contributes to represent the testing frame, the other basis vectors of that specific subdictionary will also contribute. Hence in the l^{th} activation vector, a block sparsity structure can be expected. To impose this structural constraint, following regularization term is included in the cost function,

$$\Psi(\mathbf{H}) = \sum_{l=1,..5} \sum_{g_i \in G} \log(||\mathbf{H}_{(g_i,l)}||_1 + \epsilon) \quad (4)$$

where, $G = [g_1, ..., g_D]$ represents the D background subspaces. In literature, this regularization term is also known as the log-ℓ_1 measure [20]. We will also assume that since the acoustic background is not changing within 5 frames the same subdictionary will be used to explain the data over these 5 frames. Hence, structured regularizer over the activation matrix will become,

$$\Psi(\mathbf{H}) = \sum_{g_i \in G} \log(||vec(\mathbf{H}_{(g_i,:)})||_1 + \epsilon). \quad (5)$$

Where, $vec(\mathbf{H}_{(g_i,:)})$ is the vectorized format of submatrix $\mathbf{H}_{(g_i,:)}$. In our work, we keep the hyperparameter $\epsilon = 10^{-4}$ fixed for all our experiments.

2.2.3. Structured Sparsity in Outliers

As discussed above, we employ a supergaussianity/ sparse constraint on the outlier term $\mathbf{R}$, to capture the novel event. This intuition of using a supergaussian random variable to model outliers is motivated from several well known literatures on robust regression [21, 22, 23], which have used heavytailed/supergaussian distributions, such as student's t distribution, to model the outliers in the data.With the supergaussianity assumption, we make sure the that the model mismatch error (tends to be smaller) will not get absorbed in our outlier estimate. For the outlier matrix $\mathbf{R}$, we will again employ log-ℓ_1 regularizer with group sparsity constraint, but in this case each frame/ column of $\mathbf{R}$ will be a group. This can be interpreted as, if one frame is representing a novel event, all of the frequency bins of that frame will have the opportunity to be active. Hence, the regularizer on $\mathbf{R}$ is,

$$\Pi(\mathbf{R}) = \sum_{l=1,..,5} \log(||\mathbf{R}_{(:,l)}||_1 + \epsilon). \quad (6)$$

2.2.4. Derivation of Multiplicative Updates

After combining the model mismatch error, and the two regularizers on activation matrix and outlier matrix, resulting cost function that we will minimize is,

$$\hat{\mathbf{H}}, \hat{\mathbf{R}} = \arg \min_{\mathbf{H}\geq 0, \mathbf{R}\geq 0} ||\mathbf{V} - \mathbf{WH} - \mathbf{R}||_F^2 + \lambda\Psi(\mathbf{H}) + \mu\Pi(\mathbf{R}). \quad (7)$$

To solve the optimization problem with the non-negativity constraint on $\mathbf{H}$ and $\mathbf{R}$, we follow the block co-ordinate descent framework based multiplicative update rules proposed in [24] and derive the following update rules,

$$\mathbf{H}^{t+1} = \mathbf{H}^t \otimes \frac{\mathbf{W}^T\mathbf{V}}{\mathbf{W}^T(\mathbf{W}\mathbf{H}^t + \mathbf{R}^t) + \frac{\lambda}{\mathbf{S}^t + \epsilon}}, \tag{8}$$

where, $\mathbf{S}^t_{(g_i,l)} = ||vec(\mathbf{H}^t_{(g_i,:)})||_1$, for $l = 1, ..., 5$ number of frames.

$$\mathbf{R}^{t+1} = \mathbf{R}^t \otimes \frac{\mathbf{V}}{\mathbf{W}\mathbf{H}^{t+1} + \mathbf{R}^t + \frac{\mu}{\mathbf{P}^t + \epsilon}}, \tag{9}$$

where, $\mathbf{P}^t_{(k,l)} = ||\mathbf{R}^t_{(:,l)}||_1$, for $k = 1, ..., K$ and $l = 1, ..., 5$ number of frames.

These multiplicative updates are performed for 100 iterations, and the estimated value of the outlier matrix $\mathbf{R}$ is further used for an adaptive thresholding based decision function, discussed in the next subsection. We refer to our proposed approach as Robust Non-negative Block Sparse Coding (RNBSC) in later sections.

A similar robust NMF formulation has been considered before in [25], where the authors used a similar framework to learn robust subspaces for a face recognition task. But they did not consider any sparsity on the activation matrix $\mathbf{H}$ and furthermore, no structured sparsity constraint was considered on outlier matrix $\mathbf{R}$.

2.3. Adaptive Thresholding

For the adaptive thresholding step, we compute outlier estimate and model mismatch error estimate for each frame, i.e. for i^{th} frame, outlier estimate $r(i) = ||\mathbf{R}(:,i)||_2$ and model mismatch error, $e(i) = ||\mathbf{V}(:,i) - \mathbf{W}\mathbf{H}(:,i) - \mathbf{R}(:,i)||_2$. We also perform causal moving average of both e and r over previous 10 frames to smoothen the estimates.

Since the background level can vary a lot, instead of using a single value as the outlier threshold, we use an adaptive threshold concept, where $\theta = \beta \times \text{median}(e(1:N))$ is the threshold for each audio clip, where N is the total number of frames in each audio clip during evaluation. Finally, threshold θ is applied over time series of outlier estimate r, to obtain a binary signal, i.e. novelty or no novelty.

3. DATASETS

3.1. A3Novelty Database

The A3Novelty corpus[2] consists of 56 hours of recording and was recorded in a laboratory of the Univerita Politecnica delle Marche. These recordings were performed during both day and night time to account for different acoustic backgrounds. A variety of novel sounds, such as screams, falls, alarms, breakages of objects etc., were played back randomly using a loudspeaker during these recordings to generate backgrounds with novel sounds.

In the original A3Novelty database the audio recordings were segmented in sequences of 30 seconds. Authors of [8], randomly selected 300 sequences from the background material to construct the training set (150 minutes) and 180 sequences from background with novel sounds to compose the evaluation set (90 minutes). We have used the same database for our evaluation purposes which is publicly available[3].

[2]http://www.a3lab.dii.univpm.it/research/a3novelty

[3]http://a3lab.dii.univpm.it/webdav/audio/

3.2. Own Evaluation Database (OED)

Since the A3Novelty database was recorded indoors, it does not account for highly non-stationary acoustic backgrounds, hence, detecting impulsive novel sounds from a stationary acoustic background becomes comparatively easier. To tackle this issue, we created our own database by mixing novel audio events i.e. gunshots, glass breaking and sirens (obtained from publicly available resources[4]) with acoustic background audio recordings obtained from DCASE 2016 challenge [26], which has 6 different acoustic backgrounds: beach, bus, cafe, grocery store, city center and metro station. In our application, Event to Background ratio (EBR) is defined globally, i.e., the gain is computed from the average energy over the whole background (10 secs clip) and the event signal, to raise a global EBR. As discussed in [4], by using this approach we created signals which are good representative of real life audio events. We created these mixtures for different EBRs: 0, 5, 10, 15, 20 dB.

For each of the acoustic background (total: 6) we created 60 audio clips with 3 different type of novel sounds (gunshot, glass breaking, siren). Each clip is of length 10 s, resulting in total of 60 mins of test audio for each EBR. We also included 300 audio clips (each of 10 s) of just acoustic background. During mixing of the audio events, we generated the true labels where the resolution is 1 s, i.e., for each 10 s long clip we generate a 10 dimensional vector as true label. Along with the evaluation set, for training purposes we randomly selected segments of acoustic background recordings (disjoint of evaluation set) from the same DCASE challenge recordings, totaling to 90 mins of training data.

4. EXPERIMENTAL RESULTS

4.1. Competing Methods

In [18, 8], it has been shown that recently proposed Autoencoder (AE) based approaches perform significantly better than previously proposed statistical model based approaches such as GMM [7], HMM [7], and OC-SVM [4]. Hence, we choose to compare our proposed method with two AE based methods. We also compare against S-NMF based approach to illustrate the usefulness of robust outlier term in Equation 3. Since methods, that use any look ahead information will not be feasible for real time novelty detection application, we don't compare with structures with BLSTM units.

- **DAE-MLP:** Denoising AE with Feed Forward structure 257-512-257 and input is corrupted with Gaussian noise (std: 0.1).
- **DAE-LSTM:** Denoising AE with LSTM units in hidden layer (257-512-257) and input is corrupted with Gaussian noise (std: 0.1).
- **S-NMF:** Supervised NMF based approach, where the thresholding is done on mismatch error, i.e., $e(i) = ||\mathbf{V}(:,i) - \mathbf{W}\mathbf{H}(:,i)||_2$.
- **RNBSC:** Proposed approach.

For a fair comparison, same adaptive thresholding approach has been employed for all competing algorithms.

4.2. Setup

We use Short Time Fourier Transform (STFT) based time-frequency representation of the audio clips and operate on STFT

[4]https://freesound.org/

Table 1: Results over A3Novelty Dataset (1 sec)

Methods	Precision (%)	Recall (%)	F Score (%)
DAE-MLP	95.00	97.43	96.20
DAE-LSTM	97.49	**100**	**98.73**
S-NMF	97.22	89.74	93.33
RNBSC (Proposed)	**100**	97.43	98.70

Table 2: Results over OED (10 secs)

Methods	Precision (%)	Recall (%)	F Score (%)
DAE-MLP	80.00	82.22	81.09
DAE-LSTM	74.29	80.27	77.17
S-NMF	78.51	76.11	77.29
RNBSC (Proposed)	**84.34**	**85.28**	**84.81**

magnitude spectra feature space. For OED we use frame size of 32 ms and a frame step 8 ms. All the audio materials have sampling fequency of 16 KHz. We use FFT size of 512, hence our feature space is $\frac{512}{2} + 1 = 257$ dimensional. For A3Novelty database following [8], 30 ms frame length and frame step of 10 ms are used.

For AE based approaches we have also tried Compression AE structures i.e., with less number of hidden units than input units. But we found out that Denoising AE structures perform better than traditional AEs, supporting results presented in [8]. Hence we only include results for DAEs. For our proposed method RNBSC, all the hyper parameters have been chosen empirically by maximizing F-score over a small held out dev set (10 % of test set) and they are as follows: $\lambda = 0.001, \mu = 0.01, \beta = 4$. For each subdictionary, representing one acoustic background, 50 basis vectors was used.

For all our experiments we use segment based performance metrics i.e., Precision, Recall and F-score, following the standard scoring techniques to evaluate sound event detection systems presented in [27]. For A3Novelty database, we use segment size of 1 s to evaluate all the algorithms, whereas for OED we use both 1 s segment and 10 s segment to score the system outputs. We found out that for EBR higher than 0 dB, recall of all the systems significantly improves. Further tuning/ increasing of β is required for those cases to reduce the false positives (increase precision). For that reason for evaluations on OED we only include testing material of 0 dB EBR. Proposed method and the competing methods have been trained separately for two datasets.

4.3. Results

In Table 1, we report the evaluation results of all competing algorithms over A3Novelty corpus. As discussed above, the lack of variability in acoustic background makes this corpus relatively easier to detect novelties, hence all the competing algorithms produce F-score over 90%. DAE-LSTM and our proposed method RNBSC performs the best among 4 methods (DAE-LSTM does slightly better (0.03%) than the proposed method). Our results using DAE-LSTM is also comparable (our reported result is better) to what was reported in [18] using DAE-LSTM structure for this corpus.

In Table 2 and Table 3 we report evaluation results over OED for all competing methods using 10 s segment and 1 s segment respectively. For both the cases, our proposed approach RNBSC performs the best and outperforms AE based approaches. We also report results using S-NMF and show the usefulness of the robustness, i.e., the extra supergaussian outlier term.

Fig. 1 shows the outlier estimate (r) and model mismatch error (e) for an audio clip with cafe acoustic background, and a novel gunshot sound. We highlight in Fig. 1, the position of the gun shot in the spectrogram. In the bottom figure of Fig. 1, we clearly see that the novel sound is being captured in the outlier estimate and not leaking in to the model mismatch error. Because of the supergaussian/sparse constraint, for all other times energy of the outlier term is close to zero.

Table 3: Results over OED (1 sec)

Methods	Precision (%)	Recall (%)	F Score (%)
DAE-MLP	72.31	74.72	73.49
DAE-LSTM	70.95	70.55	70.75
S-NMF	75.08	67.78	71.24
RNBSC (Proposed)	**77.17**	**81.67**	**79.35**

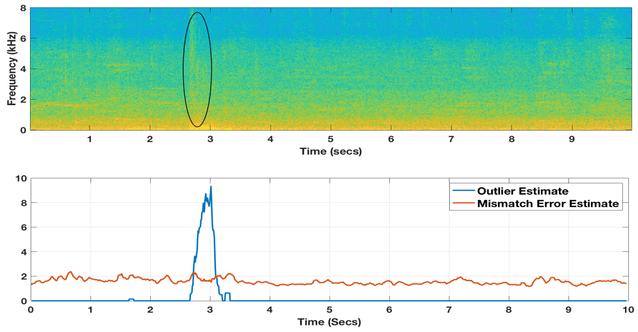


Figure 1: Spectrogram (top), Outlier and Mismatch Error estimate using RNBSC (Bottom) for an audio clip in cafe with Gunshot (novel sound)

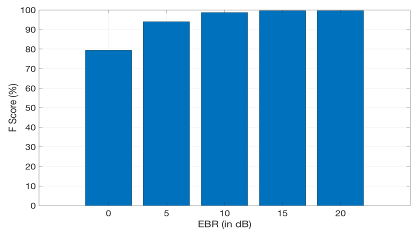


Figure 2: F Score using RNBSC over OED (1 sec) for different EBR

Finally in Fig. 2, we show F score measures of our proposed method for different Event to Background Ratio.

5. CONCLUSION

We have presented a novel unsupervised approach for acoustic novelty detection, using robust non-negative block sparse coding. Previous state-of-the-art autoencoder based approaches solve the problem by modeling only the normal acoustic background, and they detect novel sounds only when the reconstruction/ model mismatch error is above a certain threshold. Our approach on the other hand explicitly models the novel sound using a supergaussian random variable and thresholds on the energy of the expected value of that random variable to detect acoustic novelties. This makes our system much more robust in highly non-stationary acoustic backgrounds, as shown by our empirical results over OED, which has 6 different acoustic backgrounds.

6. REFERENCES

[1] M. Valera and S. A. Velastin, "Intelligent distributed surveillance systems: a review," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 152, no. 2, pp. 192–204, 2005.

[2] G. Valenzise, L. Gerosa, M. Tagliasacchi, F. Antonacci, and A. Sarti, "Scream and gunshot detection and localization for audio-surveillance systems," in *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on*, pp. 21–26, IEEE, 2007.

[3] J. T. Geiger and K. Helwani, "Improving event detection for audio surveillance using gabor filterbank features," in *Signal Processing Conference (EUSIPCO), 2015 23rd European*, pp. 714–718, IEEE, 2015.

[4] S. Lecomte, R. Lengellé, C. Richard, F. Capman, and B. Ravera, "Abnormal events detection using unsupervised one-class svm-application to audio surveillance and evaluation," in *Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference on*, pp. 124–129, IEEE, 2011.

[5] F. Aurino, M. Folla, F. Gargiulo, V. Moscato, A. Picariello, and C. Sansone, "One-class svm based approach for detecting anomalous audio events," in *Intelligent Networking and Collaborative Systems (INCoS), 2014 International Conference on*, pp. 145–151, IEEE, 2014.

[6] R. Bardeli and D. Stein, "Uninformed abnormal event detection on audio," in *Speech Communication; 10. ITG Symposium; Proceedings of*, pp. 1–4, VDE, 2012.

[7] S. Ntalampiras, I. Potamitis, and N. Fakotakis, "Probabilistic novelty detection for acoustic surveillance under real-world conditions," *IEEE Transactions on Multimedia*, vol. 13, no. 4, pp. 713–719, 2011.

[8] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, "A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional lstm neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 1996–2000, IEEE, 2015.

[9] L. Tarassenko, P. Hayton, N. Cerneaz, and M. Brady, "Novelty detection for the identification of masses in mammograms," 1995.

[10] L. Clifton, D. A. Clifton, P. J. Watkinson, and L. Tarassenko, "Identification of patient deterioration in vital-sign data using one-class support vector machines," in *Computer Science and Information Systems (FedCSIS), 2011 Federated Conference on*, pp. 125–131, Citeseer, 2011.

[11] Y.-H. Liu, Y.-C. Liu, and Y.-J. Chen, "Fast support vector data descriptions for novelty detection," *IEEE Transactions on Neural Networks*, vol. 21, no. 8, pp. 1296–1313, 2010.

[12] C. Surace and K. Worden, "Novelty detection in a changing environment: a negative selection approach," *Mechanical Systems and Signal Processing*, vol. 24, no. 4, pp. 1114–1128, 2010.

[13] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer networks*, vol. 51, no. 12, pp. 3448–3470, 2007.

[14] M. Markou and S. Singh, "A neural network-based novelty detector for image sequence analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1664–1677, 2006.

[15] M. Markou and S. Singh, "Novelty detection: a review—part 1: statistical approaches," *Signal processing*, vol. 83, no. 12, pp. 2481–2497, 2003.

[16] M. Markou and S. Singh, "Novelty detection: a review—part 2:: neural network based approaches," *Signal processing*, vol. 83, no. 12, pp. 2499–2521, 2003.

[17] Y. Kawaguchi, T. Endo, K. Ichige, and Hamada, "Non-negative novelty extraction: A new non-negativity constraint for nmf," in *Acoustic Signal Enhancement (IWAENC), 2018 IEEE International Workshop on*, IEEE, 2018.

[18] E. Marchi, F. Vesperini, S. Squartini, and B. Schuller, "Deep recurrent neural network-based autoencoders for acoustic novelty detection," *Computational intelligence and neuroscience*, vol. 2017, 2017.

[19] J. Le Roux, F. J. Weninger, and J. R. Hershey, "Sparse nmf–half-baked or well done?," *Mitsubishi Electric Research Labs (MERL), Cambridge, MA, USA, Tech. Rep., no. TR2015-023*, 2015.

[20] A. Lefevre, F. Bach, and C. Févotte, "Itakura-saito nonnegative matrix factorization with group sparsity," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pp. 21–24, IEEE, 2011.

[21] D. E. Tyler, "Robust statistics: Theory and methods," 2008.

[22] K. L. Lange, R. J. Little, and J. M. Taylor, "Robust statistical modeling using the t distribution," *Journal of the American Statistical Association*, vol. 84, no. 408, pp. 881–896, 1989.

[23] Y. Jin and B. D. Rao, "Algorithms for robust linear regression by exploiting the connection to sparse signal recovery," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 3830–3833, IEEE, 2010.

[24] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, pp. 556–562, 2001.

[25] R. Zhao and V. Y. Tan, "Online nonnegative matrix factorization with outliers," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pp. 2662–2666, IEEE, 2016.

[26] A. Mesaros, T. Heittola, and T. Virtanen, "Tut database for acoustic scene classification and sound event detection," in *Signal Processing Conference (EUSIPCO), 2016 24th European*, pp. 1128–1132, IEEE, 2016.

[27] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

SOUND SOURCE LOCALISATION IN AMBISONIC AUDIO USING PEAK CLUSTERING

Marc C. Green & Damian Murphy

AudioLab, Department of Electronic Engineering, University of York

ABSTRACT

Accurate sound source direction-of-arrival and trajectory estimation in 3D is a key component of acoustic scene analysis for many applications, including as part of polyphonic sound event detection systems. Recently, a number of systems have been proposed which perform this function with first-order Ambisonic audio and can work well, though typically performance drops when the polyphony is increased. This paper introduces a novel system for source localisation using spherical harmonic beamforming and unsupervised peak clustering. The performance of the system is investigated using synthetic scenes in first to fourth order Ambisonics and featuring up to three overlapping sounds. It is shown that use of second-order Ambisonics results in significantly increased performance relative to first-order. Using third and fourth-order Ambisonics also results in improvements, though these are not so pronounced.

***Index Terms*—** sound source localisation, direction of arrival, spatial audio, beamforming, steered-response power, DBSCAN

1. INTRODUCTION

Sound Event Localisation and Detection (SELD) is the act of detecting and tracking individual sounds in an acoustic scene consisting of a mixture of sources, typically recorded or monitored using a microphone array. Such a system has applications including audio surveillance [1], vehicle tracking for the military [2], localisation of targets in robotics [3], and as a stage in source separation that has been proposed for use in evaluation of environmental soundscapes [4, 5]. Previous work involving SELD in Ambisonics is limited to FOA [6, 7]. These systems tend to work well when localising a single source, but performance drops when the complexity of the scene is increased by adding multiple sources overlapping in time. Higher-order Ambisonic (HOA) audio has much higher spatial resolution than FOA, and there are now several portable HOA microphones commercially available, including the second-order Core-Sound OctoMic [8] and fourth-order mh Acoustics Eigenmike [9], making it simple to gather high-order Ambisonic recordings.

The EigenScape database of acoustic scenes [10] was recorded in HOA using the Eigenmike. Analysis of this database has shown that spatial audio features can be useful in acoustic scene classification [10, 11], indicating that use of spatial audio could be a fruitful area of investigation in future work on soundscape analysis. Using spatial audio to consider individual sources will require a robust method for SELD.

In this paper we introduce a new method for estimation of onset/offset times and the DOA of active sound sources (covering the first two stages of SELD as defined in [6]) in Ambisonic recordings of acoustic scenes using spherical harmonic beamforming and unsupervised clustering of power peaks by the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [12]. An unsupervised approach such as that presented here could be especially useful when the amount of data available for training is small. The paper is organised as follows: Section 2 provides a detailed description of the different stages involved in the system, including beamforming, peak-finding, clustering and regression. Section 3 describes the evaluation procedure, including the data used to test the system and the metrics used for assessment and optimisation. Section 4 presents the results of the study, with Section 5 providing discussion. Section 6 concludes the paper.

Funding was provided by a UK Engineering and Physical Sciences Research Council (EPSRC) Doctoral Training Award, an Audio Engineering Society (AES) Educational Foundation grant, and the Department of Electronic Engineering at the University of York. Thanks to Sharath Adavanne and the team at TUNI Audio Research Group for assistance with the dataset.

2. METHOD

Our system uses four main steps that will be described in detail in this section. First, spherical harmonic beamforming is used to create steered-response power (SRP) maps. These maps are then analysed to extract a list of peak power positions. DBSCAN is used to create clusters from these peaks, which correspond to identified sound sources. Finally, regression models are fit to each cluster for smoothed trajectory estimates.

2.1. Steered-Response Power Map

The first stage of the system is the creation of a series of power maps describing how the sound power varies in the scene over time. Spherical harmonic beamforming is used to create an SRP map [13] for each frame of audio as follows:

$$Z(\theta,\phi) = \sum_{k}\sum_{n=0}^{N}\frac{1}{b_n(k)}\sum_{m=-n}^{n} W_{nm}(\theta,\phi)P_{nm}(k) \qquad (1)$$

where output power Z for azimuth θ and elevation ϕ is calculated as the product of the spherical harmonic-wavenumber domain signals P for spherical harmonic order n, degree m and wavenumber k, and a weighting function W that determines the look direction of the beam. This calculation is repeated and summed across all n and m [14], and across k for a map describing power in all frequency bands. The $b_n(k)$ term is required to compensate for scattering of sound induced by the presence of the recording array [13, 14], here set to 1 as the system was tested using synthetic sound scenes. In this work, the simplest spherical harmonic beamformer was used, where the weights are simply substituted for the spherical harmonics for the given look direction, a process also known as plane-wave decomposition [15].

To create the SRP map, the beam must be steered in multiple directions to sample the 3D space. We used the Fibonacci spiral [16] to distribute 600 points in a nearly-uniform spherical pattern.

https://doi.org/10.33682/05my-4306

This distribution was chosen as it can generate any number of points with the only major irregularities in spacing occurring near the poles [17], which are uncommon positions for sound sources. A map was generated for each frame of audio, describing how the sound power impacting on the measurement position changed over time. We split our audio into frames of 256 samples, using rectangular windows and no overlap.

2.2. Peak-finding

The series of SRP maps is then passed to a peak-finding algorithm. Peak-finding in a spherical function presents a challenge in that the wraparound of the sphere at the edges of the data i.e. $f(2\pi, \phi) = f(0, \phi)$ has to be taken into account, along with the fact that the sphere has not been sampled with a regular grid.

We used the peak-finding function from the *dipy* Python library [18]. Originally designed for analysis of MRI data, this peak-finder overcomes these issues by requiring the sampling directions as input as well as the power map. There are two parameters that govern the behaviour of the function. The first, *rel_pk*, is used to calculate a threshold below which to discard peaks by:

$$\text{threshold} = \wedge + \textit{rel\_pk} \cdot R \tag{2}$$

where R is the range of the data and $\wedge$ is either the minimum of the data or 0 if the minimum is negative. The second factor is *min_sep*, an angular distance that governs the minimum separation allowed between peaks. This helps avoid groups by discarding peaks that are found within this distance of each other - only the largest peak is retained. The algorithm returns an indeterminate number of peaks, so we allocated enough memory for a maximum of 20 per frame. This could easily be extended if the application demanded it, but in practise this limit was rarely approached. The output of this stage is a list of vectors containing the angle of the detected peaks along with the time in seconds.

2.3. Clustering

In order to estimate coherent sound sources, we used the DBSCAN algorithm [12] to intelligently cluster sets of peaks proximal in space and time. DBSCAN is an unsupervised algorithm that groups data into clusters based on their proximity, with points in low-density regions (having fewer neighbours) designated as outliers. This algorithm is very useful in terms of estimating which peaks belong to the same source sounds. Onset and offset times for each source can be predicted by considering the first and last-occurring peak points grouped into each cluster.

To once again avoid the problems mentioned in Section 2.2 involving spherical wraparound points, the spherical co-ordinate component of each peak vector is converted to Cartesian co-ordinates. Each peak is therefore mapped from 3D (t, θ, ϕ) to 4D (t, x, y, z), similar to the approach used in [6]. Without this process, there would be a disconnect in the clusters identified by DBSCAN as sources moved across or near to the spherical co-ordinate boundaries.

The spatial dimensions of the data were normalised, as is standard in machine learning, to zero mean and unit variance. The time dimension was not collapsed, as in testing this resulted in clusters being made of peaks occurring in similar spatial locations but separated by large amounts of time. There are two main input parameters for the DBSCAN algorithm:

- ϵ - The largest distance between two adjacent points before the algorithm considers assigning the points to different clusters.
- *MinPts* - The number of data points required within ϵ of a given point for that point to be considered a 'core' point. This affects how dense groups need to be in order to be clustered.

2.4. Regression

Each cluster is used to train a set of Support Vector Regressors (SVRs) [19], which create models of source trajectories. Since the clusters are labelled by the DBSCAN stage, this stage of learning is supervised. A separate regressor is trained for each spatial dimension, modelling x, y, and z separately against t, and the outputs of these three models are combined for a final 4D trajectory. The regressors serve to smooth the raw data, which can exhibit a certain amount of 'jitter' as adjacent sample points are instantaneously identified as peaks in a given frame. The model can also be used to fill in missing points in the cluster, as there might not necessarily be a peak identified in the cluster for every time step. In this way we provide some mitigation for interference.

The salient input parameter for the SVR algorithm in terms of this study is C, which is the cost associated with the distance of input data from the regression line. A higher value of C causes overfitting as the cost associated with points not coinciding with the line is high. In this study we determined experimentally to use 1×10^{-3} as the value for C, as this ensured smoother predicted trajectories with minimal jitter. The output of these regressors is calculated for every frame between the first and last points of each cluster. The predictions are then re-scaled back to the original spatial ranges and the Cartesian co-ordinates are converted back to spherical co-ordinates, giving the final output of the system.

3. EVALUATION

3.1. Dataset

The system was tested using an expanded version of the TUT Sound Events 2018 Ambisonic Anechoic Dataset [20]. This dataset features synthetic Ambisonic scenes with sounds at static locations in the full range of θ and at ϕ between $\pm 60°$, with a resolution of $10°$. Scenes are included with three levels of polyphony, up to a maximum of one, two or three simultaneous sounds active (denoted *OV1* to *OV3*). Using synthetic data, the level of polyphony, position, and movement of sounds is controllable and can therefore be precisely known. Real recordings would have to be labelled manually and this would be very labour-intensive. Indeed, it is not clear how one would go about labelling real recordings for DOA in a way that would be at all reliable.

Since the original dataset is only available in FOA, we re-synthesised it in fourth-order HOA using the original scene description files and source sounds from the DCASE 2016 task 2 dataset [21], which contains a variety of everyday sounds. See [7] for more detail on the method for synthesising the data. The dataset features 240 training examples and 60 testing examples for each *OV*. Our system has no need of training, so we used only the testing examples. The examples were all resampled to 16 kHz, as would be necessary with real-world Eigenmike recordings in order to avoid spatial aliasing artefacts due to the geometry of the array [22].

3.2. Metrics

To assess the system we employ the two frame-wise DOA metrics used in the DCASE 2019 Task 3 challenge [6, 7, 23]. The first is *DOA error*, defined as:

$$\text{DOA error} = \frac{1}{\sum_{t=1}^{T} D_E^t} \sum_{t=1}^{T} \mathcal{H}(\mathbf{DOA}_R^t, \mathbf{DOA}_E^t) \qquad (3)$$

where $\mathbf{DOA}_R^t$ and $\mathbf{DOA}_E^t$ are lists of reference and estimated DOAs, respectively, in frame t. D_E^t is the number of estimates in $\mathbf{DOA}_E^t$, and $\mathcal{H}$ is the Hungarian algorithm [24], used to assign predicted angles to reference angles based on optimising pair-wise costs using angular distances. This metric gives the average error between predicted and actual DOA angles.

The second metric is *frame recall* (FR), formally defined for DCASE 2019 as [23]:

$$\text{FR} = \frac{1}{T} \sum_{t=1}^{T} \mathbb{1}(D_R^t = D_E^t) \qquad (4)$$

where D_R^t is the ground-truth number of sources present in frame t, and $\mathbb{1}$ is an indicator function which outputs one where the bracketed condition is met, otherwise returning zero. FR indicates the proportion of frames where the estimated and reference number of active sounds are equal. A perfect system would have a FR of 1 and a DOA error of 0.

3.3. Optimisation

To assess the system, we ran it on all 60 test files for each order of Ambisonics from first to fourth-order (denoted *N1* to *N4*) and each level of polyphony available in the dataset. Metrics were calculated for each file, with their means calculated to characterise the system's performance across the whole dataset.

To find the best possible performance for each *N* and *OV*, we used the *hyperopt* library [25] to run 1000 iterations using various combinations of hyperparameters, optimising for FR. Following preliminary tests to find appropriate ranges, we set the search space as follows:

- $\{\epsilon \in \mathbb{R} \mid 0.1 \leq \epsilon \leq 1.25\}$
- $\{\mathit{MinPts} \in \mathbb{Z} \mid 3 \leq \mathit{MinPts} \leq 10\}$
- $\{\mathit{rel\_pk} \in \mathbb{R} \mid 0.0 \leq \mathit{rel\_pk} \leq 1.0\}$
- $\{\mathit{min\_sep} \in \mathbb{Z} \mid 0 < \mathit{min\_sep} < 90\}$

Hyperopt uses the Tree Parzen Estimator (TPE) algorithm [26] to focus on optimal values over time. This enables more fine-tuning of the system's performance compared to the same number of iterations in a random search.

4. RESULTS

4.1. Optimised Systems

Figure 1 shows the performance metrics recorded for each level of overlap and Ambisonic order from systems optimised for maximum FR, along with results from SELDnet reported in [6], as a comparison. It can be seen that performance on *OV1* audio is very good regardless of *N*, with almost perfect FR and low DOA error of around 3°. For *OV2* there is a clear pattern of improvement in performance

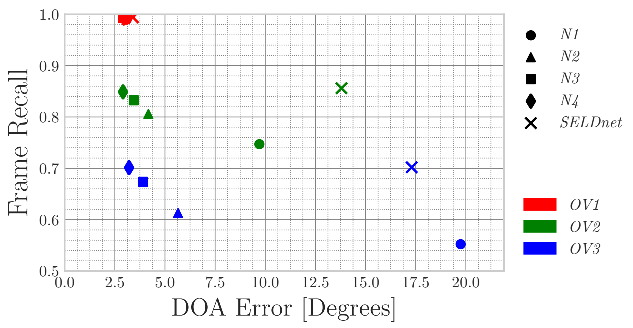


Figure 1: Plot of DOA Error against FR for systems using parameters maximising FR, as well as reported SELDnet results [6].

with increase of *N*. *OV2/N4* yields FR of 0.85 with DOA error of 2.9°, with *OV2/N2* yielding FR of 0.81 with a DOA error of 4.2°, remarkably close to the *OV2/N4* performance. There is a larger gap between *OV2/N1* and *OV2/N2* results than between the other orders. This pattern of performance difference across *N* appears to be more pronounced as *OV* increases. For *OV3/N1* the DOA error is a relatively poor 19.7°, with a FR of 0.55. *OV3/N2* reduces the DOA error to 5.7°, an improvement of 14°, whilst the gap between *OV3/N2* and *OV3/N4* is just 2.5°. FR also increases with *N*, though the difference is not so marked as that of DOA error.

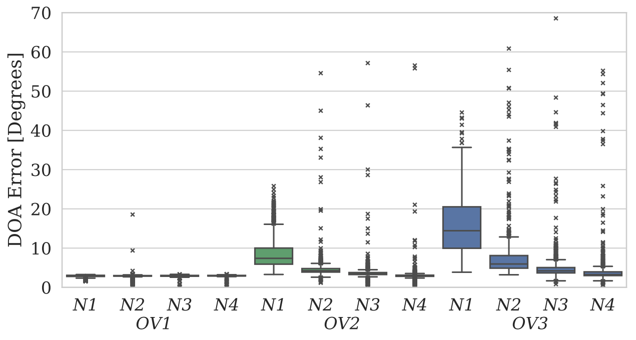
The results achieved for *OV1* are very closely aligned with those achieved by SELDnet. DOA error for *OV2* is smaller than the SELDnet result in all orders, but FR does not begin to approach the SELDnet result until higher orders are used. SELDnet outperforms this system for *OV3/N1*, but is outperformed in terms of DOA error for *OV3* using all higher orders. Again, the FR achieved by SELDnet is only approached using higher orders.

4.2. Performance Variance

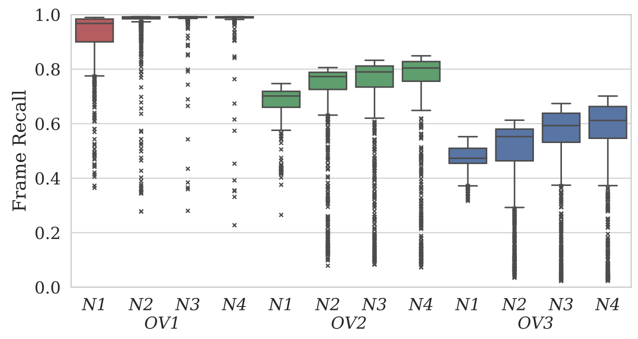
Figure 2 shows the distribution of results returned by all 1000 iterations of the system using various hyperparameters. It should be noted that due to the use of the TPE algorithm these results will be skewed, with more data on performance with hyperparameters set close to optimal values. This accounts for the large number of visible outliers, although they represent only a small proportion of the 1000 iterations.

It can clearly be seen in Figure 2(a) that increasing the order of the beamformer decreases the median and variance of DOA Error for *OV2*, and to an ever greater degree for *OV3*. Similarly to the pattern of results in Figure 1, the largest reduction in both is between *N1* and *N2*, with higher orders yielding diminishing returns in this regard. The comparatively low variance in systems using *N2* or above indicates a degree of robustness to varying hyperparameters, at least within a certain range. This could be a benefit when using the system on real-world audio in which the precise number of sources would usually be unknown.

Returning attention to the outliers, it can be seen that there are iterations of the system that achieved very low DOA error values. These are not, however, the results shown in Figure 1, as achieving this low DOA error incurs a trade-off whereby FR becomes poor. Further investigation indicated that these metrics were recorded on iterations where both peak-finding parameters discussed in Section 2.2 were set very low. This results in clusters of peaks being identi-



(a) DOA Error



(b) Frame Recall

Figure 2: Distributions of DOA Error and Frame Recall system performance metrics across all 1000 *hyperopt* iterations.

fied in the directions of the sound sources, as opposed to the single peaks enabled when the parameters are set more appropriately. The presence of these clusters may enable the regression stage to interpolate and find a truer DOA for the source lying in the centre of these peaks. Unfortunately, lowering these hyperparameters also results in an increase of spurious peaks, leading to greatly reduced FR. Despite these considerations, in a context where precise DOA measurement was of greater importance than an exact number of active sources, such optimisation for DOA error might be desirable.

The distributions of FR values are shown in Figure 2(b). Once again, higher Ambisonic orders tend to have better performance both in median and maximum values. There is once again a larger increase in performance between *N1* and *N2* than for other orders, though this is not as pronounced as with DOA error values. Unlike DOA error, FR declines consistently given increasing levels of sound overlap. The variance in FR for *OV1* decreases between *N1* and *N2*, but this pattern is not repeated for *OV2*, where variance remains consistent regardless of *N*, or *OV3*, where variance actually increases between *N1* and *N2*.

5. DISCUSSION

Results indicate, in terms of best performance as well as average performance and variance, that there is a larger gap between *N1* and *N2* than between *N2* and *N3* or *N4*. Increasing *N* increases the computational complexity of the beamforming stage, as well as the amount of storage space required for the recorded data and the number of microphone capsules required to capture real-world audio. Since this increases cost at all stages, we have an incentive to keep *N* low. The jump in performance between *N1* and *N2* indicates that *N2* may be worth the increased cost, yet limited performance gains at higher orders suggests that second-order Ambisonics might mark a good compromise point for this application. On the other hand, the results do show that the improvements in performance with increased *N* become more pronounced as *OV* increases. Since real-world acoustic scenes are far more complex than the synthesised scenes tested here, use of higher orders may still be useful dependent on context. It is interesting to note that the lowest DOA error achieved at each *OV* in the optimised systems shown in Figure 1 are very similar, at around 3°. This corresponds closely to the average angular distance between pairs of adjacent points in the 600-point Fibonacci spiral, which is 2.72°, indicating that the system could achieve even lower DOA values if a finer grid pattern or some method of interpolation were employed (though this would complicate the peak-finding stage).

The fact that FR appears to decrease linearly with increasing *OV* is interesting, especially given that the synthetic scenes used here are anechoic. The diminishing returns in terms of improvements with increasing Ambisonic order indicate a trend towards a maximum performance level which is clearly less than perfect. The best results achieved here are very closely aligned with the results from SELDnet, which provides some evidence there may be a ceiling inherent in either the dataset used or more fundamentally with this type of approach. Increasing *N* will likely result in smaller and smaller improvements whilst at the same time increasing computational complexity exponentially. This indicates that improvements to FR will probably require an improved or alternative method for producing the power map than the plane-wave decomposition SRP method used here or the neural network-generated spatial pseudo-spectrum used in [6, 7]. It is also possible that given dynamic scenes with multiple moving sources that when the trajectories of two or more sources intersect, the DBSCAN algorithm used here would link them together, thus causing the regression stage to produce wildly erroneous DOA estimates. One potential solution to this could be utilising the different frequency bands present in the power map calculation (e.g. not summing over k in Equation 1) to add another dimension that would make it less likely for overlapping sounds to be clustered provided they remained in different frequency ranges (ω-disjoint orthogonality [4]).

6. CONCLUSION

In this paper, we have specified and tested a system using spherical harmonic beamforming and unsupervised peak clustering for conducting sound event localisation in Ambisonic recordings of acoustic scenes. The system has been tested on synthetic scenes it has been shown that performance given a single active source is consistently very good across all Ambisonic orders, with reductions in performance occurring as the number of concurrent sources is increased. Increasing Ambisonic order improves performance, especially between first and second-order.

Future work on this system could seek to improve frame recall by using an alternative method for calculating the power map. DOA error performance could be improved by introducing interpolation to the peak-finding stage. Apart from these improvements, the obvious next step would be to add a labelling stage, making this a fully-fledged SELD system.

7. REFERENCES

[1] M. Crocco, M. Cristani, A. Trucco, and V. Murino, “Audio Surveillance: A Systematic Review,” *ACM Computing Surveys*, vol. 48, no. 4, Feb 2016. [Online]. Available: https://arxiv.org/abs/1409.7787

[2] M. R. Azimi-Sadjadi and N. R. A. Pezeshki, “Wideband DOA Estimation Algorithms for Multiple Moving Sources using Unattended Acoustic Sensors,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 44, no. 4, pp. 1585–1598, October 2008.

[3] K. Okutani, T. Yoshida, K. Nakamura, and K. Nakadai, “Outdoor Auditory Scene Analysis Using a Moving Microphone Array Embedded in a Quadrocopter,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[4] O. Bunting and D. Chesmore, “Time frequency source separation and direction of arrival estimation in a 3D soundscape environment,” *Applied Acoustics*, vol. 74, no. 2, pp. 264–268, Feb 2013. [Online]. Available: http://dx.doi.org/10.1016/j.apacoust.2011.05.018

[5] O. Bunting, J. Stammers, D. Chesmore, O. Bouzid, G. Y. Tian, C. Karatsovis, and S. Dyne, “Instrument for soundscape recognition, identification and evaluation (ISRIE): technology and practical uses,” in *Euronoise 2009*, October 2009.

[6] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, “Sound event localization and detection of overlapping sources in three dimensions using convolutional recurrent neural networks,” *IEEE Journal of Selected Topics in Signal Processing*, 2018. [Online]. Available: https://arxiv.org/abs/1807.00129

[7] S. Adavanne, A. Politis, and T. Virtanen, “Direction of Arrival Estimation for Multiple Sound Sources Using Convolutional Recurrent Neural Network,” in *EUSIPCO 2018*, 2018. [Online]. Available: https://arxiv.org/abs/1710.10059

[8] “Core sound octomic,” http://www.core-sound.com/OctoMic/1.php, accessed: 2019-06-20.

[9] mh Acoustics, *em32 Eigenmike® microphone array release notes*, mh acoustics, 25 Summit Ave, Summit, NJ 07901, April 2013. [Online]. Available: https://mhacoustics.com/sites/default/files/EigenmikeReleaseNotesV15.pdf

[10] M. C. Green and D. Murphy, “Eigenscape: A database of spatial acoustic scene recordings,” *Applied Sciences*, vol. 7, no. 11, p. 1204, Nov 2017. [Online]. Available: http://dx.doi.org/10.3390/app7111204

[11] ——, “Acoustic scene classification using spatial features,” in *DCASE 2017*, 2017. [Online]. Available: http://www.cs.tut.fi/sgn/arg/dcase2017/documents/workshop_papers/DCASE2017Workshop_Green_126.pdf

[12] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Second International Conference on Knowledge Discovery and Data Mining*, 1996.

[13] B. Rafaely, *Fundamentals of spherical array processing*. Heidelberg Germany: Springer, 2015.

[14] D. P. Jarrett, “Spherical Microphone Array Processing for Acoustic Parameter Estimation and Signal Enhancement,” Ph.D. dissertation, Department of Electrical & Electronic Engineering, Imperial College London, 2013.

[15] B. Rafaely, “Plane-wave decomposition of the sound field on a sphere by spherical convolution,” *The Journal of the Acoustical Society of America*, vol. 116, no. 4, pp. 2149–2157, Oct 2004. [Online]. Available: http://dx.doi.org/10.1121/1.1792643

[16] E. B. Saff and A. B. J. Kuijlaars, “Distributing many points on a sphere,” *The Mathematical Intelligencer*, vol. 19, no. 1, pp. 5 – 11, 1997.

[17] T. McKenzie, D. Murphy, and G. Kearney, “Diffuse-field equalisation of binaural ambisonic rendering,” *Applied Sciences*, vol. 8, no. 10, p. 1956, Oct 2018. [Online]. Available: http://dx.doi.org/10.3390/app8101956

[18] E. Garyfallidis, M. Brett, B. Amirbekian, A. Rokem, S. van der Walt, M. Descoteaux, and I. Nimmo-Smith, “Dipy, a library for the analysis of diffusion mri data,” *Frontiers in Neuroinformatics*, vol. 8, Feb 2014. [Online]. Available: http://dx.doi.org/10.3389/fninf.2014.00008

[19] A. J. Smola and B. Scholkopf, “A tutorial on support vector regression,” *Statistics and Computing*, vol. 14, pp. 199–222, 2004. [Online]. Available: https://alex.smola.org/papers/2004/SmoSch04.pdf

[20] S. Adavanne, A. Politis, and T. Virtanen, “TUT Sound Events 2018 - Ambisonic, Anechoic and Synthetic Impulse Response Dataset,” Apr. 2018. [Online]. Available: https://doi.org/10.5281/zenodo.1237703

[21] “IEEE DCASE 2016 task 2 dataset,” https://archive.org/details/dcase2016_task2_train_dev, accessed: 2019-06-20.

[22] mh Acoustics, *Eigenbeam Data Specification for Eigenbeams Eigenbeam Data Specification for Eigenbeams Eigenbeam Data Specification for Eigenbeams Eigenbeam Data: Specification for Eigenbeams*, 2016. [Online]. Available: https://mhacoustics.com/sites/default/files/Eigenbeam%20Datasheet_R01A.pdf

[23] “IEEE DCASE 2019 task 3,” http://dcase.community/challenge2019/task-sound-event-localization-and-detection, accessed: 2019-06-20.

[24] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, 1955.

[25] J. Bergstra, D. Yamins, and D. D. Cox, “Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures,” in *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, Atlanta, Georgia, 2013. [Online]. Available: http://proceedings.mlr.press/v28/bergstra13.pdf

[26] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kegl, “Algorithms for hyper-parameter optimization,” in *Neural Information Processing Systems*, Granada, Spain, Dec 2011. [Online]. Available: https://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf

SOUND EVENT LOCALIZATION AND DETECTION USING CRNN ON PAIRS OF MICROPHONES

François Grondin, James Glass[*]

Massachusetts Institute of Technology
CSAIL, 32 Vassar St.
Cambridge, MA 02139, USA
{fgrondin,glass}@mit.edu

Iwona Sobieraj, Mark D. Plumbley[†]

University of Surrey
Dept. Elec. & Elec. Eng., 388 Stag Hill
Guildford, Surrey, GU2 7XH, UK
{i.sobieraj,m.plumbley}@surrey.ac.uk

ABSTRACT

This paper proposes sound event localization and detection methods from multichannel recording. The proposed system is based on two Convolutional Recurrent Neural Networks (CRNNs) to perform sound event detection (SED) and time difference of arrival (TDOA) estimation on each pair of microphones in a microphone array. In this paper, the system is evaluated with a four-microphone array, and thus combines the results from six pairs of microphones to provide a final classification and a 3-D direction of arrival (DOA) estimate. Results demonstrate that the proposed approach outperforms the DCASE 2019 baseline system.

Index Terms— Sound event detection, sound source localization, time difference of arrival, neural network

1. INTRODUCTION

Sound Event Detection (SED) is an important machine listening task, which aims to automatically recognize, label, and estimate the position in time of sound events in a continuous audio signal. This is a popular research topic, due to the number of real-world applications for SED such as home-care [1], surveillance [2], environmental monitoring [3] or urban traffic control [4], to name just a few. Successful Detection and Classification of Acoustic Scenes and Events (DCASE) challenges [5, 6] now provide the community with datasets and baselines for a number of tasks related to SED. However, most of the effort so far has concentrated on classification and detection of the sound events in time only, with little work done to perform robust localization of sound event in space.

Early approaches for SED are strongly inspired by speech recognition systems, using mel frequency cepstral coefficients (MFCCs) with Gaussian Mixture Models (GMMs) combined with Hidden Markov Models (HMM) [7, 8]. Methods based on dictionary learning, mainly Non-negative Matrix Factorization (NMF), are also considered as prominent solutions for the SED task [9, 10, 11]. With the recent advancements in machine learning, deep learning methods now provide state of the art results for this task [12, 13]. The prevailing architectures used for SED are Convolutional Neural Networks (CNNs) [14], which are particularly successful in computer vision tasks. Other common approaches try to model time relations in audio signal by using recurrent neural networks (RNNs) [12]. Both can be combined in a Convolutional Recurrent Neural Network (CRNN), which achieves state of the art results on several machine listening tasks [15, 16, 17].

On the other hand, sound source localization (SSL) refers to estimating the direction of arrival (DOA) of multiple sound sources. There are two popular categories of SSL methods: 1) high resolution and 2) steered-response techniques. High resolution methods include Multiple Signal Classification (MUSIC) [18] and Estimation of Signal Parameters via Rotational Invariance Technique (ESPRIT) [19]. These approaches, although initially designed for narrowband signals, can be adapted to broadband signals such as speech [20, 21, 22, 23, 24]. Alternatively, the Steered-Response Power Phase Transform (SRP-PHAT) robustly estimates the direction of arrival of speech and other broadband sources [25]. SRP-PHAT relies on the Generalized Cross-Correlation with Phase Transform (GCC-PHAT) between each pair of microphones of a microphone array. It is therefore convenient to estimate the time difference of arrival (TDOA) values for each pair, and combine these results to estimate the direction of arrival for a source [26, 27, 28, 29, 30], which is the approach we choose for this challenge.

In this paper we propose a system for sound event detection and localization (SELD), which we submitted to Task3 of the DCASE2019 Challenge [31]. Motivated by the results obtained by [32], we propose a CRNN architecture that uses both the spectrogram and GCC-PHAT features to perform SED and estimate TDOA. However, since TDOA and SED have different cost functions, we believe they are distinct tasks with different optimal solutions, and we propose to use two separate neural networks for each of these two tasks. The results are then combined together to generate a final SED decision and estimate the DOA.

2. SOUND EVENT LOCALIZATION AND DETECTION

The goal of sound event localization and detection (SELD) is to output all instances of the sound events in the recording, its respective onset-offset times, and spatial locations in azimuth and elevation angles, given a multichannel audio input. An example of such a setup has been provided in Task 3 of the DCASE 2019 Challenge [31]. Our system uses the TAU Spatial Sound Events - Microphone Array dataset, which provides four-channel directional microphone recordings from a tetrahedral array configuration. A detailed description of the dataset and the recording procedure may be found in [17]. In our approach, we propose to predict events and TDOAs for each pair of microphones, which leads to a total six pairs.

[*]This work was funded in part by Signify.

[†]The research leading to these results has received funding from the European Union's H2020 Framework Programme (H2020-MSCA-ITN-2014) under grant agreement n 642685 MacSeNet. MDP is also supported by EPSRC grant EP/N014111/1.

https://doi.org/10.33682/4v2a-7q02

3. PROPOSED METHOD

We propose a method based on a combination of two convolutional recurrent neural networks (CRNNs), that share a similar front-end architecture. The first network, CRNN$_{SED}$, is trained to detect, label and estimate onset and offsets of sound events from a pair of microphones. The second network, CRNN$_{TDOA}$, estimates the TDOA for each pair of microphones and each class of sound events. The SED results of all pairs are then combined together and a threshold is applied to make a final decision regarding sound detection for each class. The TDOAs are also combined together for all pairs of microphones and a DOA is generated for each class. To obtain a DOA from the TDOA values, each potential DOA is assigned a set of target TDOAs, which are found during a initial calibration procedure. Figure 1 shows the overall architecture of the proposed system. The following subsections describe in details each building block of the system.

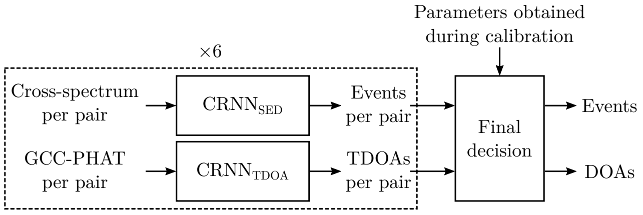


Figure 1: Architecture of the proposed system.

3.1. Calibration

The search space around the microphone array is discretized into Q DOAs, which are indexed by $q \in \mathcal{Q} = \{1, 2, \ldots, Q\}$. Each DOA q is associated to an azimuth and an elevation, denoted by (ϕ_q, θ_q), where $\phi_q \in \{-180°, -170°, \ldots, +170°\}$ and $\theta_q \in \{-40°, -30°, \ldots, +40°\}$, which corresponds to the discrete angles used when recording the DCASE dataset [31]. The number of microphones corresponds to $M \in \mathbb{N}$, and the number of pairs to $P \in \mathbb{N}$, where $P = M(M-1)/2$. Each DOA q also corresponds to a vector $\boldsymbol{\tau}^q \in \mathcal{D}^P$ of TDOA values, where $\mathcal{D} = \{-\tau_{\max}, \ldots, +\tau_{\max}\}$ and the cardinality $|\mathcal{D}| = G$. The expressions $\tau_{\max} \in \mathbb{R}^+$ and $G \in \mathbb{N}$ stand for the maximum TDOA and the number of discrete TDOA values, respectively. Assuming free field propagation of sound, the microphone array geometry and the speed of sound provide enough information to estimate the TDOA values of each DOA. However, the free field assumption becomes inaccurate when dealing with a closed microphone array (e.g. when microphones are installed around a filled support), and thus calibration based on the recorded signals is needed and is performed offline.

The expression $X_m^t[k] \in \mathbb{C}$ stands for the Short-Time Fourier Transform (STFT) coefficient at frame index $t \in \mathbb{N}$, microphone index $m \in \mathcal{M} = \{1, 2, \ldots, M\}$ and bin index $k \in \mathcal{K} = \{K_{\min}, K_{\min}+1, \ldots, K_{\max}\}$, where $K = K_{\max} - K_{\min}$ stands for the total number of frequency bins used. The frame size and hop size correspond to $N \in \mathbb{N}$ and $\Delta N \in \mathbb{N}$, respectively, and the spectral content thus spans frequencies in the interval $[K_{\min} f_S/N, K_{\max} f_S/N]$ Hz, where $f_S \in \mathbb{R}_+$ stands for the sample rate in samples/sec. The complex cross-spectrum $X_{i,j}^t[k]$ for each microphone pair $(i,j) \in \mathcal{P} = \{(x,y) \in \mathcal{M}^2 : x < y\}$ corresponds to:

$$X_{i,j}^{q,\mathcal{T}_q}[k] = \sum_{t \in \mathcal{T}_q} X_i^t[k] X_j^t[k]^* \tag{1}$$

where $\mathcal{T}_q$ is a set that contains all the frame indexes where a single source is active at DOA q, and $(\ldots)^*$ stands for the complex conjugate operator. The Generalized Cross-Correlation with Phase Transform (GCC-PHAT) is then computed as follows:

$$x_{i,j}^{q,\mathcal{T}_q}[\tau] = \sum_{k \in \mathcal{K}} W_{i,j}[\tau, k] \frac{X_{i,j}^{q,\mathcal{T}_q}[k]}{|X_{i,j}^{q,\mathcal{T}_q}[k]|} \tag{2}$$

where $W_{i,j}[\tau, k] = \exp(2\pi\sqrt{-1}\tau k/N)$, with $\tau \in \mathcal{D}$.

The TDOA value for the pair (i,j) and DOA q is then estimated as:

$$\bar{\tau}_{i,j}^q = \arg\max_{\tau \in \mathcal{D}} \{x_{i,j}^{q,\mathcal{T}_q}[\tau]\}. \tag{3}$$

Since there is a limited amount of sound events per DOA in the training dataset, the estimated TDOAs $\bar{\tau}_{i,j}^q \, \forall (i,j) \in \mathcal{P}, \forall q \in \mathcal{Q}$ can be noisy. To cope with this limitation, we apply a polynomial fitting method with an order of 27 (found empirically). For each discrete elevation angle $\theta \in \{-40°, -30°, \ldots, +40°\}$, there are 36 azimuths $\phi \in \{-180°, -170°, \ldots, +170°\}$, and the TDOAs associated to these azimuths vary smoothly. Therefore, for each pair (i,j) and elevation θ, we concatenate the estimated TDOAs three times to create a signal that spans over the azimuths $\phi \in \{-540°, -530°, \ldots, +540°\}$ and avoids the discontinuities observed at $-180°$ and $170°$ within the initial range. A first polynomial fitting is then performed, and the outliers are removed prior to performing a second fitting, which finally provides the estimated TDOA $\tau_{i,j}^q$ for each DOA q for the pair (i,j):

$$\tau_{i,j}^q = \text{polyfit}(\bar{\tau}_{i,j}^q, 27). \tag{4}$$

Figure 2 shows an example of the proposed method and how it deals effectively with outliers. Note that once the polynomial coefficients are obtained, the TDOAs are only estimated in the region of interest, which is in the range $\phi \in \{-180°, -170°, \ldots, +170°\}$.

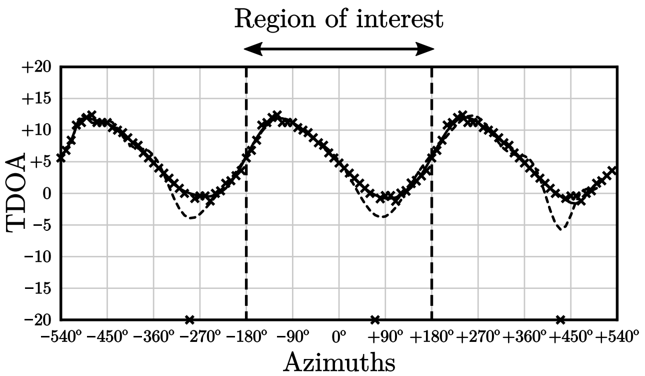


Figure 2: Calibration model for DOA estimation. First polynomial fit is shown as a dashed line, and the second one after removing the outliers is a solid line.

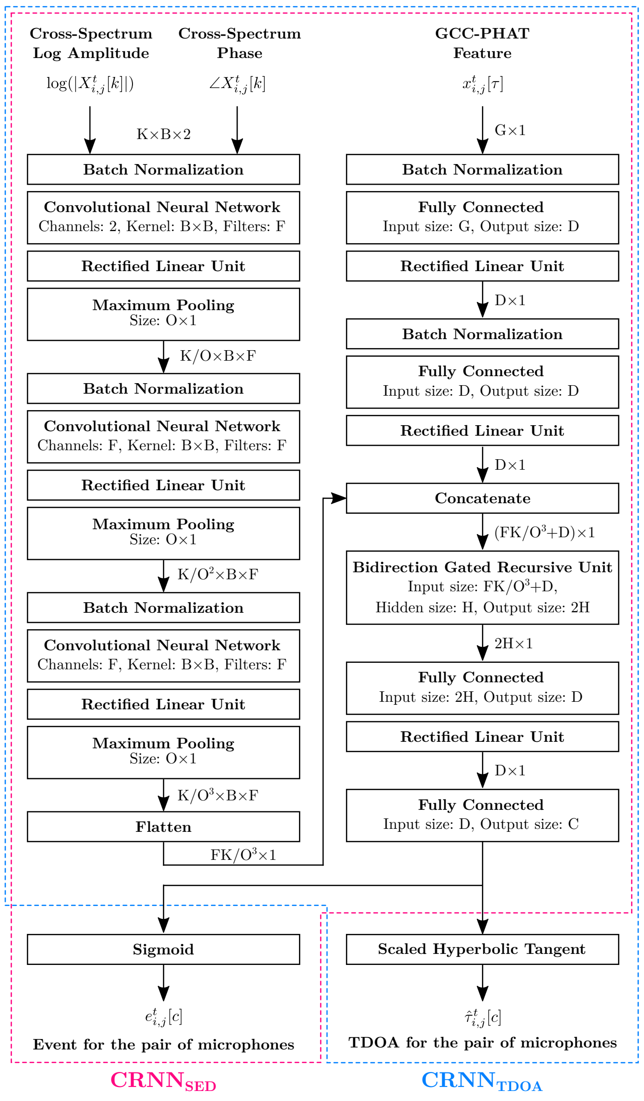

Cross-Spectrum Log Amplitude
Cross-Spectrum Phase
GCC-PHAT Feature
$\log(|X^t_{i,j}[k]|)$
$\angle X^t_{i,j}[k]$
$x^t_{i,j}[\tau]$
K×B×2
G×1
Batch Normalization
Convolutional Neural Network Channels: 2, Kernel: B×B, Filters: F
Rectified Linear Unit
Maximum Pooling Size: O×1
K/O×B×F
Convolutional Neural Network Channels: F, Kernel: B×B, Filters: F
K/O²×B×F
K/O³×B×F
Flatten
FK/O³×1
Fully Connected Input size: G, Output size: D
D×1
Fully Connected Input size: D, Output size: D
Concatenate
(FK/O³+D)×1
Bidirection Gated Recursive Unit Input size: FK/O³+D, Hidden size: H, Output size: 2H
2H×1
Fully Connected Input size: 2H, Output size: D
Fully Connected Input size: D, Output size: C
Sigmoid
Scaled Hyperbolic Tangent
$e^t_{i,j}[c]$
Event for the pair of microphones
$\hat{\tau}^t_{i,j}[c]$
TDOA for the pair of microphones
CRNN_SED
CRNN_TDOA

Figure 3: Architecture of CRNN$_{SED}$ and CRNN$_{TDOA}$

3.2. Neural network architecture

The main building block of our system are two CRNNs that share a similar front-end architecture, as shown in Fig. 3.

The network consists of two branches. This first is a series of convolutional layers (CNN), that process the log amplitude and phase of the instantaneous complex cross-spectrum input spectrograms (as in (1)) between microphones i and j. In parallel, GCC-PHAT features (as in (2), but for a single frame t) are fed into a branch of a network that consists of two feed-forward layers. The outputs of two branches are concatenated and passed to a Bidirectional Gated Recurrent Unit (Bi-GRU) layer. The resultant vector is considered as a task dependent embedding of the input data. The embedding is passed to two feed forward layers, followed by an activation function, which depends on the task of the network.

CRNN$_{SED}$ is trained in a supervised manner using SED labels, i.e. information about the onset, offset and label of a sound event. As SED task may be pinned down to a multi-label classification of time frames, we use binary cross entropy as a loss function of the network. A Sigmoid activation function outputs the probabilities between 0 and 1 of each class for each time frame.

CRNN$_{TDOA}$ is trained on TDOA labels for each pair of microphones. The problem of TDOA estimation is defined in a regression framework. Hence, Mean Squared Error (MSE) loss is used to train the network. Similarly to the CRNN$_{SED}$, the network consists of CNNs and GRU, followed by an activation function, Hyperbolic Tangent (tanh) in this case, scaled by $\tau_{\max}$ as the TDOA value lies in the range $[-\tau_{\max}, +\tau_{\max}]$. Note that the TDOA is only estimated over segments (i.e. audio samples for a given time interval) where the corresponding sound event is active according to the reference labels, as proposed in [32].

Both networks are trained separately on all pairs of microphones, using segments of 3 seconds selected randomly amongst the training dataset, and using the Adam optimizer with a learning rate of 10^{-3} and a batch size of 16. We stopped training the network when no further improvement is observed on the validation set, that is after 120,000 segments for CRNN$_{SED}$ and 160,000 segments for CRNN$_{TDOA}$.

3.3. Event detection

CRNN$_{SED}$ returns a value $e^t_{i,j}[c] \in [0,1]$ for each pair of microphones (i,j) and class $c \in \{1, 2, \ldots, C\}$. These values are summed up for all pairs and each class, and normalized by the number of pairs, which leads to a new expression $e^t[c] \in [0,1]$:

$$e^t[c] = \frac{1}{P} \sum_{i=1}^{M} \sum_{j=i+1}^{M} e^t_{ij}[c]. \tag{5}$$

An event from class c is then considered to be detected at frame t if $e^t[c]$ exceeds a threshold, which is class specific:

$$E^t[c] = \begin{cases} 1 & e^t[c] \geq \epsilon[c] \\ 0 & e^t[c] < \epsilon[c] \end{cases}. \tag{6}$$

A post-filter method finally ensures that each sound event lasts a minimum amount of frames (denoted by γ) to avoid false detection of sporadic events. For evaluation purpose, the event activity is usually defined for a given segment l, where $\mathcal{T}^l = \{tL, tL+1, \ldots, t(L+1)-1\}$ holds the L frames that belong to segment l. The estimated event activity $Event^l_E[c]$ is then said to be active if at least one frames within the interval indicates the event is active.

3.4. DOA estimation

Similarly to CRNN$_{SED}$, CRNN$_{TDOA}$ returns an estimated TDOA $\hat{\tau}^t_{i,j}[c]$ for each class c and pair of microphone (i,j) at frame t. For each DOA at index q, the estimated TDOAs $\hat{\tau}^t_{i,j}[c]$ are compared to the theoretical values $\tau^q_{i,j}$ obtained from polynomial fitting during the calibration step. A Gaussian kernel with a variance of σ^2 then generates a value close to 1 when both TDOAs are close to each other, whereas this value goes to zero when the difference increases. All DOAs are scanned for each class, and the one that returns the maximum sum corresponds to the estimated DOA index $q^t[c]^*$:

$$q^t[c] = \arg\max_{q \in \mathcal{Q}} \sum_{i=1}^{M} \sum_{j=i+1}^{M} \exp\left[\frac{\left(\hat{\tau}^t_{ij}[c] - \tau^q_{ij}\right)^2}{2\sigma^2}\right]. \tag{7}$$

The estimated DOAs are then concatenated in $\mathbf{DOA}^t_E$:

$$\mathbf{DOA}^t_E = \{(\phi_{q^t[c]}, \theta_{q^t[c]})\} \; \forall \; c \text{ where } \hat{E}^t[c] = 1. \tag{8}$$

4. RESULTS

The proposed system is evaluated on the DCASE 2019 development dataset. This set is divided into 4 cross-validation splits of 100 one-minute recordings each, as described in [17]. Table 1 lists the parameters used in the experiments. The sample rate f_S and the number of microphones M match the DCASE dataset parameters. The frame size N corresponds to 43 msecs, which allows a good trade-off between time and frequency resolutions. The hop size ΔN provides a spacing of 20 msecs between frames, which corresponds to the hop length for evaluation in the actual challenge. The values of $K_{\min}$ and $K_{\max}$ are set to provide a frequency range that goes up to 12 kHz (and exclude the DC component), which is where most of the sound event energy lies. The parameter γ is chosen to ensure a minimum sound event duration of 100 msecs, and the standard deviation σ is found empirically to provide a good DOA resolution with G TDOA values. The maximum value for a TDOA is set such that this includes all possible TDOA values for the actual array geometry. Finally, the neural network hyperparameters B, F, O, H and D are found empirically from observed performances with the validation set. Also note that the event thresholds $\epsilon[c]$ are found empirically by scanning values between 0 and 1 and selecting thresholds that lead to the best event detection metrics on the validation set.

Param.	Value
f_S	48000
M	4
N	2048
ΔN	960
$\tau_{\max}$	20.0

Param.	Value
$K_{\min}$	1
$K_{\max}$	513
γ	5
σ	2.0
G	101

Param.	Value
B	3
F	64
O	4
H	512
D	256

Table 1: Parameters of the proposed system

To evaluate the performance of the system, events are defined for segments of 1 sec ($L = 50$). We define the number of true positives (TP^l) for segment l as the number of correctly estimated events with respect to the reference events activity ($Event^l_R[c]$):

$$TP^l = \sum_{c=1}^{C} Event^l_E[c] \cdot Event^l_R[c]. \tag{9}$$

Similarly, the number of false negatives (FN^l) and false positives (FP^l) are given by:

$$FN^l = \sum_{c=1}^{C} Event^l_E[c] \cdot (1 - Event^l_R[c]) \tag{10}$$

$$FP^l = \sum_{c=1}^{C} (1 - Event^l_E[c]) \cdot Event^l_R[c]. \tag{11}$$

Finally the total number of active events corresponds to:

$$N^l = \sum_{c=1}^{C} Event^l_R[c]. \tag{12}$$

We then define substitutions (S^l), deletions (D^l) and insertions (I^l) are defined as:

$$S^l = \min\{FN^l, FP^l\} \tag{13}$$

$$D^l = \max\{0, FN^l - FP^l\} \tag{14}$$

$$I^l = \max\{0, FP^l - FN^l\}. \tag{15}$$

This leads to the event rate (ER) and F1-score (F) metrics [33]:

$$ER = \frac{\sum_l S^l + \sum_l D^l + \sum_l I^l}{\sum_l N^l} \tag{16}$$

$$F = \frac{2\sum_l TP^l}{2\sum_l TP^l + \sum_l FN^l + \sum_l FP^l}. \tag{17}$$

The DOA metrics consist of the DOA error (DOAE) and frame recall (FR) [16]. The DOAE is obtained as follows:

$$DOAE = \left(\sum_{t=1}^{T} D^t_E\right)^{-1} \sum_{t=1}^{T} \mathcal{H}(\mathbf{DOA}^t_R, \mathbf{DOA}^t_E) \tag{18}$$

where D^t_E denotes the number of estimated events, $\mathcal{H}(\dots)$ stands for Hungarian algorithm [16] and $\mathbf{DOA}^t_R$ represents the reference DOA. The pair-wise costs between individual predicted and reference DOAs corresponds to:

$$h = \arccos\left(\sin\phi_E \sin\phi_R + \cos\phi_E \cos\phi_R \cos(\theta_R - \theta_E)\right) \tag{19}$$

where ϕ_E and ϕ_R stand for the azimuth of the estimated and reference DOA, respectively, and θ_E and θ_R stand for the elevation of the estimated and reference DOA, respectively.

Finally, the frame recall corresponds to the following expression, where D^t_R denotes the number of reference events, and $\mathbb{1}(\dots)$ stands for the indicator function that generates an output one if the condition ($D^t_R = D^t_E$) is met, or zero otherwise:

$$FR = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}(D^t_R = D^t_E). \tag{20}$$

Table 2 summarizes the results for the baseline and the proposed method. This shows that the proposed system outperforms the baseline for all metrics, and improves particularly the accuracy of the estimated DOA.

Method	Dataset	ER	F	DOAE	FR
Baseline	Dev.	0.35	80.0%	30.8°	84.0%
	Eval.	0.28	85.4%	24.6°	85.7%
Proposed	Dev.	**0.21**	**87.2%**	**6.8°**	**84.7%**
	Eval.	**0.14**	**92.2%**	**7.4°**	**87.5%**

Table 2: Performances in terms of Error Rate (ER – less is better), F score (F – more is better), Direction of Arrival Error (DOA – less is better) and Frame Recall (FR – more is better).

5. CONCLUSION

In this paper, we propose a system to detect sound events and estimate their TDOA for each pair of microphones, which then combines them to detect sound events and estimate their DOA for a four-microphone array. The proposed method outperforms the DCASE 2019 baseline system.

In future work, additional neural networks architecture should be investigated for SELD. Moreover, making the system work online (by using unidirectional GRU layers for instance) would make the method appealing for real-world applications.

6. REFERENCES

[1] P. van Hengel and J. Anemüller, "Audio event detection for in-home care," in *Proc. ICA*, 2009, pp. 618–620.

[2] J. Kotus, K. Lopatka, and A. Czyzewski, "Detection and localization of selected acoustic events in acoustic field for smart surveillance applications," *Multimed. Tools Appl.*, vol. 68, no. 1, pp. 5–21, 2014.

[3] D. Stowell, M. Wood, Y. Stylianou, and H. Glotin, "Bird detection in audio: A survey and a challenge," in *Proc. IEEE MLSP*, 2016.

[4] F. Meucci, L. Pierucci, E. Del Re, L. Lastrucci, and P. Desii, "A real-time siren detector to improve safety of guide in traffic environment," in *Proc. EUSIPCO*, 2008.

[5] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 26, no. 2, pp. 379–393, 2018.

[6] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, "DCASE 2017 challenge setup: Tasks, datasets and baseline system," in *Proc. DCASE Workshop*, 2017.

[7] T. Heittola, A. Mesaros, A. Eronen, and T. Virtanen, "Context-dependent sound event detection," *EURASIP J. Audio, Spee.*, vol. 2013, no. 1, pp. 1–13, 2013.

[8] A. Diment, T. Heittola, and T. Virtanen, "Sound event detection for office live and office synthetic AASP challenge," in *Proc. IEEE AASP DCASE*, 2013, pp. 1–3.

[9] C. V. Cotton and D. P. Ellis, "Spectral vs. spectro-temporal features for acoustic event detection," in *Proc. IEEE WASPAA*, 2011, pp. 69–72.

[10] T. Komatsu, T. Toizumi, R. Kondo, and Y. Senda, "Acoustic event detection method using semi-supervised non-negative matrix factorization with a mixture of local dictionaries," in *Proc. DCASE Workshop*, 2016, pp. 45–49.

[11] O. Dikmen and A. Mesaros, "Sound event detection using non-negative dictionaries learned from annotated overlapping events," in *Proc. IEEE WASPAA*, 2013.

[12] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proc. IEEE ICASSP*, 2016, pp. 6440–6444.

[13] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, "Large-scale weakly supervised audio classification using gated convolutional neural network," in *Proc. IEEE ICASSP*, 2018, pp. 121–125.

[14] T. Inoue, P. Vinayavekhin, S. Wang, D. Wood, N. Greco, and R. Tachibana, "Domestic activities classification based on CNN using shuffling and mixing data augmentation," DCASE2018 Challenge, Tech. Rep., 2018.

[15] L. JiaKai, "Mean teacher convolution system for DCASE 2018 task 4," DCASE2018 Challenge, Tech. Rep., 2018.

[16] S. Adavanne, A. Politis, and T. Virtanen, "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network," in *Proc. IEEE EUSIPCO*, 2018, pp. 1462–1466.

[17] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *J. Sel. Topics Signal Process.*, vol. 13, pp. 34–48, 2018.

[18] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propag.*, vol. 34, no. 3, pp. 276–280, 1986.

[19] R. Roy, A. Paulraj, and T. Kailath, "Estimation of signal parameters via rotational invariance techniques - ESPRIT," in *Proc. IEEE MILCOM*, 1986.

[20] C. Ishi, O. Chatot, H. Ishiguro, and N. Hagita, "Evaluation of a MUSIC-based real-time sound localization of multiple sound sources in real noisy environments," in *Proc. IEEE/RSJ IROS*, 2009, pp. 2027–2032.

[21] K. Nakamura, K. Nakadai, and H. Okuno, "A real-time super resolution robot audition system that improves the robustness of simultaneous speech recognition," *Adv. Robotics*, vol. 27, no. 12, pp. 933–945, 2013.

[22] H. Teutsch and W. Kellermann, "EB-ESPRIT: 2D localization of mulitple wideband acoustic sources using eigen-beams," in *Proc. IEEE ICASSP*, 2005, pp. 89–92.

[23] S. Argentieri and P. Danès, "Broadband variations of the MUSIC high-resolution method for sound source localization in robotics," in *Proc. IEEE/RSJ IROS*, 2007, pp. 2009–2014.

[24] P. Danès and J. Bonnal, "Information-theoretic detection of broadband sources in a coherent beamspace MUSIC scheme," in *Proc. IEEE/RSJ IROS*, 2010, pp. 1976–1981.

[25] J. DiBiase, H. Silverman, and M. Brandstein, "Robust localization in reverberant rooms," in *Microphone Arrays*. Springer, 2001, pp. 157–180.

[26] F. Grondin, D. Létourneau, F. Ferland, V. Rousseau, and F. Michaud, "The manyears open framework," *Autonomous Robots*, vol. 34, no. 3, pp. 217–232, 2013.

[27] J.-M. Valin, F. Michaud, and J. Rouat, "Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering," *Rob. Auton. Syst.*, vol. 55, no. 3, pp. 216–228, 2007.

[28] J.-M. Valin, F. Michaud, B. Hadjou, and J. Rouat, "Localization of simultaneous moving sound source for mobile robot using a frequency-domain steered beamformer approach," in *Proc. IEEE ICRA*, 2004, pp. 1033–1038.

[29] J.-M. Valin, F. Michaud, and J. Rouat, "Robust 3D localization nad tracking of sound sources using beamforming and particle filtering," in *Proc. IEEE ICASSP*, 2006, pp. 841–844.

[30] F. Grondin and F. Michaud, "Lightweight and optimized sound source localization and tracking methods for open and closed microphone array configurations," *Rob. Auton. Syst.*, vol. 113, pp. 63–80, 2019.

[31] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and detection," in *Submitted to DCASE Workshop*, 2019.

[32] Y. Cao, Q. Kong, T. Iqbal, F. An, W. Wang, and M. D. Plumbley, "Polyphonic sound event detection and localization using a two-stage strategy," *arXiv preprint arXiv:1905.00268*, 2019.

[33] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

MULTIPLE NEURAL NETWORKS WITH ENSEMBLE METHOD FOR AUDIO TAGGING WITH NOISY LABELS AND MINIMAL SUPERVISION

Kexin He [*], *Yuhan Shen* [*] , *Wei-Qiang Zhang* [†]

Beijing National Research Center for Information Science and Technology
Department of Electronic Engineering, Tsinghua University, Bejing 100084, China
hekexinchn@163.com, yhshen@hotmail.com, wqzhang@tsinghua.edu.cn

ABSTRACT

In this paper, we describe our system for the Task 2 of Detection and Classification of Acoustic Scenes and Events (DCASE) 2019 Challenge: *Audio tagging with noisy labels and minimal supervision*. This task provides a small amount of verified data (curated data) and a larger quantity of unverified data (noisy data) as training data. Each audio clip contains one or more sound events, so it can be considered as a multi-label audio classification task. To tackle this problem, we mainly use four strategies. The first is a sigmoid-softmax activation to deal with so-called sparse multi-label classification. The second is a staged training strategy to learn from noisy data. The third is a post-processing method that normalizes output scores for each sound class. The last is an ensemble method that averages models learned with multiple neural networks and various acoustic features. All of the above strategies contribute to our system significantly. Our final system achieved labelweighted label-ranking average precision (lwlrap) scores of 0.758 on the private test dataset and 0.742 on the public test dataset, winning the 2nd place in DCASE 2019 Challenge Task 2.

***Index Terms*—** Audio tagging, noisy label, model ensemble, DCASE

1. INTRODUCTION

The Detection and Classification of Acoustic Scenes and Events (D-CASE) Challenge is gaining increasing interests among researchers with academic and industrial backgrounds. DCASE 2019 is the fifth edition of this challenge and has been held to support the development of computational scene and event analysis methods. This paper describes the methods we adopted to participate in the task 2 of DCASE 2019 Challenge.

The second task of this year's challenge is *Audio tagging with noisy labels and minimal supervision* [1]. It provides public dataset [2] with baseline and aims to develop competitive audio classification systems using a small set of manually-labeled data and a larger set of noisy-labeled data.

State-of-the-art methods are based on deep neural networks, including Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Convolutional Recurrent Neural Network (CRNN). We follow this trend and use two types of neural network architectures: a CRNN and a variant of CNN (DenseNet).

[*]The first two authors contributed equally.

[†]This work was supported by the National Natural Science Foundation of China under Grant No. U1836219. The corresponding author is Wei-Qiang Zhang.

Data augmentation has been widely utilized in recent DCASE Challenges. Mixup [3] strategy was adopted by top teams [4, 5] in DCASE 2018 Challenge. Besides, a new augmentation method named SpecAugment [6] has been proposed recently. In our work, we used the combination of both methods.

Although this is a multi-label classification problem, most audio clips contain only one sound event. For this reason, we call it a "sparse multi-label classification" problem, and propose a sigmoid-softmax activation structure to deal with this problem.

In this task, how to use noisy data is the key to achieving competitive performance. We designed a staged training strategy to select the most convincing samples from noisy data and train our model using both verified data and convincing unverified data. This new training strategy will be illustrated in the following sections.

Additionally, we did some explorations about post-processing and found an effective way of score normalization.

The rest of this paper is organized as follows: we describe our methods in detail in Section 2; we present our experiments and results in Section 3; finally we conclude our work in Section 4.

2. METHODS

2.1. Feature Extraction

We used two types of acoustic features, including log-mel energies and perceptual Constant-Q transform (p-CQT). And we also used different parameters, such as frame length, hop length, frequency range, mel bins. As shown in Table 1, we used three feature configurations in total. All features are extracted using librosa [7].

Table 1: Configurations of acoustic features

	Type A	Type B	Type C
Feature	log-mel	log-mel	CQT
Window length	1764	2048	—
Hop length	882	511	512
Low Frequency	0 Hz	0 Hz	55 Hz
High Frequency	22050 Hz	16000 Hz	—
Feature dim	80	128	128
bins per octave	—	—	16

2.2. Data Preprocessing

Raw feature data needs further preprocessing before being input to neural networks. The data preprocessing procedures used in our work include sound activity detection (SAD) and data padding.

https://doi.org/10.33682/r7nr-v396

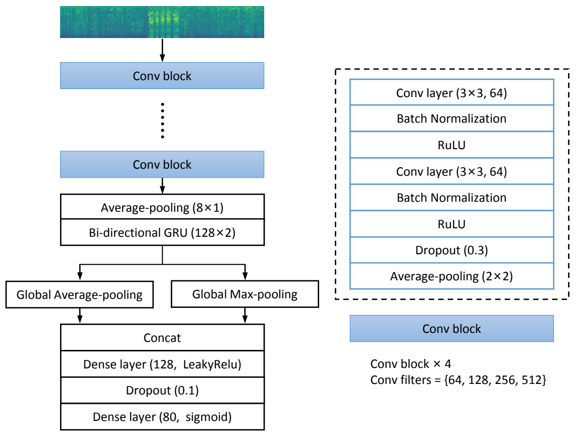


Figure 1: The architecture of CRNN. It consists of 4 convolutional blocks, a bi-GRU and two dense layers. Each convolutional block contains 2 convolutional layers. After bi-GRU, global average pooling and global max pooling operations are applied to aggregate temporal information, and the results are concatenated together before being input to dense layers.

SAD has shown powerful performance in previous work [5]. We mainly used two kinds of SAD methods: 1) We ignore the silent frames at the beginning and end of each audio. 2) We ignore the silent frames through the whole audio.

To deal with the variable lengths of acoustic features, we set a maximum padding length. All features shorter than the padding length will be repeated to the padding length. And longer features will be downsampled to align with the padding length. In our work, the padding length is set 2000. During training, we randomly select continuous 512 frames to feed into the neural network. For test, the whole 2000 frames are used to get predictions.

2.3. Data Augmentation

As mentioned above, we combined mixup [3] and SpecAugment [6] for data augmentation.

In mixup, we randomly select a pair of samples from training data. Let x_1, x_2 be the features, and y_1, y_2 be the one-hot labels respectively, the data is mixed as follows:

$$x = \lambda x_1 + (1-\lambda)x_2 \tag{1}$$

$$y = \lambda y_1 + (1-\lambda)y_2 \tag{2}$$

where the parameter λ is a random variable with Beta distribution B(0.4, 0.4).

SpecAugment is implemented by time warping, frequency masking and time masking. Detail is available in [6].

2.4. Neural Network

2.4.1. CRNN architecture

The architecture of CRNN is illustrated in Figure 1. It begins with four convolutional blocks. Each block contains two convolutional layers, followed by batch normalization [8], ReLU, dropout [9] and average pooling. Next, an average pooling is adopted on frequency axis to squeeze the frequency dimension to 1. And a bi-directional gated recurrent unit (Bi-GRU) is used to capture temporal context. Then, global max pooling and global average pooling are used on time axis to maintain various information and concatenated together. Finally, two dense layers are applied to output prediction scores for each class.

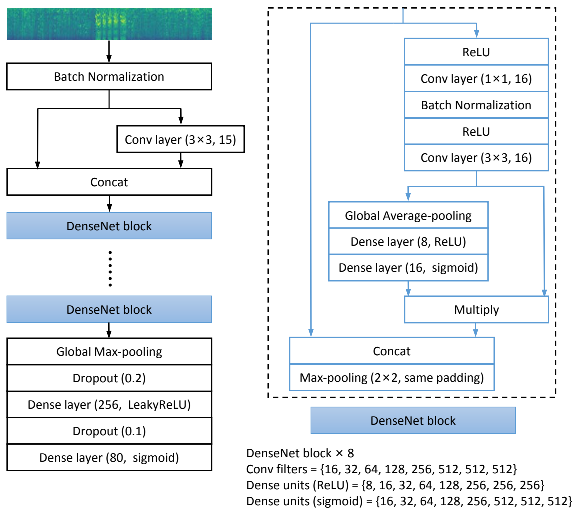

Batch Normalization
Conv layer (3×3, 15)
Concat
DenseNet block
DenseNet block
Global Max-pooling
Dropout (0.2)
Dense layer (256, LeakyReLU)
Dropout (0.1)
Dense layer (80, sigmoid)
ReLU
Conv layer (1×1, 16)
Batch Normalization
ReLU
Conv layer (3×3, 16)
Global Average-pooling
Dense layer (8, ReLU)
Dense layer (16, sigmoid)
Multiply
Concat
Max-pooling (2×2, same padding)
DenseNet block
DenseNet block × 8
Conv filters = {16, 32, 64, 128, 256, 512, 512, 512}
Dense units (ReLU) = {8, 16, 32, 64, 128, 256, 256, 256}
Dense units (sigmoid) = {16, 32, 64, 128, 256, 512, 512, 512}

Figure 2: The architecture of DenseNet. Batch normalization is applied to the input acoustic feature, followed by a convolutional layer. The input and output of this convolutional layer are concatenated along channels, followed by 8 DenseNet blocks. Then, global max pooling is applied, and 2 dense layers are utilized to output final predictions. The configuration of DenseNet block is illustrated in the dotted box.

Table 2: The number of positive labels in training dataset

#positive labels	train curated	train noisy
1	4269	16566
2	627	2558
3	69	504
4	4	141
5	0	38
6	1	4
7	0	4
average #positive labels	1.157	1.211
percentage of single label	85.9%	83.6%

2.4.2. DenseNet architecture

The architecture of DenseNet is shown in Figure 2. Our module is similar to that in [4]. In this module, the feature maps of previous layers can propagate to later layers, which can effectively alleviate the vanishing-gradient problem and encourage feature reuse. In each DenseNet block, we use Squeeze-and-Excitation Network [10], which can adaptively recalibrate channel-wise feature responses by explicitly modelling interdependencies between channels.

2.4.3. Choice of final activation

Since this is a multi-label and multi-class classification task, sigmoid is naturally the primary choice of the activation in final

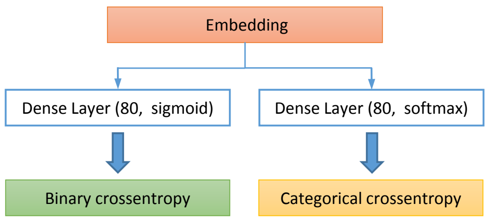


Figure 3: Combination of sigmoid and softmax activation.

layer. However, as shown in Table 2, the average number of positive labels in training dataset is very close to 1, and single-label data takes up the majority. So we call this task a "sparse multi-label classification" problem. In this condition, softmax is also a good option.

In order to combine the advantages of both sigmoid and softmax, we design a new structure named sigmoid-softmax activation. In this structure, the output embedding before the final layer will pass through two dense layers. As shown in Figure 3, one dense layer with sigmoid activation will be optimized with binary crossentropy loss, and the other dense layer with softmax activation will be optimized with categorical crossentropy loss. The outputs of both dense layers are ensembled to get final prediction.

2.5. Staged Training Strategy for Noisy Data

In this task, only a small amount of data is manually labeled, and a large quantity of data contains noisy label. Since the noisy data is not verified to have groundtruth label, we attempt to use only the most convincing noisy data. Inspired by the batch-wise loss masking in [4], we propose a staged training strategy to learn from noisy data.

To make it specific, we firstly use the verified data to train our system for several epochs. Then, we use both the verified and unverified data. However, in order to use only the most convincing noisy data, we adopt a loss masking similar to the work in [4]. The difference is that we ignore the noisy samples with the top k loss in a batch rather than set a threshold value and ignore samples with higher loss. Finally, after training for more epochs, we abandon the noisy data and finetune our model with only the verified data. Our staged training strategy has made huge improvements according to our experiments.

2.6. Score Normalization

For inference, we use score normalization strategy for further improvements. Let $x_{i,j}$ be the prediction score for the i-th class in the j-th sample. We normalize the prediction scores for each class. The normalization procedure goes as follows:

$$\overline{x}_i = \frac{\sum_{j=1}^{N} x_{i,j}}{N}, \tag{3}$$

$$\hat{x}_{i,j} = \frac{x_{i,j} - \overline{x}_i}{\sqrt{\frac{1}{N}\sum_{j=1}^{N}\left(x_{i,j} - \overline{x}_i\right)^2 + \varepsilon}} \tag{4}$$

$$\widetilde{x}_{i,j} = \frac{\hat{x}_{i,j} - \min_j \hat{x}_{i,j}}{\max_j \hat{x}_{i,j} - \min_j \hat{x}_{i,j} + \varepsilon} \tag{5}$$

where N is the total number of samples in evaluation dataset, ε is a sufficiently small value to avoid division by zero. For each class in evaluation dataset, we normalize the prediction scores to zero mean and unit variance. Then, we set min and max zoom to keep the scores between 0 and 1. According to experimental results, score normalization can raise the evaluation metric by approximately 0.002 on average in cross-validation and 0.007 in private test data.

3. EXPERIMENTS

3.1. Experiment Setup

Adam optimizer [11] is used for gradient based optimization. The learning rate is 0.001 and batch size is 64. We split our training dataset into four folds. Then we train four models using any three folds for training and the other fold for validation.

As for the staged training, we design a data generator to generate different proportions of training data during different stages. In the first stage, all data comes from curated dataset. In the second stage, the proportion of curated dataset is equal to noisy dataset. In the third stage, only curated dataset is used. In the second stage, the top k samples with the highest loss on noisy dataset would be masked. In our experiments, k is 10.

For CRNN architecture, the first stage runs for 8k iterations, the second stage runs for 12k iterations, and the third stage runs for 3k iterations. For DenseNet architecture, the first stage runs for 5k iterations, the second stage runs for 8k iterations, and the third stage runs for 2k iterations. The models with the best validation performance on each fold are selected.

3.2. Evaluation Metric

The primary competition metric is label-weighted label-ranking average precision (lwlrap). This measures the average precision of retrieving a ranked list of relevant labels for each test clip (i.e., the system ranks all the available labels, then the precisions of the ranked lists down to each true label are averaged). LRAP is calculated as follows, and lwlrap is the macro-average of per-class LRAP. [12]

$$\mathrm{LRAP}(y, \hat{f}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} \frac{1}{\|y_i\|_0} \sum_{j: y_{ij}=1} \frac{|\mathcal{L}_{ij}|}{\mathrm{rank}_{ij}} \tag{6}$$

where $\mathcal{L}_{ij} = \left\{k : y_{ik} = 1, \hat{f}_{ik} \geq \hat{f}_{ij}\right\}$, $\mathrm{rank}_{ij} = \left|\left\{k : \hat{f}_{ik} \geq \hat{f}_{ij}\right\}\right|$, $|\cdot|$ computes the cardinality of the set, and $\|\cdot\|_0$ computes the number of nonzero elements in a vector.

3.3. Model Ensemble and Submissions

Model ensemble is successful in boosting the system's performance. We ensemble our models using geometric average as follows:

$$y_{\text{ensemble}} = \exp \frac{1}{N} \sum_{n} w_n \log y_n \tag{7}$$

where N is the number of subsystems, y_n is the output score of each subsystem, and w_n is the weight coefficient for each subsystem.

We submitted two prediction results using different weights:

Table 3: Lwlrap scores of multiple configurations, on both cross-fold validation and private evaluation dataset. The score on cross-fold validation dataset are the average of scores on four folds. "sig-soft" is the abbreviation of sigmoid-softmax activation structure.

	Cross-fold Validation			Private Evaluation		
	Type A	Type B	Type C	Type A	Type B	Type C
CRNN sigmoid	0.8512	0.8547	0.8413	0.7253	0.7290	0.7140
CRNN softmax	0.8437	0.8479	0.8362	0.7208	0.7176	0.7094
CRNN sig-soft	**0.8561**	**0.8613**	**0.8502**	**0.7388**	**0.7334**	**0.7203**
DenseNet sigmoid	0.8219	0.8357	0.8321	0.7053	0.7017	0.6923
DenseNet softmax	0.8143	0.8235	0.8249	0.7014	0.7043	0.6837
DenseNet sig-soft	0.8246	0.8378	0.8412	0.7146	0.7141	0.7074

Table 4: The performance of systems using curated data only and systems using both curated and noisy data. We use CRNN architecture with three different activation functions on both cross-fold validation and private evaluation dataset. The relative improvement of adding noisy data using staged training strategy is shown in the parenthesis. Type A feature is used in experiments.

		Curated data	Curated and noisy data
Cross-fold Validation	sigmoid	0.8417	0.8512 (1.13% $\uparrow$)
	softmax	0.8278	0.8437 (1.92% $\uparrow$)
	sig-soft	0.8376	0.8561 (2.21% $\uparrow$)
Private Evaluation	sigmoid	0.7155	0.7253 (1.37% $\uparrow$)
	softmax	0.7142	0.7208 (0.92% $\uparrow$)
	sig-soft	0.7207	0.7388 (2.51% $\uparrow$)

1) *Zhang_THU_task2_1.output.csv*: achieved our highest lwlrap score of 0.742 on public leaderboard in *Kaggle*.

2) *Zhang_THU_task2_2.output.csv*: achieved our highest local lwlrap scores in each cross-fold validation, with a lwlrap score of 0.739 on public leaderboard in *Kaggle*.

3.4. Experimental Results

In order to investigate more about proposed methods, we conducted further experiments on private evaluation dataset after submitting to DCASE Challenge. Our experiments were conducted on two neural network architectures (CRNN and DenseNet), three activation functions (sigmoid, softmax, and sigmoid-softmax activation structure), and three acoustic features (Type A, B, C as mentioned in subsection 2.1).

The lwlrap scores of multiple configurations, on both cross-fold validation and private evaluation dataset, are shown in Table 3. The score on cross-fold validation dataset is the average of scores on four folds. As shown in Table 3, CRNN architecture with sigmoid-softmax activation structure can achieve the best performance in all types of features on both validation and evaluation dataset. Besides, sigmoid-softmax activation structure can outperform single sigmoid or softmax activation in all feature types and neural networks. We can draw a conclusion that proposed sigmoid-softmax can demonstrate remarkable performance in "sparse multi-label classification" problems.

To examine the performance of proposed staged training strategy, we also conducted some comparative experiments using curated data only. We compare the performance of systems using curated data only and systems using both curated and noisy data in Table 4. We use Type A feature as acoustic feature and CRNN as classifier. Three types of activations are applied to verify the effects of staged training strategy. The results show that proposed method can make good use of noisy data to enhance the classification and generalization capability of our models.

Table 5: Comparison of several systems, on both public leaderboard and private leaderboard.

	Lwlrap (public LB)	Lwlrap (private LB)
Ensemble-1	**0.7423**	0.7575
Ensemble-2	0.7392	**0.7577**
Ensemble-3	***	0.7508
Ensemble-4	***	0.7500
OUmed	**0.7474**	**0.7579**
Ebbers	0.7305	0.7552

Furthermore, we compare the following systems in Table 5:

- Ensemble-1: the architecture generating aforementioned *Zhang_THU_task2_1.output.csv*.
- Ensemble-2: the architecture generating aforementioned *Zhang_THU_task2_2.output.csv*.
- Ensemble-3: the same architecture with Ensemble-1, but without score normalization processing.
- Ensemble-4: the same architecture with Ensemble-2, but without score normalization processing.
- OUmed: the 1st place in DCASE Challenge. [13]
- Ebbers: the 3rd place in DCASE Challenge. [14]

It can be concluded that proposed score normalization strategy can increase lwlrap score by approximately 0.007. Compared with other teams, our system is also very competitive.

4. CONCLUSION

In this paper, we describe our methods and techniques used in the task 2 of DCASE 2019 Challenge. We adopted mixup and SpecAugment for data augmentation and applied two types of deep learning model including CRNN and DenseNet. Besides, a staged training strategy is applied to learn from both curated and noisy data and a sigmoid-softmax activation structure is proposed to solve sparse multi-label classification problems. Using model ensemble and score normalization strategies, our final system ranked the 2nd place in DCASE 2019 Challenge.

5. REFERENCES

[1] E. Fonseca, M. Plakal, F. Font, D. P. Ellis, and X. Serra, "Audio tagging with noisy labels and minimal supervision," *arXiv preprint arXiv:1906.02975*, 2019.

[2] E. Fonseca, J. Pons Puig, X. Favory, F. Font Corbera, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound datasets: a platform for the creation of open audio datasets," in *Proceedings of the 18th ISMIR Conference; 2017 oct 23-27; Suzhou, China.[Canada]: International Society for Music Information Retrieval; 2017. p. 486-93.* International Society for Music Information Retrieval (ISMIR), 2017.

[3] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[4] I.-Y. Jeong and H. Lim, "Audio tagging system for DCASE 2018: focusing on label noise, data augmentation and its efficient learning," DCASE2018 Challenge, Tech. Rep., 2018.

[5] T. Iqbal, Q. Kong, M. D. Plumbley, and W. Wang, "Stacked convolutional neural networks for general-purpose audio tagging," *Tech. Rep., DCASE Challenge*, 2018.

[6] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[7] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, 2015, pp. 18–25.

[8] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of The 32nd International Conference on Machine Learning*, 2015, pp. 448–456.

[9] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[10] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[11] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[12] https://www.kaggle.com/c/freesound-audio-tagging-2019/overview/evaluation.

[13] O. Akiyama and J. Sato, "Multitask learning and semi-supervised learning with noisy data for audio tagging," DCASE2019 Challenge, Tech. Rep., 2019.

[14] J. Ebbers and R. Haeb-Umbach, "Convolutional recurrent neural network and data augmentation for audio tagging with noisy labels and minimal supervision," DCASE2019 Challenge, Tech. Rep., 2019.

ACOUSTIC SCENE CLASSIFICATION USING DEEP LEARNING-BASED ENSEMBLE AVERAGING

Jonathan Huang[1], *Hong Lu*[1], *Paulo Lopez-Meyer*[2],
Hector A. Cordourier Maruri[2], *Juan A. del Hoyo Ontiveros*[2]

[1] Intel Corp, Intel Labs, 2200 Mission College Blvd.,Santa Clara, CA 95054, USA,
{jonathan.huang, hong.lu}@intel.com
[2] Intel Corp, Intel Labs, Av. Del Bosque 1001, Zapopan, JAL, 45019, Mexico,
{paulo.lopez.meyer, hector.a.cordourier.maruri, juan.antonio.del.hoyo.ontiveros}@intel.com

ABSTRACT

In our submission to the DCASE 2019 Task 1a, we have explored the use of four different deep learning based neural networks architectures: Vgg12, ResNet50, AclNet, and AclSincNet. In order to improve performance, these four network architectures were pre-trained with Audioset data, and then fine-tuned over the development set for the task. The outputs produced by these networks, due to the diversity of feature front-end and of architecture differences, proved to be complementary when fused together. The ensemble of these models' outputs improved from best single model accuracy of 77.9% to 83.0% on the validation set, trained with the challenge default's development split. For the challenge's evaluation set, our best ensemble resulted in 81.3% of classification accuracy.

Index Terms**—** DCASE, Acoustic Scene Classification, Deep Learning, Neural Networks, Transfer Learning, End-to-End architectures, Ensemble Averaging.

1. INTRODUCTION

In the Detection and Classification of Acoustic Scenes and Events 2019 challenge (DCASE 2019), acoustic data were provided to solve different audio related tasks [1]. Task 1 refers to the challenge of building a model to classify different recordings into predefined classes corresponding to recordings of different environment settings in several large European cities.

Following the guidelines provided by the challenge in the Task 1 subtask *a* (Task 1a), we experimented with four different deep learning (DL) neural network architectures: Vgg12, ResNet50, AclNet, and AclSincNet (Figure 1). The Vgg12 and ResNet50 architectures are adaptations of well-known computer vision CNNs adapted to the audio classification task, with 12 and 50 layers, respectively; they both take Mel-filterbank of 64 spectral dimensions as input features. On the other hand, AclNet is an end-to-end (e2e) architecture that takes raw audio input into two layers of 1D CNNs, followed by a VGG-like 2D CNN. AclSincNet is similarly an e2e approach, with the difference in the 1D convolution layers; the 1D convolution layers are essentially combinations of sinc functions, or equivalently band-pass filters, whose cut-off frequencies are the learnable parameters in the model training process.

2. METHODOLOGY

In this section, we describe in detail the experimentation we followed for our submission to the DCASE 2019 Task1a.

2.1. Data Processing

The DCASE 2019 Task 1a dataset consists of 10-second audio recordings obtained at 10 different acoustic scenes: airport, indoor shopping mall, metro station, pedestrian sreet, public square, street with medium level of traffic, traveling by tram, traveling by bus, traveling by and underground metro, and urban park , recorded at 12 major European cities [2].

The challenge provides as part of this task a 1-fold arrangement for development, i.e. training and validation data splits. In addition to the 1-fold defined, we also experimented with 5-fold random splits over all available data (training/validation), and city held-out validation sets resulting in 10 training/validation splits (only data from 10 cities were available in the development set). Additionally, through the development stage we used Google Audioset data [3] to pre-train all of our implemented DL architectures.

For the development of e2e DL architectures, the two binaural channels were averaged into a single one, and the resulting signal was down sampled from its original 48 kHz to 16 kHz. For the development of spectral based DL architectures, audio data from each channel were processed to generate Mel-filterbank representations with 64 filter bands over a time window of 25 milliseconds and overlaps of 10 milliseconds, resulting in two Log-Mel filterbank channels (Figure 1). Because our early experiments of using multiple channels did not yield improvement over single channel, we opted to use a randomly selected channel in the training process, and channel 0 in the testing.

2.2. Neural Network Architectures

2.2.1. AclNet

AclNet is an e2e CNN architecture, which takes a raw time-domain input waveform as opposed to the more popular technique of using spectral features like Mel-filterbank or Mel-frequency cepstral coefficients (MFCC). One of the advantages of e2e architectures like this is that the front-end feature makes no assumptions of the frequency response. Its feature representation is learned in a data-driven manner, thus its features are optimized for the task at hand as long as there is sufficient training data. We followed the specific settings corresponding to the AclNet work described in [4], with a width multplier of 1.0 and conventional convolutions.

The AclNet architecture we developed for this work was pre-trained with Audioset that resulted in 527 outputs, which in turn were used as embeddings to train a fully-connected layer classifier with ReLU activation functions in a transfer learning manner. Raw

https://doi.org/10.33682/8rd2-g787

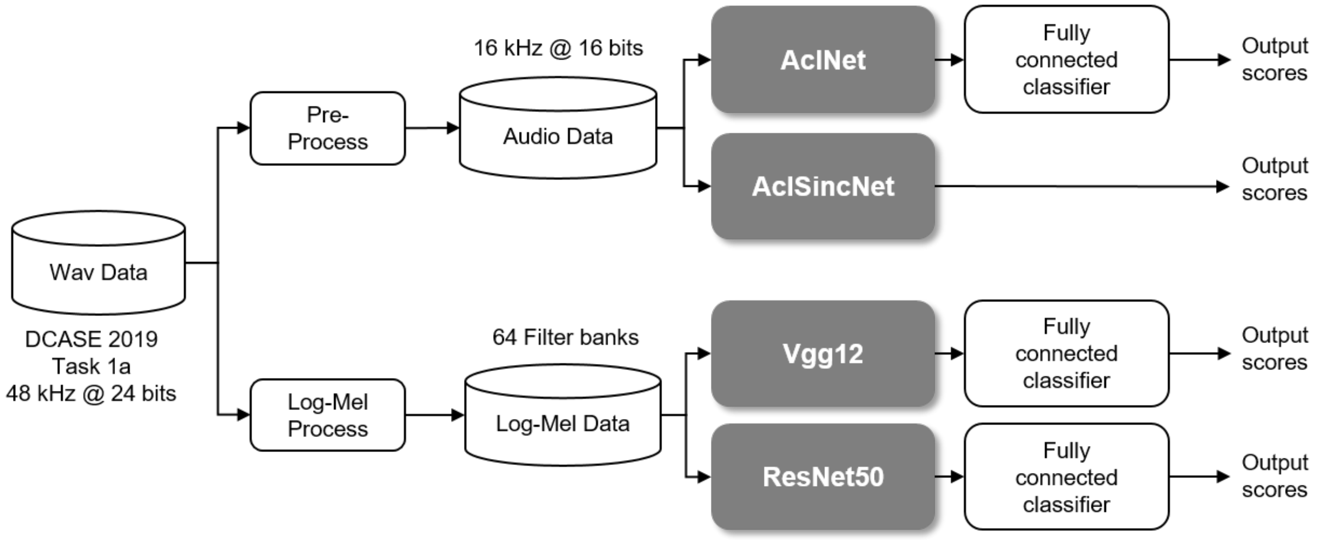


Figure 1: Development of our proposed deep learning architectures for audio scene classification in the DCASE 2019 Task 1a.

audio data at 16 kHz form the DCASE 2019 challenge was fed to the pre-trained AclNet, and embeddings were used to train the classifier.

We performed a search for the optimal parameters of this DL model. We experimented with different values and configurations, and ended with the best performing model when using the hyper parameters described in Table 2; cosine annealing for the learning rate schedule, and the use of mixup for data augmentation [5] were considered, but it was observed that they did not improve the classification performance. During the training, 1-second of audio was randomly selected from mini-batches of 64 training clips. At test time, we run the inference on 1-second non-overlapping consecutive audio segments, and then average the outputs over the length of the test audio. The experimental results obtained by this e2e architecture can be seen in Table 3.

2.2.2. *AclSincNet*

The AclSincNet architecture consists of two building blocks of the network: a low-level Spectral like features (SLF), and a high-level features (HLF).

The SLF is a set of features designed to be similar to the spectral features used in conventional audio processing. It can be viewed as a replacement of the spectral feature, but it is fully differentiable and can be trained with backward propagation. The front-end is inspired by SincNet [6]. We used the same setup as suggested in the original work but with a stride length of 16 milliseconds; the authors of SincNet treat the output of the SincNet layer as a replacement of the FFT filter bank, and feed it directly into the subsequent CNN layer. We took a different route, aiming to replicate the output of Mel-filterbank calculation more closely. We first compute the square of the output before average pooling over multiple time steps, and then follow up by the log operation to make the filterbank output less sensitive to amplitude variations. With the time-domain waveform as input, the SLF layer produces an output of 256 channels at feature frame rate of 10 milliseconds after the average pool layer. In our setup, a 1.280-second input produces an output tensor of dimension (256, 1, 128).

Taking the output from the SLF block, the high level feature

Table 1: SLF Architecture used in AclSincNet

Layer	**Description**
Sincnet1D	kernel size 251, stride length 16
BatchNorm	
Calculate energy spectrum	x = x^2
Average pooling	100ms window with 50ms stride
Clamp the min	x = x.clamp(min=1e-12)
Calculate log()	x = log(x)

layer treats it as a 2D image (e.g. a 256x128 image for the 1.28-second input) and applies standard VGG-like 2D CNN. The architecture of the HLF is similar to typical image classification CNNs. We experimented with a number of different architectures, and found that a VGG-like architecture provides good classification performance and well-understood building blocks. In our case, each of the conv layers are a standard building block that comprises a 2D convolution layer, batch normalization, and PReLU activation.

We did not use fully connected layers as in standard VGG; instead, we simply apply average pooling to output the scores. The final layers of the AclSincNet is a 1x1 convolution that reduces the number of channels to the number of classes (10 classes in the DCASE 2019 challenge). Before the input to the 1x1 convolution layer, we add a dropout layer for regularization. We found a dropout probability of 0.9 to work well on this task. Each of the 10 channels are then average pooled and output directly after SoftMax. The advantage of these final two layers is that our architecture can incorporate arbitrary length inputs for both training and testing, without any need to modify the number of hidden units of a the fully connected layers.

We pre-trained this model with AudioSet [3] and then fine tuned it with the DCASE 2019 Task 1a data. During fine-tuning, we use 6-seconds audio data (random crop from each of the 10-second sample) for training, and 10-second data (the complete clip) for testing. We experimented with different number of layers, number of channels, and kernel size on the data set. While the total number of the parameter is rather big (Table 3), we noticed that the network scales quite well when we shrink the layer, channel, and kernel size. We observed experimentally that the accuracy only drops 1 to 2% when a network with 18M parameters is used. However, we decided to stick to the larger network for this challenge.

2.2.3. *Vgg12*

The Vgg12 model is an adaptation of the well-known VGG architecture [7]. It has a total of 12 convolutional layers, with the first one having 64 channel output, and the last one 512 outputs. At the output of each conv layer, we apply batch normalization followed by ReLU activation. Through the conv layers, there are 5 max-pool layers with kernel size of 2. A key architectural difference is that the network is designed for variable input size (e.g. 64 spectral dimensions and arbitrary number of time steps). The output of the last convolutional layer is average pooled, to always produce a vector length of 512 values. This vector is then followed up by a fully connected layer to produce the 10-class output defined by the challenge.

During the training phase, 5-seconds of audio are randomly selected from the training clip. At test time, we run the inference on 1-second non-overlapping segments, and then average the outputs over the length of the test audio.

2.2.4. *ResNet50*

Our audio ResNet50 has exactly the same convolutional layers as the architecture in the original ResNet paper [8]. Again, we made the same adaptations as in Vgg12 described in 2.2.3, to take a variable length input Mel-filterbank spectra into a single 10-class output. We also used the training and testing sequence selection scheme applied for the Vgg12.

During the training, we used SpecAugment [9] as a data augmentation scheme. This data augmentation works by masking out random time and spectral bands of randomly selected width. We opted to mask only spectral bands, at random positions with width uniformly sampled from [0, 20] Mel bands. We found that this augmentation gave a 1% improvement over the same network without SpecAugment.

2.3. Training Strategy

In addition to the default train/validation split provided by DCASE 2019 Task 1a, we used two strategies to train our models for score averaging: 5-fold cross-validation and leave-one-city-out cross validation. In both cases, we merged the development and validation data set together and re-split all the labelled data set with the above two strategies to train models accordingly.

With the 5-fold cross-validation method, all the labelled data were split into 5 folds with random shuffle, i.e. 4/5 of the data set is used for training and 1/5 of the data set is used as validation set. Five models are trained and their scores are averaged as the final score. i.e. the final output is essentially the ensemble average of 5 individually trained models.

Similarly, the leave-one-city-out method was done in the same way with a different split methodology. Instead of random split, the split is done by city. i.e. 9 cities are used for training and 1 city is held out and used as the validation set. Therefore, 10 models are trained and their scores are averaged as the final output.

All four architectures were trained with the Adam optimizer. The training hyper parameters (learning rate LR, LR schedule, number of epochs E, drop out rate DO, and weight decay WD) of each of the architectures are listed in Table 2. During the fine-tuning process, we kept the part of the corresponding Audioset pre-trained network with a LR value that is 1/10th of the LR as the rest of the network. The validation set were used for model selection, i.e. the best performing model on the validation set were saved and used for inference.

Table 2: Training setup values for the four deep learning architectures proposed. The values displayed are the learning rate, learning rate schedule, number of epochs, weight decay, and drop out rate, respectively.

Architecture	LR	Schedule	E	WD	DO
AclNet	1e-4	No schedule	330	2e-4	0.2
AcLSincNet	1e-3	Cosine annealing	50	1e-6	0.9
Vgg12	1e-3	Cosine annealing	40	1e-5	0.8
ResNet50	1e-3	Cosine annealing	40	1e-5	0.5

Table 3: Classification results over the validation set obtained by the individual deep learning based neural network architectures explored in this work.

Architecture	Params	Accuracy
AclNet	19.3M	0.7481
AclSincNet	52.2M	0.7608
Vgg12	12.8M	0.7744
ResNet50	24.5M	0.7787

2.4. Ensemble Averaging

In order to reduce individual variance of each of the developed DL models described in the previous subsections, we applied ensemble averaging technique, which is one of the simplest ensemble learning methodologies used in machine learning to improve the prediction performance [10]. This approach consists on the averaging of the prediction scores obtained by different models, as seen in Figure 2.

By combining the prediction scores from different DL models that performed above the reported challenge baseline, the intention is to add a bias that counters the variance of a single trained model. Having a diversity of DL models helps to achieve this intention. In our experiments, we have extensively explored different combination sets of our DL models in order to find the ones that better generalize over the validation data set. Experimental results obtained for some of the most obvious combinations are shown in the next section.

3. RESULTS

The experimental results obtained for our developed DL models over the validation data set are shown in Table 3. These are the best experimental validation accuracy results achieved by our individually trained DL models at the time of the DCASE 2019 submission deadline. For each one of the developed models, the number of trainable parameters is listed also in Table 3 to present an idea of the size of the DL architectures used.

In Table 4, the results from the best combinations of two, three, and four DL models that were obtained through ensemble averaging of their output scores. All outputs were defined as softmax scores in order to have compatible values for averaging across the ensemble. It can be observed how the experimental results over the validation set yield into higher accuracy (Table 4) when compared to individ-

Table 4: Classification results over the validation set obtained from the ensemble averaging combination of two, three, and four deep learning based neural networks architectures.

Architecture	**Params**	**Accuracy**
AclSincNet+ResNet50	76.8M	0.8127
Vgg12+AclSincNet+ResNet50	89.7M	0.8184
AclNet+Vgg12+AclSincNet+ResNet50	109.0M	0.8301

ual models results (Table 3). These results support the idea that utilizing the combination of predictions from different DL models results in a reduction of variance, and in more accurate predictions [10].

For our four allowed submissions to DCASE 2019 Task 1a, we experimented with the DL architectures described above, developed with different types of data splits, and combining their output predictions through ensemble averaging. In the next section we provide a detailed description of the combinations used as part of our complete final submission.

4. SUBMISSIONS TO DCASE 2019 TASK 1A

Our four submissions to the DCASE 2019 Task 1a consists of the ensemble averaging of different predicted scores from different DL architectures. Below is a detailed description list of the ensemble models used to generate the submission labels on the evaluation data set provided:

1. Ensemble averaging obtained from a combination of 40 individually trained DL architectures: 1 Vgg12, trained with the default data split defined by the challenge; 10 Vgg12, each trained with 1 of the 10 leave-one-city-out splits; 5 AclSincNet, trained with 5 different random splits; 10 AclSincNet, each trained with 1 of the 10 leave-one-city-out splits; 3 Resnet50, each trained with three different data splits; 10 ResNet50, each trained with 1 of the leave-one-city-out splits; and 1 AclNet, trained with the default data split defined by the challenge. Considering each one of the 40 DL architectures combined, the total number of trainable parameters resulted in 1,264.4M.
2. Ensemble averaging obtained from a combination of 31 individually trained DL architectures: 10 Vgg12, each trained with 1 of the 10 leave-one-city-out splits; 10 AclSincNet, each trained with 1 of the 10 leave-one-city-out splits; 10 ResNet50, each trained with 1 of the 10 leave-one-city-out splits; and 1 Resnet50, trained with 1 data split. Considering each one of the 31 DL architectures combined, the total number of trainable parameters resulted in 921.7M.
3. Ensemble averaging obtained from a combination of 26 individually trained DL architectures: 10 Vgg12, each trained with 1 of the 10 leave-one-city-out splits; 5 AclSincNet, trained with 5 different random splits; 10 ResNet50, each trained with 1 of the 10 leave-one-city-out splits; and 1 Resnet50, trained with 1 data split. Considering each one of the 26 DL architectures combined, the total number of trainable parameters resulted in 798.6M.
4. Ensemble averaging obtained from a combination of 20 individually trained DL architectures: 10 Vgg12, each trained with 1 of the 10 leave-one-city-out splits; and 10 ResNet50, each trained with 1 of the 10 leave-one-city-out splits. Considering each one of the 20 DL architectures combined, the total number of trainable parameters resulted in 374.7M.

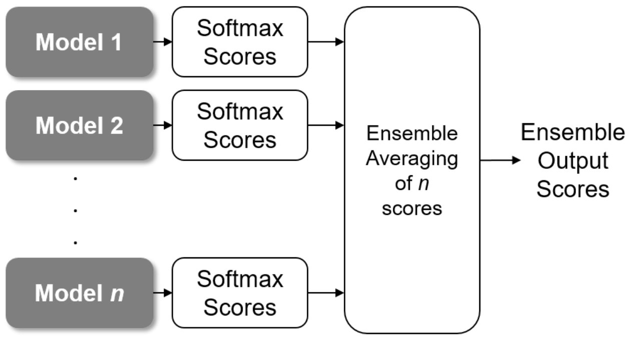


Figure 2: Ensemble averaging of *n* independently trained deep learning models.

Table 5: Classification results over the evaluation set obtained from four different ensemble averaging combinations of deep learning architectures, submitted to the DCASE 2019 challenge Task 1a.

Architecture	**Params**	**Accuracy**	**Challenge Rank**
Ensemble 1	1,264.4M	0.8050	15
Ensemble 2	921.7M	0.8110	12
Ensemble 3	798.6M	0.8130	10
Ensemble 4	374.7M	0.7950	22

The final Task 1a challenge results obtained from these four ensembles over the evaluation set are shown in Table 5. The best result obtained of 81.3% ranked 10th across all submissions, and 5th by team submissions. Our submission consisted on simple ensemble averaging; part of our ongoing efforts consists of exploring other ensemble methodologies, e.g. stacking, to increase the predictive force of the classifiers.

5. CONCLUSIONS

Starting from individually trained DL models, we were able to achieve above the baseline results as reported in the DCASE 2019 Task 1a challenge. From these, we were able to increase the performance of the audio scene classification by combining the prediction scores of different DL models through ensemble averaging. By doing this ensemble, we obtained significantly higher classification results over the validation set than the ones obtained by individual DL models, i.e. 83.0% *Vs* 77.9% , respectively. Our best ensemble model resulted in a 81.3% classification accuracy over the evaluation set provided by the challenge.

6. REFERENCES

[1] Detection and Classification of Acoustic Scenes and Events Challenge 2019. `http://dcase.community/challenge2019`.

[2] Heittola, Toni; Mesaros, Annamari; Virtanen, Tuomas. ”TAU Urban Acoustic Scenes 2019, Development dataset”, *Zenodo*, 2019. `http://doi.org/10.5281/zenodo.2589280`

[3] Gemmeke, Joft F.; Ellis, Daniel P. W.; Freedman, Dylan; Jansen, Aren; Lawrence, Wade; Moore, R. Channing;

Plakal, Manoj; Ritter, Marvin. 'Audio Set: and Ontology and human-Labeld Dataset for Audio Events", *ICASSP*, 2017. `https://ieeexplore.ieee.org/abstract/document/7952261`.

[4] Huang, Jonathan; Alvarado-Leanos, Juan. "AclNet: efficient end-to-end audio classification CNN", *CoRR*, 2018. `http://arxiv.org/abs/1811.06669`

[5] Zhang, Hongyi; Cisse, Moustapha; Dauphin, Yann N.; Lopez-Paz, David. "mixup: Beyond Empirical Risk Minimization", *arXiv:1710.09412 [cs.LG]*, 2018. `https://arxiv.org/abs/1710.09412`

[6] Ravanelli, Mirco; Bengio, Yoshua. "Speaker Recognition from Raw Waveform with SincNet", *CVPR*, 2018. `https://arxiv.org/abs/1808.00158`

[7] Simonyan, Karen; Zisserman, Andrew. "Very Deep Convolutional Networks for Large-Scale Image Recognition", *ICLR*, 2015. `https://arxiv.org/abs/1409.1556`

[8] He, Kaiming; Zhang, Xiagyu; Ren, Shaoqing; Sun, Jian. "Deep Residual Learning for Image Recognition", *CVPR*, 2019. `https://arxiv.org/abs/1512.03385`

[9] Park, Daniel S.; Chan, William; Zhang, Yu; Chiu, Chung-Cheng; Zoph, Barret; Cubuk, Ekin D.; Le, Quoc V.. "Specaugment: A simple data augmentation method for automatic speech recognition", *arXiv:1904.08779 [eess.AS]*, 2019. `https://arxiv.org/abs/1904.08779`

[10] Brownlee, Jason. "Ensemble Learning Methods for Deep Learning Neural Networks", *Machine Learning Mastery!*, 2018. `https://machinelearningmastery.com/ensemble-methods-for-deep-learning-neural-networks`

NEURAL AUDIO CAPTIONING BASED ON CONDITIONAL SEQUENCE-TO-SEQUENCE MODEL

Shota Ikawa[1], *Kunio Kashino*[1,2]

[1] Graduate School of Information Science and Technology, The University of Tokyo, Japan
[2] NTT Communication Science Laboratories, NTT Corporation, Japan

ABSTRACT

We propose an audio captioning system that describes non-speech audio signals in the form of natural language. Unlike existing systems, this system can generate a sentence describing sounds, rather than an object label or onomatopoeia. This allows the description to include more information, such as how the sound is heard and how the tone or volume changes over time, and can accommodate unknown sounds. A major problem in realizing this capability is that the validity of the description depends not only on the sound itself but also on the situation or context. To address this problem, a conditional sequence-to-sequence model is proposed. In this model, a parameter called "specificity" is introduced as a condition to control the amount of information contained in the output text and generate an appropriate description. Experiments show that the model works effectively.

Index Terms— audio captioning, unknown sounds, sequence-to-sequence model, cross-modal embedding

1. INTRODUCTION

Sound plays an important role in our daily life. It helps us to understand the events around us. In the realm of computational auditory scene analysis, the major topics have been sound source separation, acoustic event detection, and its classification [1]. For example, studies on environmental sounds include the detection or classification of acoustic events [2, 3], acoustic scene classification [4], and abnormal sound detection [5, 6]. However, little work has been done regarding detailed description of sounds.

Against this background, here we address audio captioning for non-speech audio signals. Audio captioning here means generating texts describing sounds given an audio signal as an input. Such captions can include more information than just an acoustic event label can, such as how the sound is heard and how the tone or volume changes over time.

An audio caption is a way to visualize acoustic information so that we can understand what is happening at a glance, even without actually hearing the sound. Therefore, it will be useful for multimedia content search, sound effect search, abnormality search, and closed captioning systems that can describe non-speech sounds. To the best of our knowledge, no work has been reported regarding automatic audio captioning systems that can generate a sound description in the form of a full sentence.

This paper is organized as follows. Section 2 details the audio captioning problem. Section 3 describes the proposed audio captioning models: the basic model and the conditional model. Section 4 explains the experimental results, which show the effectiveness of the proposed model. Section 5 concludes the paper.

2. PROBLEM OF AUDIO CAPTIONING

2.1. Related Works

Recently, an onomatopoeia generation system has been proposed [7, 8]. Here, onomatopoeia means a word or a sequence of phonemes that directly imitates a sound. Based on an encoder-decoder model, the system produces valid onomatopoeias for various input sounds. Onomatopoeia generation can be viewed as a kind of natural language generation for sounds. However, an audio caption is a sequence of words rather than phonemes, and the correspondence to the input sound is highly indirect. Whether such an indirect sequence conversion is possible or not has been an open problem.

Another related task is image captioning. Compared to object recognition, image captioning produces not only a list of the object labels contained in an image but also sentences that may include their attributes or the relationships among them. Recently, systems based on the encoder-decoder model [9, 10] have achieved reasonably good accuracy [11, 12]. In those studies, conditional neural networks (CNN) pre-trained for an image classification task were employed as the encoder, and the recurrent language model (RLM) [13] was used for caption generation based on a fixed size vector. Video captioning has also been addressed, and the long short-term memory (LSTM) was shown to effectively deal with an input with variable length [14].

However, information contained in audio signals can be much more ambiguous than that in images. In fact, it is often difficult even for humans to accurately recognize the objects in an audio signal. Moreover, how to decide the best description is not obvious for given audio because the validity of the description generally depends on the situation or context as well as the sound itself. For example, a short warning may be more appropriate than a long description and vice versa. It is important to note that such problems particularly come to light in the audio captioning task.

2.2. Specificity Conditioning

To deal with the avobe-mentioned nature of the audio captioning problem, we introduce a specificity measure of the output text based on the amount of information that the text carries.

Let p_w be the probability of appearance of a word w in a language. The amount of information carried by a word w is defined as a negative logarithm of its probability:

$$I_w \equiv -\log p_w \tag{1}$$

Given an arbitrary natural language corpus, or a dataset of audio captions, we can estimate p_w by $p_w = N_w/N$, where N_w is the

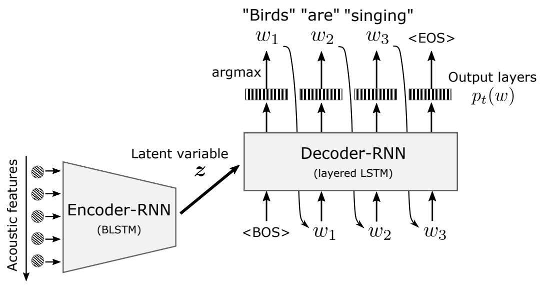


Figure 1: Block diagram of SCG.

number of appearances of w, and N is the total number of words in a language corpus or training dataset.

We consider I_w as specificity of w, and define I_s, the specificity of an audio caption s consisting of words $w_1, w_2, \ldots, w_n$, by the sum of the information values with respect to the words in $\boldsymbol{s}$:

$$I_{\boldsymbol{s}} \equiv \sum_{t=1}^{n} I_{w_t} \tag{2}$$

Obviously, $I_{\boldsymbol{s}}$ becomes high when infrequent words are contained in the caption or the caption is long in terms of the number of words.

In the audio captioning, more (or less) specificity is not always better, and therefore, the ability to control the text specificity is essential in generating a valid output text. In the following section, we first propose a caption generator based on a plain encoder-decoder model, and then we extend it to a conditional encoder-decoder model where the specificity is treated as a condition for caption generation.

3. PROPOSED MODEL

3.1. Sequence-to-Sequence Caption Generator

Figure 1 shows the audio caption generator with the plain sequence-to-sequence caption generator (SCG).

A series of acoustic features $\boldsymbol{x}$ is input to the encoder consisting of recurrent neural network (RNN) and embedded within a fixed length vector $\boldsymbol{z}$, which is a latent variable that serves as latent features of the input acoustic signal. Then, the decoder is initialized based on the derived latent variable. The decoder serves as an RLM, which calculates estimated probabilities of every word in each step and chooses the word with the highest probability w_t as input for the next step. The generated audio caption $\hat{\boldsymbol{s}} = (w_1, w_2, \ldots)$ is the series of the chosen words. The input in the first step is “BOS (beginning of the sentence)”, and the generation of the audio caption finishes in a step when “EOS (end of the sentence)” is chosen.

The SCG can be viewed as an approximation of the generative model for the following optimal audio caption $\bar{s}$:

$$\bar{\boldsymbol{s}} = \arg\max_{\boldsymbol{s}} p(\boldsymbol{s} \mid \boldsymbol{z}), \quad z = f(\boldsymbol{x}), \tag{3}$$

where f is a mapping to derive latent variables from acoustic signals and corresponds to the encoder in the model. $p(\boldsymbol{s} \mid \boldsymbol{z})$ is a probability distribution in which each audio caption is generated when the latent variable is given. The decoder is expected to generate audio captions with the highest probability.

Pairs comprising an acoustic signal and audio caption for the signal are used for learning this model. Given an acoustic signal as input, the model calculates cross entropy between the output layer of the decoder and the corresponding word of the target audio caption in each step of the decoder. Summation of all the cross entropy values is used as a loss function $\mathcal{L}_{\text{gen.}}$, which is viewed as the loss of the audio caption generation. Let $\boldsymbol{o}_t$ be the vector provided by the output layer in step t, $\boldsymbol{y}_t$ be the one-hot vector representing w_t, the tth word in the current training sentence, and n be the number of words in the sentence. The error function is then expressed as follows:

$$\mathcal{L}_{\text{gen.}} \equiv \sum_{t=1}^{n} \text{cross entropy}(\boldsymbol{o}_t, \boldsymbol{y}_t) \tag{4}$$

$$= \sum_{t=1}^{n} -\log(\hat{p}_t(w_t)) \tag{5}$$

Then, the model is optimized by backpropagation based on $\mathcal{L}_{\text{gen.}}$.

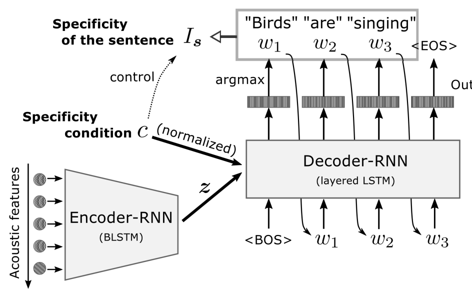


Figure 2: Block diagram of CSCG. Specificity condition c is given to the decoder, so that the resulting output sentence has the specificity close to the value of c.

3.2. Conditional Sequence-to-Sequence Caption Generator

Inspired by the conditional generative models that have been successfully applied in various works [15, 16, 17, 18, 19], here we propose a conditional sequence-to-sequence caption generator (CSCG) to control the specificity of the generated audio captions.

As illustrated in Figure 2, the encoder of the CSCG works in the same manner as that of the SCG. In addition, specificity condition c is given to the decoder in addition to the latent variable derived from the audio signal. The CSCG is trained to generate the following optimal audio caption $\bar{s}$:

$$\bar{\boldsymbol{s}} = \arg\max_{\boldsymbol{s}} p(\boldsymbol{s} \mid \boldsymbol{z}, c) \tag{6}$$

We expect $\bar{s}$ to have a specificity close to c and, at the same time, correctly correspond to the input acoustic signal.

We can train the CSCG by alternatively performing the following two steps. In the first step, audio caption generation and the specificity are learned simultaneously. The pairs of acoustic signals and audio captions are used for the learning. The specificity of an audio caption of these pairs, I_s, is input to the decoder as the specificity condition, and the model is trained by backpropagation. To control the specificity of the generated captions, we introduce the specificity loss $\mathcal{L}_{\text{sp.}}$. The total loss function in this step, $\mathcal{L}_{\text{SC-1}}$, is defined as the weighted sum of $\mathcal{L}_{\text{gen.}}$ and $\mathcal{L}_{\text{sp.}}$:

$$\mathcal{L}_{\text{SC-1}} \equiv \mathcal{L}_{\text{gen.}} + \lambda \mathcal{L}_{\text{sp.}}, \tag{7}$$

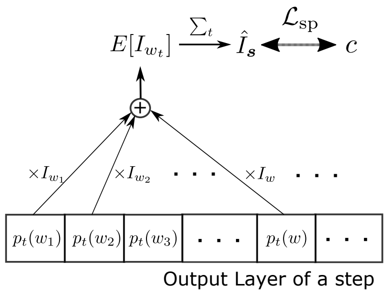


Figure 3: Estimation of specificity of a generated caption and specificity loss. Each value at the output unit, $p_t(w)$, denotes the estimated probability for the corresponding word w at step t.

where λ is a hyperparameter to balance the two loss values. Audio captioning is an ill-posed problem with potentially multiple solutions, and the second term has the role of regularization to determine the unique solution by adding constraints to the specificity of the generated caption.

The text generation with the decoder includes a discrete process to choose one word in each step, and that makes it impossible to backpropagate the losses. To solve this problem, we approximate the specificity of generated captions without discrete processes. Figure 3 illustrates the estimation process. The expectation of the amount of information of $\hat{w}_t$ corresponding to the tth step of the decoder can be calculated using its output layer, each unit of which encodes the probability of corresponding words.

$$E[I_{\hat{w}_t}] = \sum_w \hat{p}_t(w) I_w \ . \tag{8}$$

The summation of $E[I_{\hat{w}_t}]$ for all steps is $\hat{I}_{\hat{\boldsymbol{s}}}$, which is the estimated value of the specificity of the generated caption $\hat{\boldsymbol{s}}$.

$$\hat{I}_{\hat{\boldsymbol{s}}} = \sum_{t=1}^{n} E[I_{\hat{w}_t}] \tag{9}$$

Here, we define specificity loss $\mathcal{L}_{\text{sp.}}$ as follows:

$$\mathcal{L}_{\text{sp.}} \equiv (\hat{I}_{\hat{\boldsymbol{s}}} - c)^2 \ . \tag{10}$$

We can optimize the model by backpropagation using $\mathcal{L}_{\text{sp}}$, because it is calculated from the vectors obtained from the output layer of the decoder by using multiplication and addition only.

The second step is introduced to alleviate overfitting with respect to specificity. In this step, the decoder is trained only with texts rather than audio-text pairs. First, a latent variable $\boldsymbol{z}$ is extracted in advance from an audio signal by using the encoder with the current parameter. This means that we sampled $\boldsymbol{z}$ from real audio signals, rather than using random vectors, but signals not associated with any audio captions can be used here. Then, the specificity condition c is generated randomly. As training sentences, any captions with the closest specificity value to c can be used. We train only the decoder using backpropagation based on the following loss function $\mathcal{L}_{\text{SC-2}}$:

$$\mathcal{L}_{\text{SC-2}} \equiv \lambda' \mathcal{L}_{\text{gen.}} + \lambda \mathcal{L}_{\text{sp.}} \tag{11}$$

Hyperparameter λ' smaller than 1 is chosen. Even when the audio caption for calculating $\mathcal{L}_{\text{gen.}}$ does not correspond to the input signal, the first term has the role of regularization to suppress the generation of unnatural sentences.

Table 1: Experimental conditions.

Decoder LSTM layers	3
LSTM cells	512
Latent variable dimensions	256
Output word labels	1177
Normalization of c	division ($\max(I_s) \to 2.0$)
Batchsize	200
Total epoch	400
Hyper-parameter λ	2.0×10^{-2}
Hyper-parameter λ'	1.0×10^{-2}
Optimizer	ADAM [20]
MFCC dimensions	80
FFT window (MFCC)	2048 samples
FFT shift (MFCC)	512 samples

4. EVALUATION

To evaluate the effectiveness of the proposed model, we performed objective and subjective experiments.

4.1. Dataset

We used a part of the audio signals contained in Free Sound Dataset Kaggle 2018 [21], which is a subset of FSD [22] and includes various sound samples digitized at 44.1 kHz with linear PCM of 16 bit accuracy. We chose 392 signals for the training set and 29 for the test set. These signals are not longer than 6 s in length and include various everyday sounds.

To build the training set, audio captions were collected from human listeners. All the collected captions were in Japanese. Since one audio signal can correspond to various captions with various specificity values, multiple audio captions were attached to each audio signal. To accomplish this, 72 Japanese speakers were asked to describe the sound in Japanese text. We associated one to four audio captions for each training signal, and five audio captions for each test signal. The total numbers of captions were 1,113 in the training dataset and 145 in the test dataset. Then, the captions for the training signals were augmented to be expanded to 21,726, by manually deleting or replacing the words.

4.2. Conditions

Table 1 lists the experimental conditions. We used a series of mel-frequency cepstral coefficients (MFCC) and f0 as the input. The vocabulary size for the system was 1,440, as there were 1,437 kinds of words in the audio captions for training, and three special symbols “BOS”, “EOS”, and “UNK” (unknown word).

4.3. Examples

Table 2 shows some examples of the captions generated from the test signals. They were manually translated from the Japanese output.

4.4. Controllability of Specificity

Table 3 lists the averages and the standard deviations of the specificities for generated captions. Since the SCG does not deal with the specificity, the standard deviation is relatively large. On the other hand, the specificity values with the CSCG on average are quite close to the conditioned input c. This shows that the proposed conditioning mechanism works effectively.

Table 2: Examples of generated audio captions (English translation).

Sound source	Methods	c	Generated captions
Bell	SCG		A high-pitched metallic sound echoes.
	CSCG	20	A loud sound.
		50	A high sound like friction of metals.
		80	A metal bell is hit only once and makes loud, high and sustained sound.
		110	A high-pitched sound sounds as if a metal is hit, first loudly and then gradually fades out.
Bass drum	SCG		A low sound rings for a moment.
	CSCG	20	A low sound sounds for a moment.
		50	A light and low-pitched sound as if something is dashed on the mat for a moment.
		80	A drum is uninterestedly played, making a faint, very low-pitched sound only once.
		110	A faint, low-pitched sound sounds as if something is hit dully, and it soon disappears.
Glass	SCG		High-pitched sounds continue as if a metal is rolling
	CSCG	20	Glass is broken.
		50	A dry sound of breaking glass sounds once a little loudly.
		80	A high-pitched sound as if glass is breaking diminishes in a moment.
		110	A high, cold sound as if glass is breaking is heard for one or two seconds.

Table 3: Specificity of the generated captions.

Methods	c	Average	SD
SCG		38.0	21.2
CSCG	20	21.7	2.4
	50	57.7	5.0
	80	90.5	9.5
	110	107.2	20.6

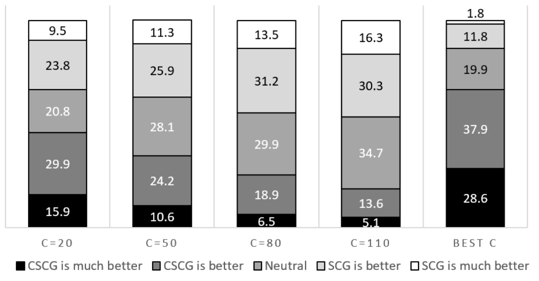


Figure 4: Comparisons of SCG and CSCG. The "BEST c" bar shows the results obtained by using the best c value (among the four) for each signal. The numbers in the bars show the percentage.

4.5. Objective Scores

Table 4 shows the BLEU scores. The CSCG with $c = 50$ marks the best BLEU, 17.01%, but it is still lower than that of human captions. Note that BLEU has a penalty for short sentences, which adversely affected the BLEU of the CSCG with low c values.

Table 4: BLEU Scores.

Methods	c	BLEU [%]
SCG		13.02
CSCG	20	5.83
	50	17.01
	80	12.52
	110	11.21
Human		22.35

4.6. Subjective Evaluation

We evaluated the proposed methods with two kinds of subjective evaluations.

Evaluation 1 investigated acceptability for the generated captions. The test audio signals and corresponding generated captions

Table 5: Acceptability scores.

Methods	c	Average	SD
SCG		1.45	1.13
CSCG	20	1.69	1.17
	50	1.29	1.11
	80	1.14	1.16
	110	1.07	1.07
Human		2.11	0.87

were presented to 41 subjects who understand Japanese. The subjects evaluated the captions in four levels: "very suitable", "suitable", "partly suitable" and "unsuitable". These answers were converted to points of 3, 2, 1 and 0, and the values of the average were the metric of acceptability. The captions given by humans were also evaluated for comparison. All the subjects responded to the 29 sound sources, for a total of 1,189 responses. Table 5 shows the results. The average scores of all methods are over 1.0, which is higher than the point of "partly suitable". The CSCG with $c = 20$ has the best acceptability within the proposed method.

Evaluation 2 compared the SCG and the CSCG models. The subjects were presented with one audio signal and two audio captions, "A" and "B." They were then asked to choose one of the five options: "A is much better", "A is better", "Neutral", "B is better", or "B is much better". Either "A" or "B" (randomly selected) was the audio caption generated with the SCG and the other was the one generated with the CSCG. Figure 4 shows the results. With an appropriate choice of c, CSCG outperformed SCG for about 2/3 of the test samples. That is, if the optimal c value is known somehow in advance, CSCG can produce better captions compared with SCG.

5. CONCLUSION

This paper proposed a neural audio captioning system for audio signals. The experiments showed that two versions of the proposed method, SCG and CSCG, work effectively and that the conditional version (CSCG) can successfully control the amount of information contained in the output sentence. It was also shown that CSCG generated subjectively better captions than SCG when we could choose the best specificity value for each signal. Unlike the existing audio classification systems, the proposed system does not solve the classification problem but performs sentence generation using the learned vocabulary, as in machine translation. For this reason, it tends to perform reasonably well even for unknown or ambiguous sounds. In our future work, we will investigate a specificity adaptation method for individual sounds, situations, and applications.

6. REFERENCES

[1] Mark D. Plumbley, Christian Kroos, Juan P. Bello, Gael Richard, Daniel P.W. Ellis, and Annamaria Mesaros: *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*. Tampere University of Technology. Laboratory of Signal Processing (2018).

[2] Haomin Zhang, Ian McLoughlin, and Yan Song: Robust sound event recognition using convolutional neural networks. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 559–563 (2015).

[3] Emre Cakir, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen Polyphonic sound event detection using multi label deep neural networks. In *2015 International Joint Conference on Neural Networks (IJCNN)*, 1–7 (2015).

[4] Daniele Barchiesi, Dimitrios Giannoulis, Dan Stowell, and Mark D. Plumbley Acoustic scene classification: Classifying environments from the sounds they produce. *IEEE Signal Processing Magazine*, 32, 3, 16–34 (2015).

[5] Cristhian Potes, Saman Parvaneh, Asif Rahman, and Bryan Conroy: Ensemble of feature-based and deep learning-based classifiers for detection of abnormal heart sounds. In *2016 Computing in Cardiology Conference (CinC)*, 621–624 (2016).

[6] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, and Noboru Harada: Optimizing acoustic feature extractor for anomalous sound detection based on Neyman-Pearson lemma. In *2017 25th European Signal Processing Conference (EUSIPCO)*, 698–702 (2017).

[7] Shota Ikawa and Kunio Kashino: Generating Sound Words from Audio Signals of Acoustic Events with Sequence-to-Sequence Model. In *Acoustics, Speech and Signal Processing (ICASSP)*, 346 – 350 (2018).

[8] Shota Ikawa and Kunio Kashino: Acoustic event search with an onomatopoeic query: measuring distance between onomatopoeic words and sounds. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE)*, 59–63 (2018).

[9] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le: Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3104–3112 (2014).

[10] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *CoRR* abs/1406.1078 (2014).

[11] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan: Show and Tell: A Neural Image Caption Generator. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2015).

[12] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio: Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning (ICML)*, 2048–2057 (2015).

[13] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Cernockỳ, and Sanjeev Khudanpur: Recurrent neural network based language model. In *INTERSPEECH*, 1045–1048 (2010).

[14] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. : Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2625–2634 (2015).

[15] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve J. Young: Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems. *CoRR* abs/1508.01745 (2015).

[16] Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan: A Persona-Based Neural Conversation Model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, 1, 994–1003 (2016).

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros.: Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 1125–1134 (2017).

[18] Mehdi Mirza and Simon Osindero: Conditional generative adversarial nets. *arXiv:1411.1784* (2014).

[19] Emily L. Denton, Soumith Chintala, Rob Fergus, et al.: Deep generative image models using a Laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems (NIPS)*, 1486–1494 (2015).

[20] Diederik P. Kingma and Jimmy Ba: Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representation (ICLR)* (2015).

[21] FSD: Freesound General-Purpose Audio Tagging Challenge — Kaggle. `https://www.kaggle.com/c/freesound-audio-tagging/data`. accessed 2018/12/31.

[22] Eduardo Fonseca, Jordi Pons, Xavier Favory, Frederic Font, Dmitry Bogdanov, Andrés Ferraro, Sergio Oramas, Alastair Porter, and Xavier Serra: Freesound Datasets: a platform for the creation of open audio datasets. In *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR)*, 486–493 (2017).

RU MULTICHANNEL DOMESTIC ACOUSTIC SCENES 2019: A MULTICHANNEL DATASET RECORDED BY DISTRIBUTED MICROPHONES WITH VARIOUS PROPERTIES

Keisuke Imoto

Ritsumeikan University, Japan
keisuke.imoto@ieee.org

Nobutaka Ono

Tokyo Metropolitan University, Japan
onono@tmu.ac.jp

ABSTRACT

Acoustic scene analysis has seen extensive development recently because it is used in applications such as monitoring, surveillance, life-logging, and advanced multimedia retrieval systems. Acoustic sensors, such as those used in smartphones, wearable devices, and surveillance cameras, have recently rapidly increased in number. The simultaneous use of these acoustic sensors will enable a more reliable analysis of acoustic scenes because they can be utilized for the extraction of spatial information or application of ensemble techniques. However, there are only a few datasets for acoustic scene analysis that make use of multichannel acoustic sensors, and to the best of our knowledge, no large-scale open datasets recorded with multichannel acoustic sensors composed of different devices. In this paper, we thus introduce a new publicly available dataset for acoustic scene analysis, which was recorded by distributed microphones with various characteristics. The dataset is freely available from `https://www.ksuke.net/dataset`.

Index Terms— Distributed microphone array, acoustic scene classification, publicly available dataset

1. INTRODUCTION

Acoustic scene classification (ASC), which associates a sound with a related scene, has recently attracted much attention because of its many useful applications such as those in monitoring systems for elderly people or infants [1, 2], automatic surveillance systems [3, 4, 5, 6], automatic life-logging systems [7, 8, 9], and advanced multimedia retrieval [10, 11, 12, 13].

Many approaches to ASC are based on machine learning techniques, especially deep neural network (DNN)-based methods [14, 15, 16, 17, 18, 19, 20, 21]. For instance, Valenti *et al.* have proposed a method based on convolutional neural networks (CNNs) [17], which allows robust feature extraction of acoustic scenes against time and frequency shifts in the spectrogram domain. More sophisticated models such as VGG [22], ResNet [23], and Xception [24], which achieve reasonable performance in image recognition, have also been applied to acoustic scene analysis [18, 19, 20]. Ren *et al.* have applied the attention mechanism to CNN-based acoustic scene classification [21]. These DNN-based approaches for ASC require a large-scale dataset; thus, the large-scale datasets that are publicly available have contributed to related research and development. Moreover, evaluation using a publicly available dataset is an impartial means of assessing a method under development. There are some open datasets for ASC, such as the LITIS dataset [25], TUT Acoustic Scenes 2016 [26] and 2017 [27], and TUT Urban Acoustic Scenes 2018 [28], which were recorded with a single or stereo microphone(s). There are also other publicly available datasets for detecting sound events that occur in a domestic environment, such as the CHiME-Home dataset [29].

On the other hand, acoustic sensors that are easily accessible, such as those in smartphones, smart speakers, IoT devices, and surveillance cameras, have rapidly increased in number. By making use of these microphones simultaneously, we obtain spatial information, which will help to recognize acoustic scenes [30, 31, 32, 33]. For instance, an acoustic scene "cooking" and related sounds tend to occur in a kitchen, whereas an acoustic scene "shaving" and related sounds are likely to occur in a powder room. There are also datasets for ASC or sound event classification based on multichannel observation, such as ITC-Irst AED Database [34], FINCA Multi-channel Acoustic Event Dataset [35], and SINS Database [36]. For example, ITC-Irst AED Database consists of sound recordings including 16 types of acoustic events, such as "door knock," "cough," and "keyboard." Eight T-shaped microphone arrays, each of which had four microphones, were used for the recording. SINS Database consists of sound recordings including 16 different activities in the home, such as "cooking," "vacuuming," and "phone call." The recording was conducted using 13 microphone arrays, all of which were composed of four Sonion N8AC03 MEMS microphones.

Considering that a large microphone array is constructed by combining microphones that are mounted on smartphones, smart speakers, IoT devices, and surveillance cameras, some microphones often have a mismatch under acoustic conditions, such as the sampling rate, frequency response, sensitivity, and/or noise level. This condition mismatch often has a detrimental effect on the classification performance of acoustic scenes and needs to be addressed. However, there are no open datasets for ASC that were recorded in a home environment using multiple microphones with various properties. In this paper, we thus introduce a dataset for ASC named Ritsumeikan University (RU) Multichannel Domestic Acoustic Scenes 2019, which was recorded by distributed microphones with various properties. The characteristics of RU Multichannel Domestic Acoustic Scenes 2019 are as follows:

- The dataset consists of 21 kinds of acoustic scenes including an "absent" scene and high-privacy scenes such as "toilet," "sleeping," and "taking a bath/shower."
- The dataset was recorded using 42 distributed microphones with various characteristics.
- The dataset consists of a total of 1,995.8 h of sounds (47.5 h $\times$ 42 ch.), which can be divided into about 11,400 segments of 15 s sounds for each channel.
- The dataset can be utilized for evaluating ASC methods using spatial information, ensemble techniques, or domain adapta-

https://doi.org/10.33682/hy6p-g607

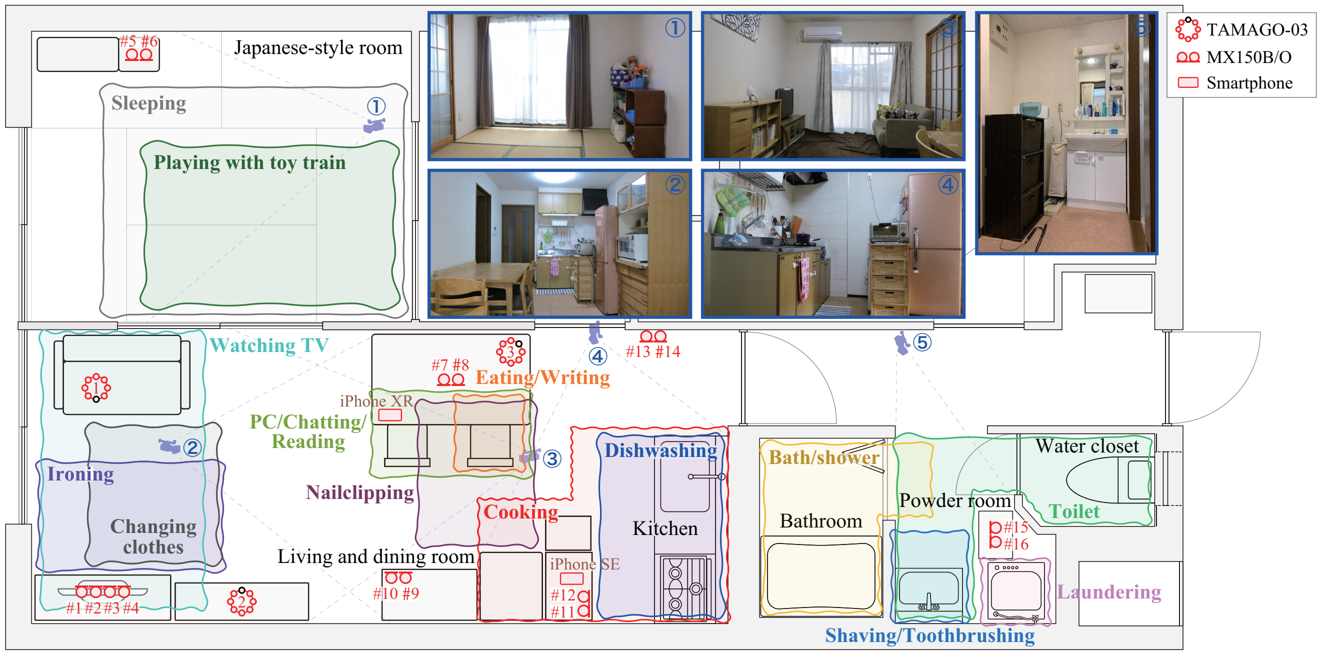


Figure 1: Floor plan of recording environment, microphone arrangement, and approximate positions of sound sources

tion techniques (by combining this and another multichannel dataset such as SINS Database [36]).

- The dataset includes sample videos of most of the sound clips (for understanding of recording environments and situations)

This dataset is freely available and can be downloaded at [37].

The remainder of this paper is structured as follows. In section 2, we provide an overview of RU Multichannel Domestic Acoustic Scenes 2019. In section 3, the benchmark evaluation results are reported. Finally, a conclusion is given in section 4.

2. OVERVIEW OF RU MULTICHANNEL DOMESTIC ACOUSTIC SCENES 2019

2.1. Recording conditions

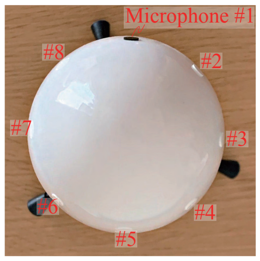
The dataset was recorded in an apartment where people actually live. As shown in Fig. 1, the recording was conducted in six different rooms: a Japanese-style room (washitsu), hall, powder room, bathroom, water closet, and a combined living room, dining room, and kitchen. As the recording equipment, three TAMAGO-03 microphone arrays [38] (8ch $\times$ 3), 16 Shure MX150B/O microphones (1ch $\times$ 16), one iPhone SE (1ch $\times$ 1), and one iPhone XR (1ch $\times$ 1) were used. Each TAMAGO-03 array consisted of eight microphones mounted on a circle of a 36.5 mm radius at 45° intervals, as shown in Fig. 2-(a). The sampling rate and bit depth of the TAMAGO-03 microphones were 16 kHz and 16, respectively. The Shure MX150B/O microphones were arranged in pairs with 50.0 mm intervals. As the microphone amplifier and AD converter for the MX150B/O microphones, we used two MOTU 8Ms [39]. The sampling rate and bit depth of the MX150B/O microphones [40], iPhone XR, and iPhone SE were 48 kHz and 16, respectively. The microphones were synchronized between microphones in each TAMAGO-03 array and 16ch MX150B/O microphones, respectively, but not between different devices. The recording conditions are given in detail in [37].

Table 1: Recorded acoustic scenes and their durations

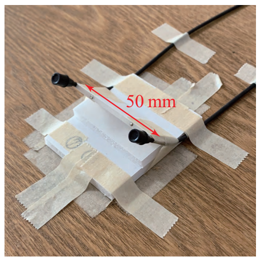
Acoustic scene	# clips	Duration (min)
Absent	26	125.3
Changing clothes	67	119.8
Chatting	23	121.5
Cooking	14	228.0
Dishwashing	36	122.8
Eating	24	129.3
Ironing	25	129.6
Laundering	10	138.0
Moving	30	122.0
Nail clipping	37	121.1
Operating PC	22	123.3
Playing with toys	21	127.5
Reading newspaper/magazine	25	121.5
Shaving	59	146.5
Sleeping	23	144.0
Taking a bath/shower	18	181.5
Toilet	101	134.6
Toothbrushing	42	132.5
Vacuuming	29	122.8
Watching TV	28	128.4
Writing	18	131.2

2.2. Recorded acoustic scenes and recording procedure

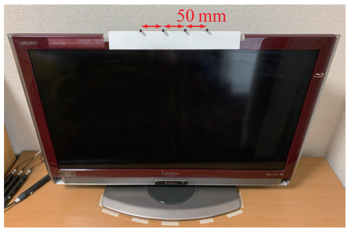
We recorded 21 acoustic scenes that frequently occur in daily activities at home. Table 1 lists the recorded acoustic scenes, which include “absent” and high-privacy scenes such as “toilet,” “chang-



(a) TAMAGO-03



(b) MX150B/O



(c) MX150B/O (mounted on TV)

Figure 2: Detailed microphone arrangements

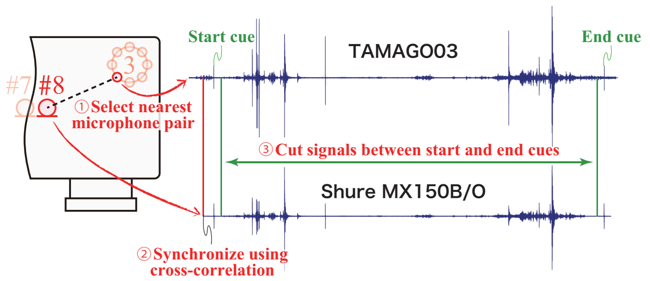
ing clothes," "taking a bath/shower," and "sleeping." Each sound clip includes all the sounds derived from a series of actions in one scene, for instance, a sound clip of "toothbrushing" includes sounds derived from "picking up toothbrush," "putting toothpaste on toothbrush," "brushing teeth," and "rinsing mouth." The approximate position of the sound source in each acoustic scene is also shown in Fig. 1, except for the acoustic scenes "absent," "moving," and "vacuuming," in which the sound may occur over the entire apartment. Each recording was started with a cue, which was an impulsive sound, but detailed scenarios and recording times were not directed.

To ensure the diversity of recorded sounds, we used various household commodities and electronic devices such as four different kitchen sponges, two irons, three nail clippers, three PCs, four computer mouses, three electric shavers, five toothbrushes, and two vacuum cleaners. Figure 3 shows these household commodities and electronic devices.

2.3. Postprocessing

Since the microphones were not synchronized between different devices, after recordings, we simply synchronized the sound clips using the cross-correlation between the nearest microphone pair. The procedure for the synchronization and reshaping of recorded signals is shown in Fig. 4. We first selected the nearest microphone pair from the unsynchronized microphones, and we then synchronized the acoustic signals recorded by the microphone pair using the cross-correlation all over the signals. Since the sampling rates of the TAMAGO-03 microphones and the other microphones were 16 kHz and 48 kHz, respectively, the recorded sound at 48 kHz was downsampled to 16 kHz when synchronizing. After that, we cut the acoustic signals to remove cue sounds, which are irrelevant to recorded scenes. Note that we did not take an arrival time difference of sounds between channels, which is a significant cue for extracting spatial information, into account; thus, sound clips needs to be resynchronized accurately using blind compensation techniques for distributed microphone array [41, 42] if we extract spatial information using conventional methods of microphone array processing.

Figure 3: Household commodities and electronic devices used for recording


Start cue
TAMAGO03
End cue
#7
#8
3
①Select nearest microphone pair
③Cut signals between start and end cues
Shure MX150B/O
②Synchronize using cross-correlation
Figure 4: Synchronization procedure between unsynchronized microphones

Moreover, the different devices have sampling frequency mismatch; however we did not compensate the mismatch between devices.

Although the length of the sound differs from sound clip to sound clip, we suppose that each sound clip will be divided into 10 or 15 s segments, which are the units of analysis. A manipulation tool that divides each sound clip into shorter segments is also included in the dataset.

2.4. Contents of RU Multichannel Domestic Acoustic Scenes 2019

RU Multichannel Domestic Acoustic Scenes 2019 includes the following contents:

- Sound files in wav format (RIFF waveform audio format)
- Impulse responses at each microphone position (RIFF waveform audio format)
- Documents of recording conditions and postprocessing procedures
- Sample videos (for understanding of recording environments and situations)
- Tools for manipulating sound files

Each sound file is stored in the wave format, and 42-channel sound files obtained in each recording are stored in one directory. The dataset also contains impulse responses from some sound source

Table 2: Experimental conditions

# total microphones	42
Sound clip length	15 s
Frame length	40 ms
Frame shift	20 ms
Network structure	3 conv. & 3 FC layers
Pooling in CNN layers	3×3 max pooling
Activation function	ReLU, softmax (output layer)
# channels of CNN	42, 32, 16
# units of FC layers	128, 64, 32
Dropout ratio in FC layer	0.5
# epoch	150

locations to all microphone positions. Documents providing the details of recording conditions, postprocessing procedures, and photographs of recording environments are also included in the dataset. We provide some sample videos for understanding of the recording environments and useful tools for manipulating sound files (e.g., a tool for dividing sound clips into segments of 10 or 15 s length).

3. BENCHMARK OF ACOUSTIC SCENE CLASSIFICATION TASK

3.1. Experimental conditions

As the benchmark system in ASC, we evaluated the performance of a CNN-based method using RU Multichannel Domestic Acoustic Scenes 2019. In this experiment, we cut sound files into 15 s sounds. We then resampled the sound files to 44.1 kHz and extracted the 64-dimensional mel-band energies, which were calculated for each 40 ms time frame with 50% overlap. The implemented system was based on [17]; the detailed network structure and the parameter settings of the networks were determined with reference to [32]. Forty-two acoustic feature maps extracted from 42-channel recordings were input to different channels in the first CNN layer. The network was trained using the Adam optimizer with a learning rate of 0.001. The other experimental conditions are listed in Table 2. The evaluation was conducted using a four-fold cross-validation setup, where each fold had roughly the same number of sound clips with respect to each acoustic scene.

3.2. Experimental results

The performance of ASC using the CNN-based method was 58.3% the average F-score for all acoustic scenes. This result indicates that the ASC task using RU Multichannel Domestic Acoustic Scenes 2019 is still difficult even using the CNN architecture, which enables scene classification with reasonable performance. Thus, we consider that this dataset is suitable for evaluating ASC performance with more sophisticated acoustic features based on spatial information and/or models based on neural networks. More detailed experimental results are given in [37].

4. CONCLUSION

In this paper, we introduced the RU Multichannel Domestic Acoustic Scenes 2019 dataset, which was recorded by multichannel distributed microphones with various devices. This dataset consists of over 45 h $\times$ 42 channels of sounds recorded in a home environment in which people actually live. We hope that RU Multichannel Domestic Acoustic Scenes 2019 will be widely used for evaluating methods of ASC utilizing spatial information, ensemble techniques, and domain adaptation techniques.

5. ACKNOWLEDGEMENTS

This work was supported by JSPS KAKENHI Grant Numbers JP16H01735 and JP19K20304, KDDI Foundation, and Support Center for Advanced Telecommunications Technology Research.

6. REFERENCES

[1] Y. Peng, C. Lin, M. Sun, and K. Tsai, “Healthcare audio event classification using hidden Markov models and hierarchical hidden Markov models,” *Proc. IEEE International Conference on Multimedia and Expo* (*ICME*), pp. 1218–1221, 2009.

[2] P. Guyot, J. Pinquier, and R. André-Obrecht, “Water sound recognition based on physical models,” *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp. 793–797, 2013.

[3] A. Harma, M. F. McKinney, and J. Skowronek, “Automatic surveillance of the acoustic activity in our living environment,” *Proc. IEEE International Conference on Multimedia and Expo* (*ICME*), 2005.

[4] R. Radhakrishnan, A. Divakaran, and P. Smaragdis, “Audio analysis for surveillance applications,” *Proc. 2005 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics* (*WASPAA*), pp. 158–161, 2005.

[5] S. Ntalampiras, I. Potamitis, and N. Fakotakis, “On acoustic surveillance of hazardous situations,” *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp. 165–168, 2009.

[6] T. Komatsu and R. Kondo, “Detection of anomaly acoustic scenes based on a temporal dissimilarity model,” *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp. 376–380, 2017.

[7] A. Eronen, V. T. Peltonen, J. T. Tuomi, A. P. Klapuri, S. Fagerlund, T. Sorsa, G. Lorho, and J. Huopaniemi, “Audio-based context recognition,” *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 14, no. 1, pp. 321–329, 2005.

[8] K. Imoto and S. Shimauchi, “Acoustic scene analysis based on hierarchical generative model of acoustic event sequence,” *IEICE Trans. Inf. Syst.*, vol. E99-D, no. 10, pp. 2539–2549, 2016.

[9] J. Schröder, J. Anemiiller, and S. Goetze, “Classification of human cough signals using spectro-temporal Gabor filterbank features,” *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp. 6455–6459, 2016.

[10] T. Zhang and C. J. Kuo, “Audio content analysis for online audiovisual data segmentation and classification,” *IEEE Trans. Audio Speech Lang. Process.*, vol. 9, no. 4, pp. 441–457, 2001.

[11] Q. Jin, P. F. Schulam, S. Rawat, S. Burger, D. Ding, and F. Metze, “Event-based video retrieval using audio,” *Proc. INTERSPEECH*, 2012.

[12] Y. Ohishi, D. Mochihashi, T. Matsui, M. Nakano, H. Kameoka, T. Izumitani, and K. Kashino, "Bayesian semi-supervised audio event transcription based on Markov Indian buffet process," *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp. 3163–3167, 2013.

[13] J. Liang, L. Jiang, and A. Hauptmann, "Temporal localization of audio events for conflict monitoring in social media," *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp. 1597–1601, 2017.

[14] K. Imoto, "Introduction to acoustic event and scene analysis," *Acoustical Science and Technology*, vol. 39, no. 3, pp. 182–188, 2018.

[15] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *the Detection and Classification of Acoustic Scenes and Events* (*DCASE*), pp. 1–5, 2017.

[16] H. Jallet, E. Çakır, and T. Virtanen, "Acoustic scene classification using convolutional recurrent neural networks," *the Detection and Classification of Acoustic Scenes and Events* (*DCASE*), pp. 1–5, 2017.

[17] M. Valenti, S. Squartini, A. Diment, G. Parascandolo, and T. Virtanen, "A convolutional neural network approach for acoustic scene classification," *Proc. International Joint Conference on Neural Networks* (*IJCNN*), pp. 547–1554, 2017.

[18] R. Tanabe, T. Endo, Y. Nikaido, T. Ichige, P. Nguyen, Y. Kawaguchi, and K. Hamada, "Multichannel acoustic scene classification by blind dereverberation, blind source separation, data augmentation, and model ensembling," *Tech. Rep. DCASE*, 2018.

[19] A. Raveh and A. Amar, "Multi-channel audio classification with neural network using scattering transform," *Tech. Rep. DCASE*, 2018.

[20] Y. Liping, C. Xinxing, and T. Lianjie, "Acoustic scene classification using multi-scale features," *Tech. Rep. DCASE*, 2018.

[21] Z. Ren, Q. Kong, K. Qian, M. D. Plumbley, and B. W. Schuller, "Attention-based convolutional neural networks for acoustic scene classification," *Proc. Detection and Classification of Acoustic Scenes and Events* (*DCASE*) *Workshop*, pp. 39–43, 2018.

[22] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv, arXiv:1409.1556*, 2014.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv, arXiv:1512.03385*, 2015.

[24] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), pp. 1251–1258, 2017.

[25] A. Rakotomamonjy and G. Gasso, "Histogram of gradients of time-frequency representations for audio scene classification," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 23, no. 1, pp. 142–153, 2015.

[26] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," *Proc. European Signal Processing Conference* (*EUSIPCO*), pp. 1128–1132, 2016.

[27] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, B. Raj, and T. Virtanen, "DCASE 2017 challenge setup: Tasks, datasets and baseline system," *Proc. Workshop on Detection and Classification of Acoustic Scenes and Events* (*DCASE*), pp. 85–92, 2017.

[28] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," *Proc. Workshop on Detection and Classification of Acoustic Scenes and Events* (*DCASE*), pp. 9–13, 2018.

[29] P. Foster, S. Sigtia, S. Krstulovic, J. Barker, and M. D. Plumbley, "Chime-home: A dataset for sound source recognition in a domestic environment," *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics* (*WASPAA*), pp. 1–5, 2015.

[30] H. Kwon, H. Krishnamoorthi, V. Berisha, and A. Spanias, "A sensor network for real-time acoustic scene analysis," *Proc. IEEE International Symposium on Circuits and Systems*, pp. 169–172, 2009.

[31] K. Imoto and N. Ono, "Spatial cepstrum as a spatial feature using distributed microphone array for acoustic scene analysis," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 25, no. 6, pp. 1335–1343, 2017.

[32] K. Imoto, "Acoustic scene analysis using partially connected microphones based on graph cepstrum," *Proc. European Signal Processing Conference* (*EUSIPCO*), pp. 2453–2457, 2018.

[33] K. Nakadai and D. R. Onishi, "Partially-shared convolutional neural network for classification of multi-channel recorded audio signals," *Tech. Rep. DCASE*, 2018.

[34] C. Zieger and M. Omologo, "Acoustic event detection - itcirst aed database," *Internal ITC report, Tech. Rep.*, 2005.

[35] J. Kürby, R. Grzeszick, A. Plinge, and G. A. Fink, "Bag-of-features acoustic event detection for sensor networks," *Proc. Workshop on Detection and Classification of Acoustic Scenes and Events* (*DCASE*), pp. 1–5, 2016.

[36] G. Dekkers, S. Lauwereins, B. Thoen, M. W. Adhana, H. Brouckxon, B. V. Bergh, T. Waterschoot, B. Vanrumste, M. Verhelst, and P. Karsmakers, "The SINS database for detection of daily activities in a home environment using an acoustic sensor network," *Proc. Workshop on Detection and Classification of Acoustic Scenes and Events* (*DCASE*), pp. 1–5, 2017.

[37] https://www.ksuke.net/dataset.

[38] http://www.sifi.co.jp/system/modules/pico/index.php?content_id=39&ml_lang=en.

[39] https://motu.com/products/avb/8m.

[40] https://pubs.shure.com/guide/MX150/en-US.

[41] N. Ono, H. Kohno, and S. Sagayama, "Blind alignment of asynchronously recorded signals for distributed microphone array," *Proc. Applications of Signal Processing to Audio and Acoustics* (*WASPAA*), pp. 161–164, 2009.

[42] Z. Liu, "Sound source separation with distributed microphone arrays in the presence of clock synchronization errors," *Proc. International Workshop on Acoustic Echo and Noise Control* (*IWAENC*), pp. 1–4, 2008.

SHUFFLING AND MIXING DATA AUGMENTATION FOR ENVIRONMENTAL SOUND CLASSIFICATION

Tadanobu Inoue[1], *Phongtharin Vinayavekhin*[1], *Shiqiang Wang*[2], *David Wood*[2], *Asim Munawar*[1], *Bong Jun Ko*[2], *Nancy Greco*[2], *Ryuki Tachibana*[1]

[1] IBM Research, Tokyo, Japan, {inouet, pvmilk, asim, ryuki}@jp.ibm.com
[2] IBM Research, Yorktown Heights, NY, USA, {wangshiq, dawood, bongjun_ko, grecon}@us.ibm.com

ABSTRACT

Smart speakers have been recently adopted and widely used in consumer homes, largely as a communication interface between human and machines. In addition, these speakers can be used to monitor sounds other than human voice, for example, to watch over elderly people living alone, and to notify if there are changes in their usual activities that may affect their health. In this paper, we focus on the sound classification using machine learning, which usually requires a lot of training data to achieve good accuracy. Our main contribution is a data augmentation technique that generates new sound by shuffling and mixing two existing sounds of the same class in the dataset. This technique creates new variations on both the temporal sequence and the density of the sound events. We show in DCASE 2018 Task 5 that the proposed data augmentation method with our proposed convolutional neural network (CNN) achieves an average of macro-averaged F1 score of 89.95% over 4 folds of the development dataset. This is a significant improvement from the baseline result of 84.50%. In addition, we also verify that our proposed data augmentation technique can improve the classification performance on the Urban Sound 8K dataset.

Index Terms— Domestic Activities, Data Augmentation, Deep Learning, Convolutional Neural Network

1. INTRODUCTION

In recent years, there is an increasing popularity in installing smart speakers in a home environment due to its capability to interact and activate home appliances through its voice interface. The low cost of these smart speakers encourages the use of more than one device to cover a larger area of a home. The technology in smart speakers, Micro Electro Mechanical Systems (MEMS) array microphones, can be additionally used for monitoring sounds other than human voice. The smart speaker capability can be adapted through machine learning to monitor and detect human activities in daily life routine [1, 2].

We consider human activity monitoring and detection as a multi-class classification problem [3]. The task is to identify acoustic scenes and events using environmental sounds [4]. A supervised machine learning technique, deep learning based on convolutional neural networks (CNNs) to be specific, is used as a classifier. CNNs have been widely used in acoustic scene classification tasks due to their promising performance [5, 6, 7, 8, 9].

It is well-known that deep learning requires a large amount of data to train an accurate model. To increase the amount of training data and reduce overfitting, numerous data augmentation methods have been studied in the acoustic literature. Some musically inspired deformations such as pitch shifting and time stretching are adopted to augment training sound data [6, 10]. Jaitly and Hinton [11] showed that the data augmentation based on vocal tract length perturbation (VTLP) is effective to improve the performance of automatic speech recognition (ASR). Takahashi *et al.* [12] mixed two sound sources within the same class to generate a new sound. Tokozume *et al.* [13] proposed a method to mix two sound sources from different classes. Both labels and sounds are mixed and referred to as between-class data. They train the model solely using the generated data without using the original data. Zhang *et al.* [14] proposed a similar approach to use between-class data, but they also use mixing of sounds from the same class in the training.

In these previous works, the temporal order of the sound events is kept and does not generate new variations on the sound sequence. In addition, mixing by linearly combining two sounds [12, 13, 14] usually increases the number of sound events (event density) which could introduce bias in the model.

In contrast to the existing approaches mentioned above, we propose a method that increases the variation in the training samples on *both the temporal sequence and event density* (the number of the sound events in a time period) of the sound events. Our proposed method can both increase and decrease the density of sound events, while keeping the overall average density of events the same as in the original sound and thus introducing *no bias* to the model. Our proposed method augments input acoustic data by combining sounds from two sound sources of the same class. Each sound source is di-

https://doi.org/10.33682/wgyb-bt40

vided into multiple segments, and the new sound is generated by shuffling and mixing these segments of two sounds from different sound sources. This is based on our observation that environmental sound is generally composed of background sound and events; each event often occurs discretely in a sound sequence without any temporal relation with others. The fact that mixing is randomized keeps the overall average event density the same as in the original sounds.

We conduct experiments with two acoustic datasets. First, we applied the proposed method to DCASE 2018 Task 5 dataset [3, 15], which includes sounds in a home environment. Our proposed method alleviates the effect of unbalanced classes in the dataset, and significantly increases the classification performance (F1 score) and is a main ingredient in building the system [16] that won the challenge[1]. Second, the proposed method is applied to Urban Sound 8K dataset [17], where the results show that our proposed method produces comparable results to other data augmentation techniques that are designed for this dataset.

This paper starts by describing the proposed data augmentation technique in Section 2. Experimental results on two datasets are described in Section 3. Finally, the conclusion is given in Section 4.

2. SHUFFLING AND MIXING DATA AUGMENTATION

In this section, we introduce the shuffling and mixing data augmentation to increase variation of training samples for training a deep learning model. We augment sound data based on two assumptions.

First, based on our observation, we assume that environmental sound is generally composed of background sounds and foreground event sound. The foreground events often occur discretely and have no temporal relation with each other. For example, let us consider *eating* sounds as shown in Fig. 1. Foreground event sounds can be caused by the sound of dishes or kitchen utensils; however, these events are temporally independent of each other. In other words, even when the order of these sound events is swapped, the sound can still be categorized as *eating*. Therefore, it is possible to generate a new sound clip by shuffling the order of sound segments. Second, we assume that mixing two sound sources within the same class results in a new sound in the same category. This assumption has been also used in previous works [12, 14].

Based on these two assumptions, we propose a simple but effective data augmentation technique, which is comprised of two steps: (a) shuffling, and (b) mixing two sounds of the same class, as shown in Fig. 2. To simplify the explanation, let us consider two sound clips of the same class and the

[1]http://dcase.community/challenge2018/task-monitoring-domestic-activities-results

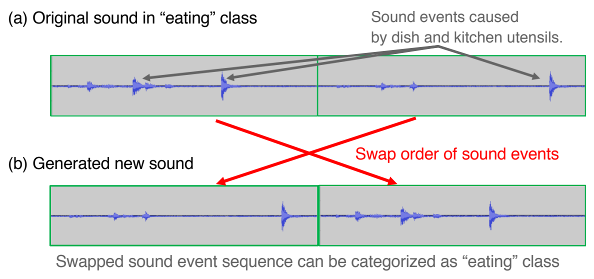

(a) Original sound in "eating" class
Sound events caused by dish and kitchen utensils.
(b) Generated new sound
Swap order of sound events
Swapped sound event sequence can be categorized as "eating" class

Figure 1: Swapping the order of sound events creates a sound in the same class.

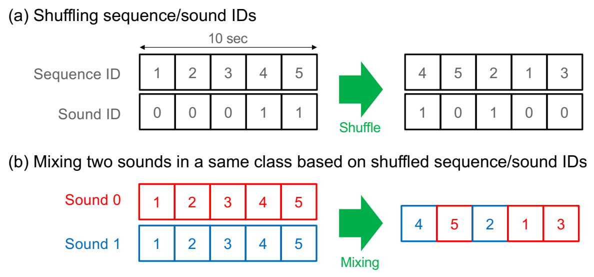


Figure 2: Generating new data based on shuffling and mixing.

same length of 10 seconds. We divide them into segments. The length of each segment can be arbitrary and is considered as a hyper-parameter that represents an estimated length of sound that contains at least one atomic foreground event. In the above example, the length of each segment is 2 seconds. We define two arrays to keep sequence IDs and sound IDs respectively and shuffle them as shown in Fig. 2(a). The sequence ID represents the order of sound segments, that is, when a sequence ID array is shuffled from $(1, 2, 3, 4, 5)$ to $(4, 5, 2, 1, 3)$, it means that the forth segment of the original sound is used as the first segment of the new sound, the fifth segment is used as the second segment and so on. The sound ID represents sound source from two sounds, *Sound*-0 or *Sound*-1, and how to mix them. For example, a sound ID $(0, 0, 0, 1, 1)$ represents a 60% *Sound*-0 mixing ratio, which is also a hyper-parameter of the method. When the sound ID is shuffled to $(1, 0, 1, 0, 0)$, the first segment of the new sound is picked from *Sound*-1 and the second segment is picked from *Sound*-0, and so on. We mix two sounds of the same class based on the shuffled sequence/sound IDs as shown in Fig. 2(b).

Generating new training samples in this way results in more variations of the temporal event location in the sound source. It also creates more variation in the number of sound events in a time period (event density). If the new sound is composed of multiple segments each containing a small number of sound events, it results in a decrease of event density. Similarly, if it is composed of multiple segments each containing a large number of sound events, the new sound will have higher event density. This is in contrast to the previous methods [12, 14] that mix the two sound sources by overlaying on top of each other, where the resulting sound keeps the same event order and tends to have higher event density than the original sound.

3. EXPERIMENTS

In this section, we evaluate our proposed data augmentation technique on two datasets with different characteristics. DCASE 2018 Task 5 dataset [15] is based on continuous recording sounds of a single person living in a vacation home over a period of one week [3]. It is composed of nine sound classes. Most of the sounds are created by one particular person and are relatively low volume except for *vacuum cleaning*. On the other hand, Urban Sound 8K dataset is created by downloading sounds from an online sound repository, Freesound.org. The recorded sounds come from various sound sources, containing ten classes of urban environmental sounds. Most of the sounds are quite noisy compared to the sounds in DCASE 2018 Task 5 dataset.

3.1. DCASE 2018 Task 5 dataset

The DCASE 2018 Task 5 dataset contains sound data captured in the living room. Each individual sound data is recorded using a single microphone array (with four microphones). There are microphone arrays at seven undisclosed locations. The dataset is divided into a development dataset and evaluation dataset. We focus on the development dataset in this paper. Each sound is 10 seconds long consisting of 4-channel 16-bit data sampled at 16 kHz. There are unequal numbers of samples in different classes, which possibly reflects the frequency of activities in real life. The amount of data in the following six classes: *cooking*, *dishwashing*, *eating*, *other*, *social activity*, and *vacuum cleaning*, is extremely small compared to the other three classes: *absence*, *watching TV*, and *working*.

The proposed data augmentation approach is used to increase the training data of the six classes to create a more balanced training set. Each sound data is divided into five segments with two seconds in length and mixed with 3-to-2 (60%) mixing ratio. Fig. 3 illustrates the amount of data in each class before and after applying our shuffling and mixing augmentation on Fold 1 of the development dataset.

As shown in Fig. 3, 30% of the training data is selected as validation data. All sounds recorded in the same session are only in either the training or validation data. This corresponds to how it is done in the baseline system. We converted the 10 second sound waveform into log-scaled mel-spectrogram (logmel) of size 40×501 matrix and used as the input to the deep learning model. More details of preprocessing are in our technical report of the DCASE 2018 challenge [16].

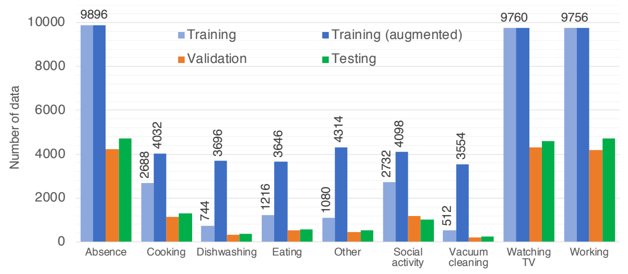


Figure 3: Number of data before and after data augmentation in Fold 1 of the development dataset. The augmentation is conducted on training data only.

Table 1: Proposed network architecture.

Layer	Output size
Input	$40 \times 501 \times 1$
Conv(7×1, 64) + BN + ReLU	$40 \times 501 \times 64$
Max pooling(4×1) + Dropout(0.2)	$10 \times 501 \times 64$
Conv(10×1, 128) + BN + ReLU	$1 \times 501 \times 128$
Conv(1×7, 256) + BN + ReLU	$1 \times 501 \times 256$
Global max pooling + Dropout(0.5)	256
Dense	128
Softmax output	9

In addition to data augmentation, we designed a new deep neural network architecture, where the main characteristics is that it starts with multiple convolutional layers across frequency axis where the kernel size on the time axis is fixed to one and then it followed by a convolutional layer across time where the kernel size on the frequency axis is fixed to one. This allows the network to look for local patterns across frequency bands and also the short-connected temporal components which represent sound events in the input data. In addition, the network also maintains the size of the time axis of the logmel until the final pooling layer. The complete network architecture and parameters are shown in Table 1.

In the dataset, one test sample has 4-channels. Sound in each channel is pre-processed and passed through the classifier independently. We average these four softmax predictions of each channel to calculate the final probability prediction for each test sample.

The experiments are carried out using the 4-fold cross validation setting of the development dataset. This corresponds to the test protocol of the DCASE 2018 Challenge. The model is trained with Adam optimizer [18] and an initial learning rate of 0.0001. We use a batch size of 256 samples and train the classifier for 500 epochs. The network weights which result in the best accuracy on the validation data is used to evaluate the test data. We examine the following configurations and compare the result with the baseline system: i) proposed CNN without data augmentation, ii) baseline CNN with proposed data augmentation, and iii)

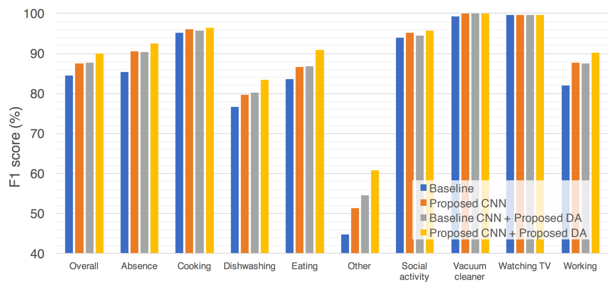


Figure 4: Comparison of macro-average F1 scores and F1 scores of each class.

proposed CNN with proposed data augmentation.

Fig. 4 shows the overall F1 scores and also for each class separately. We can see that the proposed network architecture and data augmentation approach each improves the classification performance and the combination of them gives the best performance. The overall F1 score by the proposed system is 89.95%, while the overall F1 score by the baseline is 84.50%. The proposed system improves F1 scores in all classes, especially the F1 score of *other* class.

3.2. Urban Sound 8K dataset

Urban Sound 8K dataset contains 10 sound classes of urban environmental sounds and has been widely used in acoustic classification literatures. Salamon and Bello [6] has investigated the effect of various data augmentation techniques and could be considered as a command baseline in this dataset. This experiment aims to compare an effect of those techniques the proposed shuffling and mixing data augmentation.

In the previous work [6], the details of how to augment each sound data is provided; however, implementation of the model and training procedure is not given. We attempted to replicate the result and our implementation achieved a mean accuracy of 71.6% across 10 folds for a baseline without data augmentation. Additional details are listed below: (a) We padded all sound data to 4 seconds by repeating the sound (self-concatenating) if required. During training, a 3 second segment is randomly chosen for each data sample in each epoch. However, during inference on a test sample, we slice a 3 second window with 1-frame hop in temporal axis of log-mel, pass them through the network, and ensemble the probability by averaging. (b) We replicated SB-CNN, use glorot uniform initialization for all layers and add batch normalization after each CNN layer [19]. (c) Model is trained for 50 epochs, with a minibatch of 100 samples. In each epoch, all sounds are considered in the training while undersampling method is applied to balance the number of data between all class. The model weights that performed best on the validation set are chosen for the final weights. (d) We strictly followed 10-fold cross-validation protocol[3]. During testing the Nth fold, the $(N-1)$th fold is used as validation.

We applied the proposed shuffling and mixing augmentation to the data by dividing each sound clip into 2 segments with 2 seconds in length and mixed them with 1-to-1 (50%) mixing ratio. Fig. 5 shows the difference for each class in the classification accuracy when adding each data augmentation compared to using only the original training set. Our proposed technique improves the accuracy compared to the baseline, although pitch shifting gives the best result for this dataset. The suitability of different data augmentation techniques to different datasets is worth studying in the future.

[3]https://urbansounddataset.weebly.com/urbansound8k.html

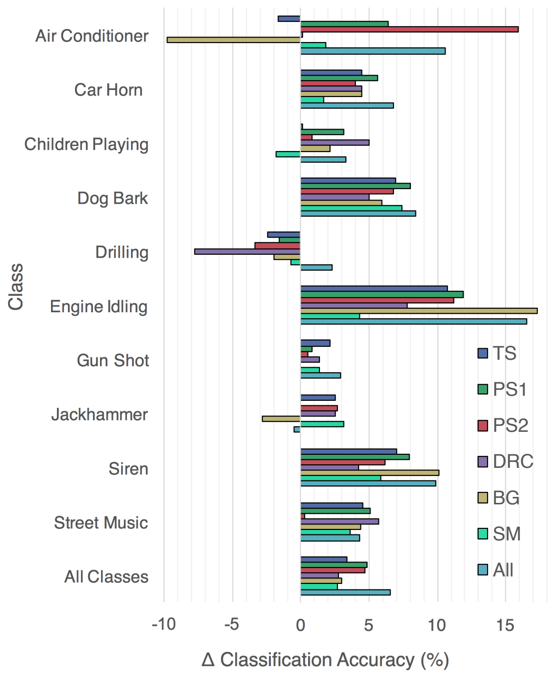


Figure 5: Overall and class-wised Δaccuracy from baseline of each data augmentation: Time Stretch (TS), Pitch Shift (PS1, PS2), Dynamic Range Compression (DRC), Back-Ground noise (BG), the proposed Shuffling and Mixing (SM), and All combined (All).

4. CONCLUSIONS

We have proposed a data augmentation technique that shuffles and mixes two sounds of the same class in training datasets. This data augmentation can generate new variations on both the sequence and the density of sound events. The proposed method is applied to DCASE 2018 Task 5 dataset and the Urban Sound 8K dataset. In general, the method improves classification results in both datasets. Specifically, it is a part of the system that won the DCASE 2018 Task 5 challenge and it also shows comparable results to other data augmentation techniques in the Urban Sound 8K dataset.

5. REFERENCES

[1] Michel Vacher, Francois Portet, Anthony Fleury, and Norbert Noury, "Development of audio sensing technology for ambient assisted living: Applications and challenges," in *International Journal of E-Health and Medical Communications (IJEHMC)*, 2011.

[2] Lode Vuegen, Peter Van Den Broeck, Bertand Karsmakers, Hugo Van hamme, and Bart Vanrumste, "Automatic monitoring of activities of daily living based on real-life acoustic sensor data: a preliminary study," in *Proc. Fourth workshop on speech and language processing for assistive technologies (SLPAT)*, 2013.

[3] Gert Dekkers, Steven Lauwereins, Bart Thoen, Mulu Weldegebreal Adhana, Henk Brouckxon, Toon van Waterschoot, Bart Vanrumste, Marian Verhelst, and Peter and Karsmakers, "The SINS database for detection of daily activities in a home environment using an acoustic sensor network," in *DCASE 2017*, November 2017.

[4] Dan Stowell, Dimitrios Giannoulis, Emmanouil Benetos, Mathieu Lagrange, and Mark D. Plumbley, "Detection and classification of acoustic scenes and events," in *IEEE Transactions on Multimedia*, 2015, vol. 17, pp. 1733–1746.

[5] Karol J. Piczak, "Environmental sound classification with convolutional neural networks," in *MLSP 2015*, 2015.

[6] Justin Salamon and Juan P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," in *IEEE Signal Processing Letters*, March 2017, vol. 24, pp. 279–283.

[7] Yoonchang Han, Jeongsoo Park, and Kyogu Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," in *DCASE 2017*, 2017.

[8] Sharath Adavanne, Pasi Pertilä, and Tuomas Virtanen, "Sound event detection using spatial features and convolutional recurrent neural network," in *ICASSP 2017*, 2017.

[9] Wei Dai, Chia Dai, Shuhui Qu, Juncheng Li, and Samarjit Das, "Very deep convolutional neural networks for raw waveforms," in *ICASSP 2017*, 2017.

[10] Brian McFee, Eric J. Humphrey, and Juan P. Bello, "A software framework for musical data augmentation," in *ISMIR 2015*, 2015, pp. 248–254.

[11] Navdeep Jaitly and Geoffrey E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in *ICML 2013*, 2013.

[12] Naoya Takahashi, Michael Gygli, Beat Pfister, and Luc Van Gool, "Deep convolutional neural networks and data augmentation for acoustic event recognition," in *INTERSPEECH 2016*, September 2016.

[13] Yuji Tokozume, Yoshitaka Ushiku, and Tatsuya Harada, "Learning from between-class examples for deep sound recognition," in *ICLR 2018*, 2018.

[14] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz, "mixup: Beyond empirical risk minimization," in *ICLR 2018*, 2018.

[15] Gert Dekkers, Lode Vuegen, Toon van Waterschoot, Bart Vanrumste, and Peter Karsmakers, "DCASE 2018 Challenge - Task 5: Monitoring of domestic activities based on multi-channel acoustics," Tech. Rep., KU Leuven, July 2018.

[16] Tadanobu Inoue, Phongtharin Vinayavekhin, Shiqiang Wang, David Wood, Nancy Greco, and Ryuki Tachibana, "Domestic activities classification based on CNN using shuffling and mixing data augmentation," Tech. Rep., DCASE 2018 Challenge, September 2018, non-peer-reviewed technical report.

[17] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello, "A dataset and taxonomy for urban sound research," in *Proc of 22nd ACM international conference on Multimedia*, 2014, pp. 1041–1044.

[18] Diederik P. Kingma and Jimmy Lei Ba, "Adam: a method for stochastic optimization," in *ICLR 2015*, 2015.

[19] Rui Lu, Zhiyao Duan, and Changshui Zhang, "Metric learning based data augmentation for environmental sound classification," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2017.

DISTILLING THE KNOWLEDGE OF SPECIALIST DEEP NEURAL NETWORKS IN ACOUSTIC SCENE CLASSIFICATION

Jee-weon Jung[*], *Hee-Soo Heo*[*], *Hye-jin Shim, and Ha-Jin Yu*[†]

School of Computer Science, University of Seoul, South Korea

ABSTRACT

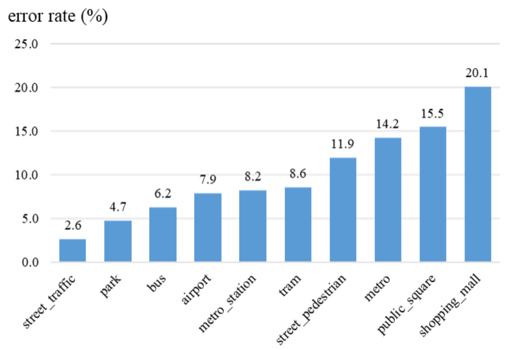
Different acoustic scenes that share common properties are one of the main obstacles that hinder successful acoustic scene classification. Top two most confusing pairs of acoustic scenes, ‘airport-shopping_mall’ and ‘metro-tram’ have occupied more than half of the total misclassified audio segments, demonstrating the need for consideration of these pairs. In this study, we exploited two specialist models in addition to a baseline model and applied the knowledge distillation framework from those three models into a single deep neural network. A specialist model refers to a model that concentrates on discriminating a pair of two similar scenes. We hypothesized that knowledge distillation from multiple specialist models and a pre-trained baseline model into a single model could gather the superiority of each specialist model and achieve similar effect to an ensemble of these models. In the results of the Detection and Classification of Acoustic Scenes and Events 2019 challenge, the distilled single model showed a classification accuracy of 81.2 %, equivalent to the performance of an ensemble of the baseline and two specialist models.

Index Terms— Acoustic scene classification, Specialist models, Knowledge distillation, Teacher-student learning, Deep neural networks

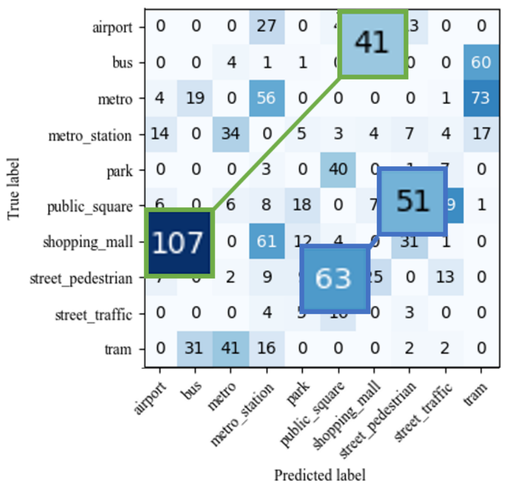
1. INTRODUCTION

Recently, various studies on acoustic scene classification (ASC) systems have been being conducted upon increasing demand from several different industries. The Detection and Classification of Acoustic Scenes and Events (DCASE) challenge is providing a common platform for various studies to compare and examine proposed methods [1, 2]. Based on the efforts of the organizers of the challenge, many different types of research have been conducted to improve the performances of ASC systems. In [3], a sophisticated training procedure for an ASC system was proposed. Other studies have focused mainly on investigating feature extraction and data augmentation techniques for ASC tasks [4, 5]. With such studies and the annual DCASE challenge, the performance of ASC systems has incrementally increased each year. However, to our knowledge, there have been few studies that have analyzed errors that occur due to the characteristics of the ASC task. We believe that such an analysis of the task errors is necessary in addition to designing an elaborate system.

In ASC tasks, common acoustic properties among the different acoustic scenes are a known obstacle that degrade the performance of developed systems [6]. These acoustic properties evoke a few frequently misclassified pairs of acoustic scenes. For more details, Figure 1 (a) shows the proportion of each class among the total misclassified audio segments that use the baseline ASC system. In this figure, we verified that the three most difficult classes to identify occupy more than half of the total error. In addition, Figure 1 (b) shows that most of the errors from the frequently misclassified classes are due to a certain confusing pair. For example, most of the errors from the two classes ‘public_square’ and ‘street_pedestrian,’ which were frequently misclassified classes, were caused by the



(a)



(b)

Figure 1: (a) The proportion of each class among the total errors of the baseline model (b) Illustration of the confusion matrix from the baseline model.

[*]Equal contribution.
[†] Corresponding author.

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT & Future Planning(2017R1A2B4011609)

https://doi.org/10.33682/gqpj-ac63

confusion between each other.

The phenomenon of a few misclassified acoustic scenes severely degrading the overall performance has been alleviated by adopting a knowledge distillation scheme with a *'soft-label'* that models the common properties of acoustic scenes [6]. In this study, we further alleviated this problematic phenomenon by using a *'specialist'* scheme. A specialist model refers to a model that concentrates more on a specific pair of frequently misclassified classes. However, when adopting specialist models, there are still some issues, such as the growing number of parameters (model capacity) and the number of required specialists. To overcome these issues, we further utilized the knowledge distillation scheme combined with the specialist models.

The scheme used in this study distilled the knowledge from the baseline model and two specialist models into a student model. In this scheme, the number of parameters in the distilled model was identical to that in the baseline model. Experimental results on the DCASE 2019 task 1 competition demonstrated that one distilled models shows a performance equal to that of the ensemble of all other models. The main contributions of this paper can be summarized as follows:

1. Adoption of specialist models for frequently misclassified pairs of acoustic scenes.
2. Application of knowledge distillation from a baseline model and two specialist models into one single distilled model in acoustic scene classification.

The rest of this paper is organized as follows: the knowledge distillation (also referred to as teacherstudent learning) framework is introduced in Section 2. Section 3 describes the specialist models and how it is used in this study. The experimental settings and results are detailed in Sections 4 and 5, respectively, and the paper is concluded in Section 6.

2. KNOWLEDGE DISTILLATION IN THE ASC TASK

Knowledge distillation (KD) is a framework where the '*soft-label*' extracted from a DNN is used to train the other DNN (this framework is also referred to as the teacher-student framework) [7, 8]. We refer to the DNN that provides the soft-label as the teacher DNN, and the DNN that is trained using the soft-label is referred to as the student DNN for clarity throughout this paper.

The KD framework was conducted with the following steps. First, a teacher DNN was trained using the categorical cross-entropy (CCE) objective function. After training of the teacher DNN was complete, its parameters were frozen, and only used for providing soft-labels, which were used to train the student DNN. Note that we initialized the student DNN using the parameters from the teacher DNN. The KD framework has been successfully applied to many tasks [9, 10]. It is important to design the teacher DNN to be superior by considering the work flow of the KD framework that trains the student DNN using the output of the teacher DNN. For example, a larger capacity for a model compression task [8], or close talk utterance input for far-field compensation [10] make the teacher DNN superior.

In the ASC task, Heo *et al.* [6] first adopted the KD framework to model the common properties among different acoustic scenes using soft-labels. For example, babbling sounds that occur in both shopping_mall and airport (pre-defined acoustic scenes of the DCASE 2019 challenge) are sometimes labeled as shopping_mall but labeled as airport at other times using a hard-label scheme. Using the KD framework, soft-labels were hypothesized to model these correlations between pre-defined labels based on their common acoustic properties. This approach was successful in that not only was the overall classification accuracy increased but also the number of misclassified audio segments in the most frequently misclassified pair of scenes significantly decreased.

3. KNOWLEDGE DISTILLATION WITH SPECIALIST MODELS

3.1. Specialist Knowledge Distillation

In the KD framework that involves specialist models [7], soft-labels extracted from multiple teacher DNNs were exploited to train a student DNN. In this framework, multiple teacher DNNs comprise one baseline model and a defined number of specialist models. Here, the specialist model refers to a DNN that classifies a subset of classes assigned by a clustering algorithm (e.g. in [7], 300 detailed classes that are in the bird category were set to a specialist model among a total of 15000 categories from *Google*'s internal dataset).

The training process of specialist knowledge distillation is as follows. First, we train the baseline model using a CCE objective function. Next, a defined number of specialist models are initialized using the weight parameters of the baseline model (DNN architecture is identical except for the output layer). Each specialist model is then trained using the CCE objective function with defined subset labels. Finally, the student DNN is trained using multiple soft-labels each extracted from the baseline and specialist models. The loss function $\mathcal{L}_{KD}$ for the training of the student DNN model can be defined as follows:

$$\mathcal{L}_{KD}(\theta;\theta_b,S) = -\sum_{i=1}^{N}\sum_{j=1}^{M} logQ(j|\boldsymbol{x}_i;\theta)[Q(j|\boldsymbol{x}_i;\theta_b) + \sum_{\theta_s\in S} Q(j|\boldsymbol{x}_i;\theta_s)], \tag{1}$$

$$Q(j|\boldsymbol{x};\theta) = \frac{exp(z_j/T)}{\sum_i exp(z_i/T)}, \tag{2}$$

where N and M denote the size of the mini-batch and the acoustic scenes in the training set, respectively, each input audio segment is referred to as $\boldsymbol{x}_i$, $Q(j|\boldsymbol{x}_i,\theta)$ denotes the posterior probability for the $j'th$ acoustic scene using the concept of temperature T [7], θ_b is the set of parameters in the baseline model, θ_s is the parameter set of specialist model s, and S is the set of specialist models. The function $Q(j|\boldsymbol{x}_i,\theta)$, defined in Eq. (2), has the role of smoothing the results of applying the softmax function to the output $\boldsymbol{z}$ of the DNN. A loss function, $\mathcal{L}_{KD}$, has been proposed to train the single model that can achieve the same as the ensemble of models that have different characteristics [7, 8].

3.2. Specialist Knowledge Distillation for the ASC Task

In this sub-section, we introduce the modifications we make on the knowledge distillation framework with specialists to suit the ASC task. First, we fixed the number of classes to classify rather than selecting a subset of classes. In Hinton *et al.* [7], the number of total classes was too large, making selection of a subset of classes necessary. However, the DCASE 2019 challenge dataset defines ten classes.

Second, in our configuration, one specialist model concentrated on classifying one pair of frequently misclassified acoustic scenes.

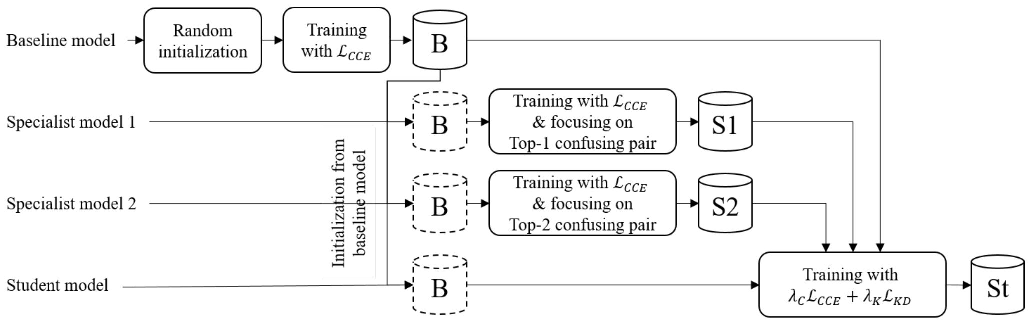

Baseline model
Random initialization
Training with $\mathcal{L}_{CCE}$
B
Specialist model 1
Training with $\mathcal{L}_{CCE}$ & focusing on Top-1 confusing pair
S1
Specialist model 2
Training with $\mathcal{L}_{CCE}$ & focusing on Top-2 confusing pair
S2
Initialization from baseline model
Student model
Training with $\lambda_C\mathcal{L}_{CCE} + \lambda_K\mathcal{L}_{KD}$
St

Figure 2: Workflow of the training procedure.

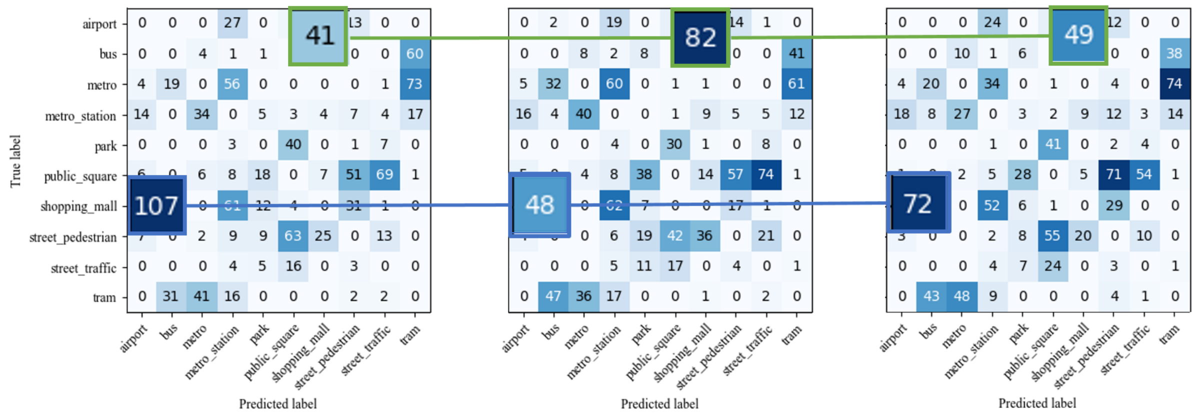


Figure 3: Illustration of the performance analysis based on confusion matrices (right: baseline model, middle: first specialist model, right: student model).

We use two specialist models, where the pair of acoustic scenes to concentrate on is decided based on the confusion matrix of the baseline model; the top two most confusing pairs of acoustic scenes are dealt with two specialist models, respectively. This configuration is based on the analysis that few frequently misclassified acoustic scenes occupy the majority of misclassified samples (see Figure 1). To train the specialist model, we construct half of the mini-batch with target pairs to concentrate on, and the other half with pairs of randomly selected samples from other classes.

After training the specialist models, we train the student model using an objective function composed of the function defined in Eq. (1) and the CCE function, as follows:

$$\mathcal{L} = \lambda_C \mathcal{L}_{CCE} + \lambda_K \mathcal{L}_{KD}, \tag{3}$$

where λ_C and λ_K are the weights of $\mathcal{L}_{CCE}$ and $\mathcal{L}_{KD}$, respectively. The CCE function defined by the true label is used to correct errors that may occur in the teacher models. The values of the two weight coefficients were fixed based on the validation results on the DCASE2019 fold-1 configuration.

The overall training process for the framework used in our study is illustrated in Figure 2.

By applying knowledge distillation using the specialist models, we expect two results. First, class-wise accuracy of the top misclassified acoustic scenes should decrease. Second, the superiority of each specialist model regarding a target pair of acoustic scenes should be well distilled into a single student DNN. To observe whether this objective is successfully achieved, we analyze not only the overall accuracy but also the class-wise accuracies and the number of misclassified samples between each pair of target acoustic scenes that the specialist focused on.

4. EXPERIMENTAL SETTINGS

We conducted all experiments using PyTorch, a deep learning library written in Python [11][1].

We used the Detection and Classification of Acoustic Scenes and Events Challenge Task 1-a dataset for all our experiments. This dataset comprises audio segments that were 10 s and were recorded at 48 kHz with 24-bit resolution; each stereo segment was labelled as one of the pre-defined ten acoustic scenes. The dataset was divided into a development and an evaluation set, where the development set comprised 14400 labelled audio segments, and the evaluation set was not revealed. We constructed a four-fold cross-validation setup using all the data and independently trained four systems. The first fold followed the configuration from the DCASE2019 challenge organizer, and the remaining folds were constructed, taking into account the city where each audio segment was recorded.

We built two separate DNNs for each configuration, where one input raw waveforms and the other input log Mel-energy features. In particular, the model for the raw waveform inputs was con-

[1]Codes used for experiments are available at
`https://github.com/Jungjee/dcase2019specialistkd`

structed following the ResNet architecture based on 1-D convolutional layers, and the model for Mel-energy was constructed following the ResNet architecture based on 2-D convolutional layers [12, 13]. We exploited a score-level ensemble in which one uses a CNN that inputs raw waveforms and the other uses a CNN that inputs log Mel-energy features. For data augmentation, we applied a mix-up technique [14] defined as

$$\hat{\boldsymbol{x}} = \lambda \boldsymbol{x}_i + (1-\lambda)\boldsymbol{x}_j, \tag{4}$$

$$\hat{\boldsymbol{y}} = \lambda \boldsymbol{y}_i + (1-\lambda)\boldsymbol{y}_j, \tag{5}$$

$$\lambda = B(\alpha, \alpha), \tag{6}$$

where the pair of $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ represent a set of randomly selected input utterances and the corresponding label, respectively, and $B(\alpha,\alpha)$ is the beta distribution with coefficient α [14]. Label $\boldsymbol{y}_i$ is defined by the true label when training with the CCE function and is referred to the output of the teacher DNN when applying the KD framework.

Refer to the authors' technical report [15] for other details regarding the input features, model architectures, and training procedures[2].

5. RESULTS ANALYSIS

Figure 3 depicts the change of mis-classified samples regarding the most confusing pair ('shopping_mall' and 'airport') in three confusion matrices of the baseline, first specialist, and the student (distilled) model. The number of mis-classified samples of the most confusing pair in these three models was 148, 130, and 121 respectively. Comparing the baseline and the specialist, this result first demonstrates that the mis-classified number of audio segments in target confusing pair decrease in the specialist model than the baseline. However, the overall classification accuracy was similar (for Mel-energy, baseline was 74.33 % and the specialist1 was 74.12 %). Comparing the specialist and the student model, the result of mis-classified samples from 130 to 121 shows that not only the overall accuracy increases, but the knowledge of the specialist model is successfully distilled.

To verify whether the superiority of each specialist model was actually distilled to the student DNN, we analyzed the accuracy of the overall and top two most frequently confusing pairs of acoustic scenes. Figure 4 shows the results. Note that these results were from the fold-1 and Mel-energy configuration. We found that the overall accuracy of the student DNN was actually higher than those of all other models. Additionally, we confirmed that for each specialist model the class-wise accuracy of the concentrated pairs increased while the accuracies of other pairs decreased, resulting in similar overall accuracy. The class-wise accuracy of the most confusing pairs in the student model is equal to or higher to those that were the focus of each specialist model. According to this result, we concluded that the designed superiority of each specialist was well distilled to the student DNN. The additional results on the fold-1 configuration are demonstrated in Table 1.

The success of the knowledge distillation is further addressed in Table 2, which reports overall classification accuracies on the evaluation set. This table shows the performances on the evaluation set according to the score-level ensemble methods. The ensemble of 'B+S1+S2+St' means combining the outputs from 32 models (four kinds of models × two fold configurations × two types of input features), and the ensemble of 'St' refers to combining the outputs from eight models. The performance of the student DNN was the same as that of the ensemble of the baseline and two specialist models. This result also verifies that the student DNN trained with specialist knowledge distillation better conducted the ASC task with less number of parameters.

[2]http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Jung_98.pdf

Table 1: Performances of various systems with the fold-1 configuration according to their accuracies (%) (B: baseline model, S1: $1'th$ specialist model, S2: $2'nd$ specialist model, St: student model).

System	B	S1	S2	St
Raw waveform	73.71	74.89	74.53	75.81
Mel-energy	74.33	74.12	74.48	76.15

Table 2: Performances of various systems with the evaluation configuration according to their accuracies (%) (B: baseline model, S1: $1'th$ specialist model, S2: $2'nd$ specialist model, St: student model).

Systems	B+S1+S2+St	St
Accuracy (%)	81.2	81.2

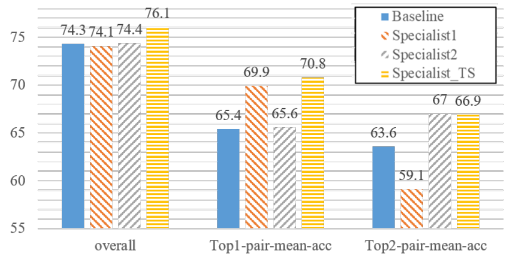


Figure 4: Illustration of the performance analysis based on the mean accuracy of the two classes from the most confusing pairs.

6. CONCLUSION

In this study, we observed that a few pairs of frequently misclassified acoustic scenes occupy more than half of the total misclassified audio segments in an ASC task. For addressing the issue, we adopted the concept of the specialist model, which was designed to concentrate on specific subsets of a task. We modified and trained the specialist models to suit the ASC task. The results show that the specialist model could have not only the superiority that reduces errors for certain confusing pairs but also the inferiority that decreases the discriminative power for other classes. We hypothesized that the KD framework could achieve the identical effect with the ensemble of multiple models by combining superiority into a single model, excluding the inferiority of individual models. The experimental results demonstrated that the KD framework was successful, coherent to our hypothesis and it resulted in overall performance improvements for the ASC system.

7. REFERENCES

[1] A. Mesaros, T. Heittola, and T. Virtanen, “A multi-device dataset for urban acoustic scene classification,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 9–13. [Online]. Available: https://arxiv.org/abs/1807.09840

[2] A. Mesaros, A. Diment, B. Elizalde, T. Heittola, E. Vincent, B. Raj, and T. Virtanen, “Sound event detection in the DCASE 2017 challenge,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2019, in press.

[3] M. Dorfer and G. Widmer, “Training general-purpose audio tagging networks with noisy labels and iterative self-verification,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 178–182.

[4] H. Zeinali, L. Burget, and J. H. Cernocky, “Convolutional neural networks and x-vector embedding for DCASE2018 acoustic scene classification challenge,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 202–206.

[5] J.-W. Jung, H.-S. Heo, I.-H. Yang, S.-H. Yoon, H.-J. Shim, and H.-J. Yu, “DNN-based audio scene classification for DCASE2017: Dual input features, balancing cost, and stochastic data duplication,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, November 2017, pp. 59–63.

[6] H.-S. Heo, J.-w. Jung, H.-j. Shim, and H.-J. Yu, “Acoustic scene classification using teacher-student learning with soft-labels,” in *INTERSPEECH*, 2019.

[7] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015.

[8] J. Li, R. Zhao, J.-T. Huang, and Y. Gong, “Learning small-size dnn with output-distribution-based criteria,” in *Fifteenth annual conference of the international speech communication association*, 2014.

[9] J.-w. Jung, H.-s. Heo, H.-j. Shim, and H.-j. Yu, “Short utterance compensation in speaker verification via cosine-based teacher-student learning of speaker embeddings,” *arXiv preprint arXiv:1810.10884*, 2018.

[10] J. Li, R. Zhao, Z. Chen, C. Liu, X. Xiao, G. Ye, and Y. Gong, “Developing far-field speaker system via teacher-student learning,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5699–5703.

[11] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, “Automatic differentiation in pytorch,” *NIPS-W*, 2017.

[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[13] ——, “Identity mappings in deep residual networks,” in *European conference on computer vision*. Springer, 2016, pp. 630–645.

[14] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” *arXiv preprint arXiv:1710.09412*, 2017.

[15] J.-w. Jung, H.-S. Heo, H.-j. Shim, and H.-J. Yu, “Knowledge distillation with specialist models in acoustic scene classification,” DCASE2019 Challenge, Tech. Rep., June 2019.

SOUND SOURCE DETECTION, LOCALIZATION AND CLASSIFICATION USING CONSECUTIVE ENSEMBLE OF CRNN MODELS

Sławomir Kapka[*], *Mateusz Lewandowski*

Samsung R&D Institute Poland
Artificial Intelligence
Warsaw, 00-844, Poland
{s.kapka, m.lewandows4}@samsung.com

ABSTRACT

In this paper, we describe our method for DCASE2019 task 3: Sound Event Localization and Detection (SELD). We use four CRNN SELDnet-like single output models which run in a consecutive manner to recover all possible information of occurring events. We decompose the SELD task into estimating number of active sources, estimating direction of arrival of a single source, estimating direction of arrival of the second source where the direction of the first one is known and a multi-label classification task. We use custom consecutive ensemble to predict events' onset, offset, direction of arrival and class. The proposed approach is evaluated on the TAU Spatial Sound Events 2019 - Ambisonic and it is compared with other participants' submissions.

***Index Terms*—** DCASE 2019, Sound Event Localization and Detection, CRNN, Ambisonics

1. INTRODUCTION

Sound Event Localization and Detection (SELD) is a complex task which naturally appears when one wants to develop a system that possesses spatial awareness of the surrounding world using multi-channel audio signals. This year, the task 3 from the IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE 2019) [1] concerned the SELD problem. SELDnet introduced in [2] is a single system of a good quality designed for the SELD task, and the slight modification of SELDnet was set as the baseline system [3] during the DCASE 2019 Challenge. Solely based on [2] and [3], we develop a novel system designed for the task 3 from the DCASE 2019 Challenge.

In our work, we follow the philosophy that if a complex problem can be split into simpler ones, one should do so. Thus we decompose the SELD task with up to 2 active sound sources into the following subtasks:

- estimating the number of active sources (*noas*),
- estimating the direction of arrival of a sound event when there is one active sound source (*doa1*),
- estimating the direction of arrival of a sound event when there are two active sound sources and we posses the knowledge of the direction of arrival of one of these sound events, which we will call an *associated event* (*doa2*),
- multi-label classification of sound events (*class*).

[*]Corresponding author.

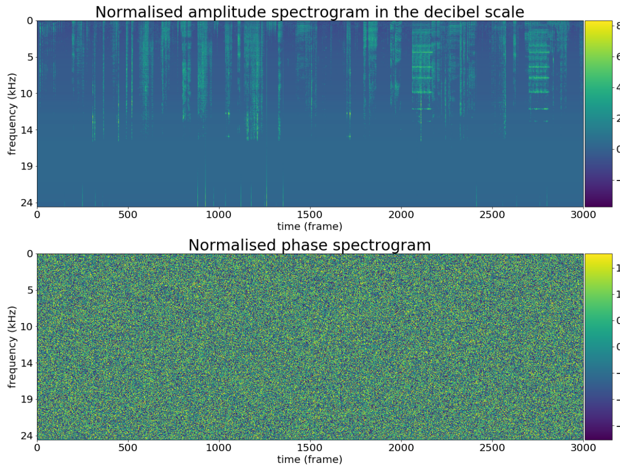


Figure 1: An example of the normalised amplitude spectrogram in the decibel scale and the normalised phase spectrogram obtained from the first *foa* channel from some randomly selected recording. The horizontal and vertical axes denote frame numbers and frequencies respectively obtained from the STFT. Note that the values from the legends on the right are dimensionless due to the normalization used in the preprocessing.

For each of this subtasks, we develop a SELDnet-like convolutional recurrent neural network (CRNN) with a single output. We discuss it in detail in section 3. Given such models, we develop a custom consecutive ensemble of these models. This allows us to predict the events' onset, offset, direction of arrival and class, which we discuss in detail in section 4. Due to the sequential nature of generating predictions in our system, errors in models' predictions may cascade, and thus an overall error may cumulate. Despite this drawback, our system acquire very good results on the TAU Spatial Sound Events 2019 - Ambisonic database. We discuss the results in detail in section 5.

2. FEATURES

The DCASE 2019 task 3 provides two formats of the TAU Spatial Sound Events 2019 dataset: first order ambisonic (*foa*) and 4 channels from a microphone array (*mic*) [3]. In our method we only use the ambisonic format.

https://doi.org/10.33682/9f2t-ab23

Each recording is approximately 1 minute long with sampling rate of 48k. We use the short time Fourier transform (STFT) with Hann window. We use the window of length 0.4s and hop of length 0.2s in STFT to transform a raw audio associated to each *foa* channel into the complex spectrogram of size 3000x1024. If audio is longer than 1 minute, we truncate spectrograms. If an audio is shorter than 1 minute, we pad them with zeros.

From each complex spectrogram we extract its module and phase point-wise, that is amplitude and phase spectrograms, respectively. We transform amplitude spectrograms to the decibel scale. Finally, we standardize all spectrograms frequency-wise to zero mean and unit variance, to obtain spectrograms as in Figure 1.

In summary, from each recording we acquire 4 standardized amplitude spectrograms in the decibel scale and 4 standardized phase spectrograms corresponding to 4 *foa* channels.

3. ARCHITECTURE

As mentioned in the introduction, each of the subtasks (*noas*, *doa1*, *doa2* and *class*) has its own SELDnet-like CRNN. Each of these models is a copy of a single SELDnet node with just minor adjustments so that it fits to the specific subtask and for the regularization purpose.

Each of these models takes as an input a fixed length subsequence of decibel scale amplitude spectrograms (in case of *noas* and *class* subtasks) or both decibel scale amplitude and phase spectrograms (in case of *doa1* and *doa2* subtasks) from all 4 channels.

In each case, the input layers are followed by 3 convolutional layer blocks. Each block is made of a convolutional layer, batch norm, relu activation, maxpool and dropout. The output from the last convolutional block is reshaped so that it forms a multivariate sequence of a fixed length. In the case of *doa2*, we additionaly concatenate directions of arrivals of associated events with this multivariate sequence. Next, there are two recurrent layers (GRU or LSTM) with 128 units each with dropout and recurrent dropout. Next layer is a time distributed dense layer with dropout and with the number of units depending on subtask.

Lastly, depending on a subtask, the model has a different output. For *noas*, the model has just a single time distributed output that corresponds to the number of active sources (0, 1 or 2). For *doa1* and *doa2*, the models have 3 time distributed outputs that corresponds to cartesian xyz coordinates as in [2]. Cartesian coordinates are advantageous over spherical coordinates in this task due to their continuity. Lastly, for *class*, the model has 11 time distributed outputs corresponding to 11 possible classes. We present the detailed architecture in Table 1.

Depending on a subtask, we feed the network with the whole recordings or just their parts. For *noas*, we feed all the data. For *doa1*, we extract only those parts of the recordings where there is just one sound source active. For *doa2*, we extract only those parts of the recordings where there are exactly two active sound sources. For *class*, we extract those parts of the recordings where there are at least one active source.

As for the learning process, we used mean square error loss for the *noas*, *doa1*, *doa2* subtasks and binary cross-entropy loss for the *class* subtask. For all subtasks we initialised learning process using Adam optimizer with default parameters [4]. The *noas* and *class* subtasks were learned for 500 epochs with exponential learning rate decay; every 5 epochs the learning rate were multiplied by 0.95. In *doa1* and *doa2* subtasks, we run learning process for 1000 epochs without changing the initial learning rate.

As for complexity, the *noas*, *doa1*, *doa2* and *class* have 572,129, 753,603, 591,555 and 572,299 parameters respectively, making total of 2,651,634 parameters.

4. CONSECUTIVE ENSEMBLE

In this section, we introduce and describe the idea of the consecutive ensemble which is the core of our approach. This custom binding of our four models allows us to predict the events' onset, offset, direction of arrival and class.

4.1. The algorithm

We assume that recordings have at most 2 active sound sources at once and the sound events occur on a 10 degrees resolution grid. In our setting, the audios after feature extraction have exactly 3000 vectors corresponding to the time dimension. Henceforth we will call these vectors as frames. The algorithm itself goes as follows:

1. We feed the features to the *noas* network to predict the number of active sources (NOAS) in each frame.

2. We transform the predicted NOAS so that each recording starts and ends with no sound sources and the difference of NOAS between each frames is no greater than 1.

3. From the predicted NOAS we deduce the number of events, their onsets and the list of possible offsets for each event. If NOAS in two consecutive frames increases, then we predict that a new event happened at the second frame. If in two consecutive frames NOAS decreases, then we append the first frame to all events since last time NOAS was 0 as a possible offset.

4. In order to determine which offset corresponds to which event we use the *doa1* network. We extract chunks (intervals of equal NOAS) of audio where the predicted NOAS equals 1 and we feed it to *doa1* network. For each chunk where NOAS was 1 we predict the average azimuth and elevation, and we round it to the closest multiple of 10. If two consecutive chunks have the same azimuth and elevation then we conclude that the first event covered two chunks and the second event started and ended between those chunks. If two consecutive chunks have a different azimuth or elevation, then we conclude that the first event ended when the second chunk started and the second event continued in the second chunk.

5. To determine the remaining information about angles we need to predict the direction of arrival (DOA) of events that start and end while the associated event is happening. We feed the chunks where NOAS is 2 to the *doa2* network with the second input being DOA of the associated event in cartesian xyz coordinates. Similarly as in step 4, we average the predicted results from chunks and round it to the closest multiple of 10.

6. Lastly, we predict the events' classes. If an event has chunks where the event is happening in an isolation (NOAS = 1), then all such chunks are feed to the *class* network and the most probable class (using soft voting among frames) is taken as a predicted class. If an event has no such chunks, i.e. the event is only happening with an associated event, then such chunk (NOAS = 2) is fed to the network and two most probable classes are extracted. We choose the first one which does not equal to the class of the associated event.

4.2. An example

The algorithm itself may seem quite complex at first glance. Hence, we investigate here a concrete example.

Table 1: The architecture and the parameters of the networks

Layer Type	**Parameters**	*noas*	*doa1*	*doa2*	*class*
Input	Shape	$256 \times 1024 \times 4$	$128 \times 1024 \times 8$	$128 \times 1024 \times 8$	$128 \times 1024 \times 4$
*ConvBlock**	Pool	8	8	8	8
*ConvBlock**	Pool	8	8	8	8
*ConvBlock**	Pool	4	4	4	4
Reshape	Sequence length $\times$ features	256×-1	128×-1	128×-1	128×-1
Doa2 input	Is used	False	False	True	False
Concatenate	Is used	False	False	True	False
*RecBlock***	Unit type	GRU	LSTM	GRU	GRU
*RecBlock***	Unit type	GRU	LSTM	GRU	GRU
TD Dense	Number of units	16	128	128	16
Dropout	Dropout rate	0.2	0.2	0.2	0.2
TD Dense	Number of units	1	3	3	11
Activation	Function	linear	linear	linear	sigmoid
**ConvBlock(P)*					
Conv2D	64 filters, 3×3 kernel, 1×1 stride, same padding				
BatchNorm	—				
Activation	ReLu function				
MaxPooling2D	$1 \times P$ pooling				
Dropout	0.2 dropout rate				
***RecBlock(U)*					
Recurrent	128 recurrent units of type U, 0.2 recurrent dropout rate				
Activation	tanh function				
Dropout	0.2 dropout rate				

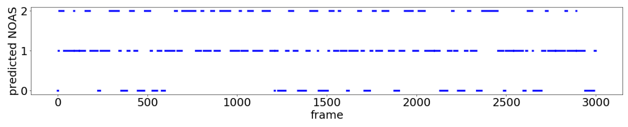


Figure 2: The plot visualising the predicted number of active sources for some randomly selected recording.

Given a recording constituting of 3000 vectors, we predict its NOAS in each frame as in Figure 2. For the sake of clarity we constrain only to a part of the recording. Consider a block with predicted NOAS as in the top plot from Figure 3. According to the step 3 from the algorithm, we predict that 3 events happened here: E_1, E_2, E_3 with 3 corresponding onsets On_1, On_2, On_3. Events E_1 and E_2 may end at Off_1, Off_2 or Off_3 and event E_3 may end at Off_2 or Off_3 (see the bottom plot from Figure 3). According to the step 4 from the algorithm, we predict DOA using *doa1* in chunks from On_1 to On_2, from Off_1 to On_3 and from Off_2 to Off_3. Based on that we deduce the events' offsets as in Figure 3. Based on step 5 from the algorithm, we predict the DOA of chunk from On_3 to Off_2 using *doa2* where the associated DOA is the DOA of E_2. Lastly we deduce classes of the events E_1, E_2 and E_3. According to the step 6 form the algorithm, we predict class of E_1 based on the chunk from On_1 to On_2, predict the class of E_2 based on chunks from Off_1 to On_3 and from Off_2 to Off_3. Finally, we predict the class of E_3 based on the chunk from On_3 to Off_2. If the predicted class of E_3 is the same as the class of E_2 then we predict it to be the second most probable class from the *class* network.

Table 2: The average results from all 4 splits.

	Error rate	F-score	DOA error	Frame recall	Seld score
Train	0.03	0.98	2.71	0.98	0.02
Val.	0.15	0.89	4.81	0.95	0.08
Test	0.14	0.90	4.75	0.95	0.08
Baseline	0.34	0.80	28.5	0.85	0.22

5. RESULTS

We evaluate our results on TAU Spatial Sound Events 2019 - Ambisonic dataset. This dataset constitutes of two parts: the development and evaluation sets. The development part consists of 400 recordings with predefined 4-fold cross-validation and the evaluation part consists of 100 recordings. The results from this section relate to our submission `Kapka_SRPOL_task3_2`.

5.1. Development phase

As for the development part, we used 2 splits out of 4 for training for every fold using the suggested cross-validation even though validation splits do not influence the training process.

We show in Table 2 the averaged metrics from all folds for our setting and metrics for the baseline [3]. In order to demonstrate the variance among folds, we present in Table 3 the detailed results on the test splits from each fold. The development set provides the distinction for the files where there is up to 1 active sound source at once (ov1) and where there are up to 2 (ov2). In Table 4 we compare metrics for the ov1 and ov2 subsets.

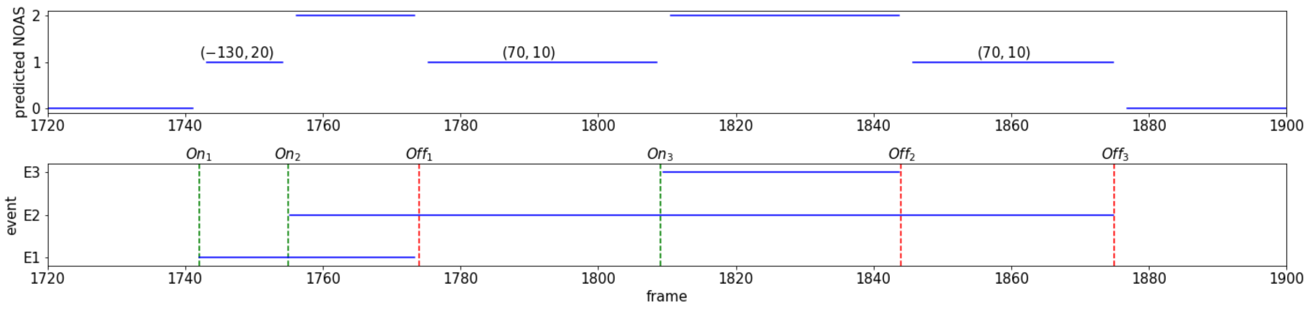


Figure 3: Given the predicted NOAS from the part of some recording as in the top plot, we deduce that there are 3 events E_1, E_2 and E_3 with corresponding onsets denoted by the green lines in the bottom plot. Based on the predicted DOA, which we placed in the top plot above the segments, we deduce the events' offsets denoted by the red lines in the bottom plot.

Table 3: The results on the test splits from each fold.

	Error rate	F-score	DOA error	Frame recall	Seld score
Split 1	0.13	0.91	6.01	0.95	0.07
Split 2	0.16	0.88	6.01	0.95	0.09
Split 3	0.11	0.93	4.93	0.96	0.06
Split 4	0.17	0.86	5.89	0.96	0.10

Table 4: The results on the ov1 and ov2 subsets.

	Error rate	F-score	DOA error	Frame recall	Seld score
ov1	0.07	0.94	1.28	0.99	0.04
ov2	0.18	0.87	7.96	0.93	0.11

5.2. Official results

For the evaluation part, we used all 4 splits for training from the development set. We compare our final results with the selected submissions in Table 5.

The idea of decomposing the SELD task into simpler ones proved to be a very popular idea among contestants. The recent two-stage approach to SELD introduced in [5] was used and developed further by many. The best submission using two-step approach `Cao_Surrey_task3_4` [6] obtained results very similar to ours. `He_THU_task3_2` [7] and `Chang_HYU_task3_3` [8] outperform our submission in SED metrics and DOA error respectively. However, our approach based on estimating NOAS first allows us to outperform all contestants in frame recall.

6. SUBMISSIONS

Overall, we created 4 submissions for the competition:

- ConseqFOA (`Kapka_SRPOL_task3_2`),
- ConseqFOA1 (`Kapka_SRPOL_task3_3`),
- ConseqFOAb (`Kapka_SRPOL_task3_4`),
- MLDcT32019 (`Lewandowski_SRPOL_task3_1`).

Table 5: The comparison of the selected submissions.

Rank	**Submission name**	**Error rate**	**F-score**	**DOA error**	**Frame recall**
1	`Kapka_SRPOL_task3_2`	0.08	94.7	3.7	**96.8**
4	`Cao_Surrey_task3_4`	0.08	95.5	5.5	92.2
6	`He_THU_task3_2`	**0.06**	**96.7**	22.4	94.1
19	`Chang_HYU_task3_3`	0.14	91.9	**2.7**	90.8
48	`DCASE2019_FOA_baseline`	0.28	85.4	24.6	85.7

The first three submissions use the approach described in the above sections. The only difference is that ConseqFOA is trained on all four splits from development dataset. ConseqFOA1 is trained on splits 2,3,4. ConseqFOAb is trained on all splits but the classifier in this version was trained using categorical cross-entropy instead of binary cross-entropy loss.

Our MLDcT32019 submission uses a different approach. It works in the same way as the original SELDnet architecture but with the following differences:

- We implemented the Squeeze-and-Excitation block [9] after the last convolutional block. We pass the output from the last convolutional block through two densely connected neural layers with respectively 1 and 4 neurons, we multiply it with the output of the last convolutional block and we pass it further to recurrent layers.
- We set all dropout rates to 0.2.
- We used SpecAugment [10] as an augmentation technique to double the training dataset.
- We replaced recurrent layer GRU units with LSTM units.

7. CONCLUSION

We conclude that decomposing the SELD problem into simpler tasks is instinctive and efficient. However, we are aware that our solution has some serious limitations and it fails when one wants to consider a more general setup. For example when there are more than 2 active sources at once or when the grid resolution is more refined. Thus, we claim that the pursuit for universal and efficient SELD solutions is still open.

8. ACKNOWLEDGEMENT

We are most grateful to Zuzanna Kwiatkowska for spending her time on careful reading with a deep understanding the final draft of this paper.

9. REFERENCES

[1] http://dcase.community/challenge2019/task-sound-event-localization-and-detection.

[2] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–1, 2018.

[3] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and uetection," in *Submitted to Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019. [Online]. Available: https://arxiv.org/abs/1905.08546

[4] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[5] Y. Cao, Q. Kong, T. Iqbal, F. An, W. Wang, and M. D. Plumbley, "Polyphonic sound event detection and localization using a two-stage strategy," 2019. [Online]. Available: https://arxiv.org/abs/1905.00268

[6] Y. Cao, T. Iqbal, Q. Kong, M. Galindo, W. Wang, and M. Plumbley, "Two-stage sound event localization and detection using intensity vector and generalized cross-correlation," 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Cao_74.pdf

[7] J. Zhang, W. Ding, and L. He, "Data augmentation and prior knowledge-based regularization for sound event localization and detection," 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_He_97.pdf

[8] K. Noh, C. Jeong-Hwan, J. Dongyeop, and C. Joon-Hyuk, "Three-stage approach for sound event localization and detection," 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Chang_81.pdf

[9] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018, pp. 7132–7141.

[10] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," 2019. [Online]. Available: https://arxiv.org/abs/1904.08779

RECEPTIVE-FIELD-REGULARIZED CNN VARIANTS FOR ACOUSTIC SCENE CLASSIFICATION

Khaled Koutini[1], *Hamid Eghbal-zadeh*[1,2], *Gerhard Widmer*[1,2]

[1]Institute of Computational Perception (CP-JKU) & [2]LIT Artificial Intelligence Lab, Johannes Kepler University Linz, Austria

ABSTRACT

Acoustic scene classification and related tasks have been dominated by Convolutional Neural Networks (CNNs) [2–10]. Top-performing CNNs use mainly audio spectograms as input and borrow their architectural design primarily from computer vision. A recent study [1] has shown that restricting the receptive field (RF) of CNNs in appropriate ways is crucial for their performance, robustness and generalization in audio tasks. One side effect of restricting the RF of CNNs is that more frequency information is lost. In this paper, we perform a systematic investigation of different RF configuration for various CNN architectures on the DCASE 2019 Task 1.A dataset. Second, we introduce Frequency Aware CNNs to compensate for the lack of frequency information caused by the restricted RF, and experimentally determine if and in what RF ranges they yield additional improvement. The result of these investigations are several well-performing submissions to different tasks in the DCASE 2019 Challenge.

Index Terms— Acoustic Scene Classification, Frequency-Aware CNNs, Receptive Field Regularization

1. INTRODUCTION

Convolutional Neural Networks (CNNs) have shown great promise as end-to-end classifiers in many tasks such as image classification [11, 12] and acoustic scene classification [5, 13]. Although every year new architectures are proposed that achieve better image recognition performance, we showed in a recent study [1] that these performance gains do not seem to translate to the audio domain. As a solution, we proposed regularizing the *receptive field (RF)* of such CNN architectures in specific ways. The method was applied to several state-of-the-art image recognition architectures, and the resulting models were shown to then achieve state-of-the-art performance in audio classification tasks [1].

Although CNNs can learn their own features and build internal representations from data, the details of how they actually function is crucial to their success in a specific task. In the image recognition domain, a recent study [14] shed light on the decision making procedure of CNNs and showed that using occurrences of small local image features without taking into account their spatial ordering, CNNs can still achieve state-of-the-art results. However, while spatial ordering and local neighboring relations might not be crucial for object recognition in images, this is not the case in audio representations such as spectrograms. A local pattern in lower frequencies does not represent the same acoustic event as the same pattern appearing in higher frequencies. Since CNNs with limited receptive fields[1] are only capable of capturing local features and unlike models such as capsule networks [15], they are unable to find spatial relations between these local patterns. As convolution is equivariant, each filter is applied to the input to generate an output activation, but the output of the network does not know where exactly each filter is. This means that if a specific pattern appears in lower frequencies, and a very similar pattern appears in higher frequencies, later convolutional layers cannot distinguish between the two, and this can result in vulnerabilities in such cases.

[1]as shown in [1], large RFs result in overfitting in audio classification.

In [16], Liu et al. analyzed a generic inability of CNNs to map a pixel in a 2D space, to its exact cartesian coordinate. They address this problem by adding an additional channel to the convolutional layers that contains only the pixel coordinates. Inspired by this solution, we propose a new convolutional layer for audio processing – the *Frequency-aware Convolutional Layer* – to cope with the aforementioned problems in CNNs. We use an additional channel in the convolutional layer that only contains the frequency information which connects each filter to the frequency bin it is applied to.

In this paper, we extend our previous work [1] by modifying the receptive field (RF) of various new architectures such as Resnet [11], PreAct ResNet [17, 18], Shake-shake [18, 19], Densenet [12], and our new frequency-aware *FAResNet* according to the guidelines provided in [1], aiming at pushing the performance of these models on acoustic scene classification tasks. Systematic experiments permit us to determine optimal RF ranges for various architectures on the DCASE 2019 datasets. We show that configuring CNNs to have a receptive field in these ranges has a significant impact on their performance. Based on these insights, we configured network classifiers that achieved a number of top results in several DCASE 2019 challenge tasks [13], as will be briefly reported in Section 4.3.

2. REGULARIZING CNN ARCHITECTURES AND INTRODUCING FREQUENCY AWARENESS

As shown in our previous work [1], the size of the receptive field (RF) is crucial when applying CNNs to audio recognition tasks. Following the guidelines in [1], we adapted various ResNet and DenseNet variants. Using the provided development set for Task 1.A [20], we performed a grid search on the RF of the various ResNet architectures and show the performance of different CNNs under different RF setups. The goal of this investigation, reported in Section 2.1.1, is to introduce a method to systematically push the performance of a single CNN architecture for acoustic scene classification. We base on this method our submissions [13] to the DCASE 2019 challenge [21], especially our top performing single architecture submission for Task 1.A (*cp_resnet*).

Furthermore, in Section 2.2 we introduce *Frequency-aware CNNs* to address the possible shortcomings of models with a smaller

https://doi.org/10.33682/cjd9-kc43

Table 1: Modified ResNet architectures

RB Number	RB Config
	Input 5×5 stride=2
1	$3 \times 3, 1 \times 1$, P
2	$x_1 \times x_1, x_2 \times x_2$, P
3	$x_3 \times x_3, x_4 \times x_4$
4	$x_5 \times x_5, x_6 \times x_6$, P
5	$x_7 \times x_7, x_8 \times x_8$
6	$x_9 \times x_9, x_{10} \times x_{10}$
7	$x_{11} \times x_{11}, x_{12} \times x_{12}$
8	$x_{13} \times x_{13}, x_{14} \times x_{14}$
9	$x_{15} \times x_{15}, x_{16} \times x_{16}$
10	$x_{17} \times x_{17}, x_{18} \times x_{18}$
11	$x_{19} \times x_{19}, x_{20} \times x_{20}$
12	$x_{21} \times x_{21}, x_{22} \times x_{22}$

RB: Residual Block, P: 2×2 max pooling after the block.
$x_k \in \{1, 3\}$: hyper parameter we use to control the RF of the network. Number of channelds per RB:
128 for RBs 1-4; 256 for RBs 5-8; 512 for RBs 9-12.

Table 2: Mapping ρ values to the maximum RF of ResNet variants (networks configured as in Table 1). ρ controls the maximum RF by setting the x_k as explained in Eq. (1).

ρ value	Max RF	ρ value	Max RF
0	23×23	1	31×31
2	39×39	3	55×55
4	71×71	5	87×87
6	103×103	7	135×135
8	167×167	9	199×199
10	231×231	11	263×263
12	295×295	13	327×327
14	359×359	15	391×391
16	423×423	17	455×455
18	487×487	19	519×519
20	551×551	21	583×583

receptive field. Systematic experiments will then show whether, or in what cases, this actually helps improve the results.

2.1. Adapting the Receptive Field of CNNs

2.1.1. ResNet

ResNet [11] and its variants (such as preact-ResNet [17]) achieve state-of-the-art results in image recognition. As we show in our recent study [1], such architectures can be adapted to audio tasks using RF regularization. We adapt the RF of the ResNet in a similar fashion to [1] as explained below. The resulting network architectures are detailed in Table 1. We use the hyper-parameters $x_k \in \{1, 3\}$, corresponding to filter sizes at different CNN levels (see Fig. 1), to control the RF of the network. In order to simplify the process of adjusting the RF of the network, we introduce a new hyper-parameter ρ. We use ρ to control x_k as explained in (1).

$$x_k = \begin{cases} 3 & \text{if } k \le \rho \\ 1 & \text{if } k > \rho \end{cases} \tag{1}$$

For example, setting $\rho = 5$ will result in a ResNet configured as in Table 1 with $x_k = 3$ for $k \in [1, 5]$ and $x_k = 1$ otherwise. The resulting ResNet has a RF of 87×87. Table 2 maps ρ values to the maximum RF of the resulting network[2].

Networks with larger receptive fields degrade in performance as shown in [1]. For this reason, we present the results of ρ values in the range $\rho \in [1, 12]$

2.1.2. PreAct ResNet

PreAct ResNet is a ResNet variant where residual branches are summed up before applying the non-linearity [17]. We specifically use *PreActBN* as explained in [18], since it improves the performance of vanilla PreAct ResNet with and without Shake-Shake regularization for speech emotion recognition.

We control the RF of PreAct ResNets in the same manner as ResNets (Section 2.1.1). Table 1 and Equation 1 explain the configurations of our tested networks.

2.1.3. Shake-Shake ResNet

The Shake-Shake architecture [19] is a variant of ResNet that is proposed for improved stability and robustness. Each residual block has 3 branches; an identity map of the input and 2 convolutional branches, which are summed with random coefficients (in both the forward and backward pass) [19]. This regularization technique has shown empirically to improve the performance of CNNs on many tasks. We also specifically use Shake-Shake regularized *PreActBN* from [18]. In Shake-Shake regularized ResNets, each residual block only has a new branch that is added to the sum. Therefore, the resulting maximum RF of the network is not changed. In result, we use the same techniques to control the RF (Section 2.1.1). Table 1 shows the configuration of both branches of the residual blocks.

Although, Shake-Shake ResNet is not performing well in the classic acoustic scene classification problem (as shown in Section 4), it excels in the case of domain mismatch (Task 1.B [20,21]) and noisy datasets (Task 2 [22]) [13].

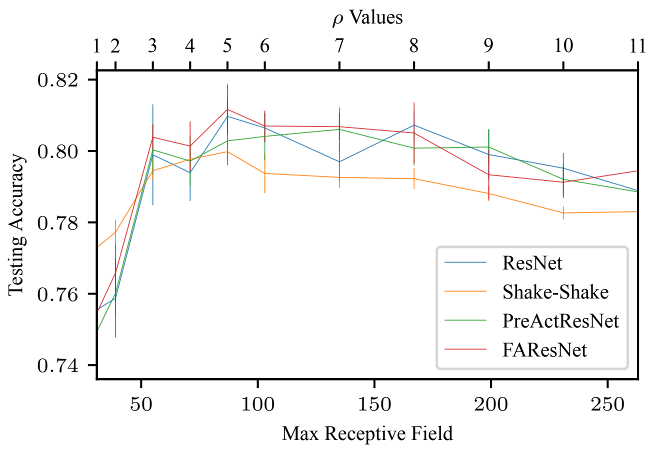
2.1.4. DenseNet

We adapted DenseNet [12] in a similar fashion to DN1 in [1]. We report on two DenseNet configurations with maximum RF of 87×87 and 71×71 pixels (Section 4).

2.2. Frequency-aware Convolution

In CNNs that have a large enough RF, deeper convolutional layers can infer the frequency information of their input feature maps. However, CNNs with large RF degrade in performance and fail to generalize in acoustic scene classification as shown in [1]. On the other hand, in high-performing fully convolutional CNNs, learned CNN filters are agnostic to the frequency range information of the feature maps. In other words, the spectrograms and feature maps

[2]We will release the source code used to produce these networks and replicate the experiments at `https://github.com/kkoutini/cpjku_dcase19`



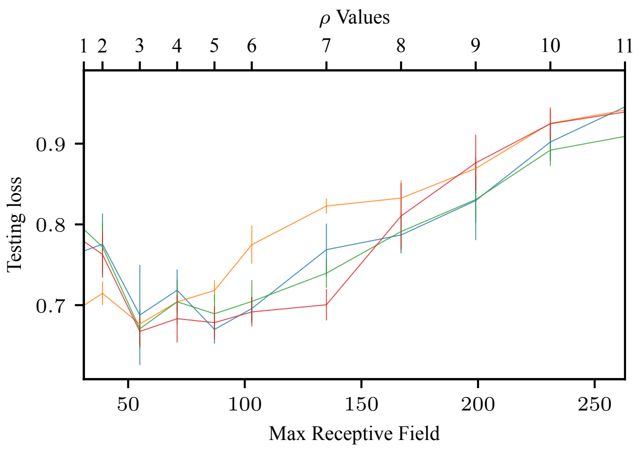


Figure 1: Testing Loss/Accuracy of the provided development split of Task 1 a dataset, for ResNet variants with different receptive fields over the input **without** mix-up.

can be rolled over both the time and frequency dimension with a minor impact on the network predictions. This is one side effect of limiting the receptive field of CNNs on spectograms. We propose a new convolutional layer, which we call *Frequency-aware Convolution*, to make filters aware and more specialized in certain frequencies by concatenating a new channel containing the frequency information of each spatial pixel to each feature map. This technique is similar to CoordConv [16], where the network input is padded with the pixels' coordinates. In our case, we pad all feature maps with a real number indicating the frequency context of each pixel.[3]

The CNN models that incorporate our frequency-aware layer will be called the *Frequency-Aware Convolutional Neural Networks (FACNNs)*. Similarly, we call the frequency-aware ResNet *FAResNet*. We denote the value of the pixel with spatial index (f, t) in the new channel as $V(f, t)$; it is calculated as

$$V(f,t) = f/F \tag{2}$$

where F is the size of the frequency dimension of the feature map, f is the pixel index in the frequency dimension, and t is the pixel index in the time dimension. This new channel gives the convolutional filters a frequency context.

Since making CNNs frequency-aware (by adding the new channel) does not alter the maximum RF of the network, we control the maximum RF of FAResNets similar to ResNet (Section 2.1.1) by changing ρ.

3. EXPERIMENTAL SETUP

3.1. Data Preparation and Training

We extracted the input features using a Short Time Fourier Transform (STFT) with a window size of 2048 and 25% overlap. We perceptually weight the resulting spectrograms and apply a Mel-scaled filter bank in a similar fashion to Dorfer et al. [5]. This preprocessing results in 256 Mel frequency bins. The input is first down-sampled to 22.05 kHz. We process each input channel of the stereo audio input independently and provide the CNN with a two-channel-spectrogram input. The input frames are normalized using the training set mean and standard deviation.

We used Adam [23] with a specific scheduler. We start training with a learning rate of 1×10^{-4}. From epoch 50 until 250, the learning rate decays linearly from 1×10^{-4} to 5×10^{-6}. We train for another 100 epochs with the minimum learning rate 5×10^{-6} in a setup similar to [1].

3.2. Testing Setup

We use the provided development split of DCASE 2019 task 1A [20, 21]. We train our models on the provided training set and treat the provided test set as unseen set. In other words, we don't select best models based on their performance on the test set. Instead, for each model, we report the average results of the last 25 training epochs of two runs on the test set.

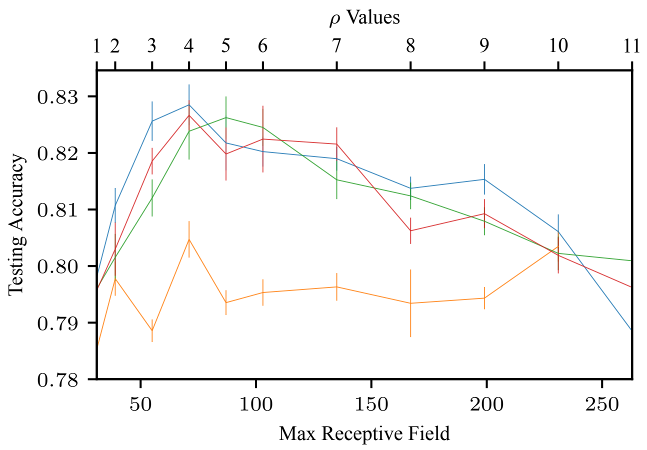
3.3. Data Augmentation

Mix-Up: *Mix-up* [24] is an effective augmentation method that works by linearly combining two input samples and their targets. It was shown to have a great impact on the performance and the generalization of the models.
Spectogram Rolling: We roll the spectograms randomly over the time dimension.

4. RESULTS

Table 3 shows the ρ and Max RF configuration that achieves the top accuracy (mean/std over the last 25 epochs) for each architecture, with and without mix-up. It is also worth noting that the *maximum* RF is different from the *effective* RF as explained in [1, 25]. We control the maximum RF using ρ, while the *effective* RF is dependent on the architecture, the initialization and the data [25]. This is one possible explanation for why different architectures may have a slightly shifted optimal maximum RF range (for example, PreAct in

[3] In this paper, we used a number between -1 and 1, where -1 represents the lowest and 1 the highest frequency in the spectrogram. But this range can be adapted according to the value range of the input.



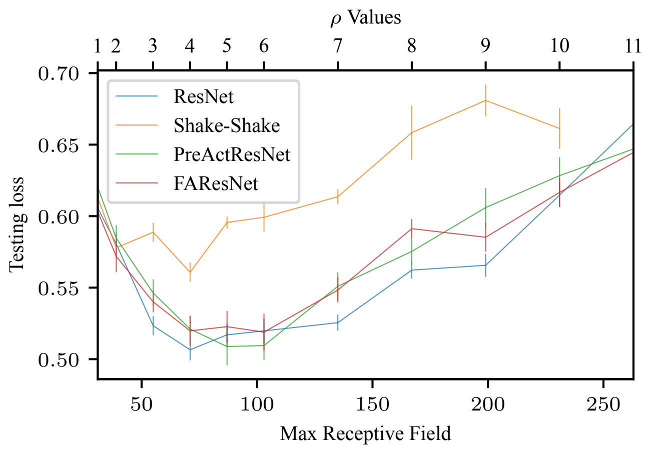


Figure 2: Testing Loss/Accuracy of the provided development split of Task 1 a dataset, for ResNet variants with different receptive fields over the input **with** mix-up.

Table 3 and Figure 2). Likewise, using mix-up can alter the optimal maximum RF range for the networks.

4.1. Without Mix-up

Figure 1 shows the testing loss and accuracy for different architectures over a range of ρ values and – consequently (see Eq. (1)) – maximum RF values. The plots summarize the results for the last 25 epochs of 2 runs. We notice that FAResNet excels mostly in smaller RF networks ($\rho < 8$) where frequency context is more valuable. The figure also shows the best-performing maximum RF range for different architectures to correspond to ρ values in the range $[3, 8]$. In this range, FAResNet outperforms other ResNet variants.

4.2. With Mix-up

Figure 2 shows the testing loss and accuracy when we use mix-up data augmentation. We note that when using mix-up, ResNet outperforms the other variants. Further experiments and investigation are still needed to fully understand the effect of mix-up on these architectures. The figure shows that the best-performing maximum RF range for architectures corresponds to ρ values in the range $[3, 5]$ for *ResNet* and *FAResNet*, and $[4, 6]$ for *PreActResnet*. Shake-Shake achieves its best performance for $\rho = 4$. We see that performance degrades outside these maximum RF ranges for different architectures, in accordance with [1].

4.3. Performance at DCASE 2019

Our receptive field regularized networks achieved the second place [13] (team ranking) in Task 1.A of the DCASE 2019 challenge [20, 21]. We averaged ResNet, PreAct and FAResNet configured with $\rho = 5$ to achieve 83.8% accuracy on the evaluation set. Our ResNet configured with $\rho = 5$ (our single architecture submission *cp_resnet*) achieved 82.8% accuracy when trained on the whole development set; we averaged the prediction of the last training epochs [13]. When instead averaging the predictions of the same architecture trained on a 4-fold cross-validation of the development data, it achieves 83.7% accuracy on the evaluation set. Furthermore, the submission achieved the highest accuracy on the unseen cities in the evaluation set (78.1%).

Table 3: Configurations with top accuracy per network architecture and its corresponding ρ and max RF values with/without mix-up

Network	ρ	**Max RF**	**M/U**	**Accuracy**
ResNet	4	71 × 71	✓	**82.85% ± .36**
PreAct	5	87 × 87	✓	82.62% ± .37
Shake-Shake	4	71 × 71	✓	80.47% ± .32
FAResNet	4	71 × 71	✓	82.66% ± .27
DenseNet		71 × 71	✓	81.53% ± .26
ResNet	5	87 × 87	✗	80.97% ± .46
PreAct	7	135 × 135	✗	80.6% ± .61
Shake-Shake	5	87 × 87	✗	79.98% ± .27
FAResNet	5	87 × 87	✗	**81.17 % ± .7**
DenseNet		87 × 87	✗	79.9% ± .3

M/U: using Mix-Up

The generality and robustness of the proposed RF regularization strategy is demonstrated by the fact that our highly-performing submissions to DCASE 2019 Tasks 1.B and 2 [13] are also based on these architectures.

5. CONCLUSION

In this paper, we have investigated different configurations of deep CNN architectures that correspond to different maximum receptive fields over audio spectograms. We showed that this helps to better design deep CNNs for acoustic classification tasks, and to adapt CNNs performing well in other domains (notably, image recognition) to acoustic scene classification. The good results achieved with this basic strategy in several DCASE 2019 tasks suggest that this is a very general and robust approach that may prove beneficial in various other audio processing tasks.

6. ACKNOWLEDGMENT

This work has been supported by the COMET-K2 Center of the Linz Center of Mechatronics (LCM) funded by the Austrian federal government and the Federal State of Upper Austria.

7. REFERENCES

[1] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The Receptive Field as a Regularizer in Deep Convolutional Neural Networks for Acoustic Scene Classification," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*, A Coruña, Spain, 2019.

[2] H. Eghbal-Zadeh, B. Lehner, M. Dorfer, and G. Widmer, "CP-JKU submissions for DCASE-2016: A hybrid approach using binaural i-vectors and deep convolutional neural networks," in *DCASE 2016-challenge on Detection and Classification of Acoustic Scenes and Events*. DCASE2016 Challenge, 2016.

[3] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 131–135.

[4] B. Lehner, H. Eghbal-Zadeh, M. Dorfer, F. Korzeniowski, K. Koutini, and G. Widmer, "Classifying short acoustic scenes with I-vectors and CNNs: Challenges and optimisations for the 2017 DCASE ASC task," in *DCASE 2017-challenge on Detection and Classification of Acoustic Scenes and Events*. DCASE2017 Challenge, 2017.

[5] M. Dorfer, B. Lehner, H. Eghbal-zadeh, C. Heindl, F. Paischer, and G. Widmer, "Acoustic scene classification with fully convolutional neural networks and I-vectors," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Challenge (DCASE2018)*, 2018.

[6] Y. Sakashita and M. Aono, "Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divisions." DCASE2018 Challenge, 2018.

[7] M. Dorfer and G. Widmer, "Training general-purpose audio tagging networks with noisy labels and iterative self-verification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018, pp. 178–182.

[8] T. Iqbal, Q. Kong, M. Plumbley, and W. Wang, "Stacked convolutional neural networks for general-purpose audio tagging." DCASE2018 Challenge.

[9] D. Lee, S. Lee, Y. Han, and K. Lee, "Ensemble of convolutional neural networks for weakly-supervised sound event detection using multiple scale input." DCASE2017 Challenge.

[10] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Iterative knowledge distillation in R-CNNs for weakly-labeled semi-supervised sound event detection," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018, pp. 173–177.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[12] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[13] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Acoustic scene classification and audio tagging with receptive-field-regularized CNNs," DCASE2019 Challenge, Tech. Rep., June 2019.

[14] W. Brendel and M. Bethge, "Approximating CNNs with bag-of-local-features models works surprisingly well on ImageNet," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=SkfMWhAqYQ

[15] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in neural information processing systems*, 2017, pp. 3856–3866.

[16] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski, "An intriguing failing of convolutional neural networks and the coordconv solution," in *Advances in Neural Information Processing Systems*, 2018, pp. 9605–9616.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," *arXiv preprint arXiv:1603.05027*, 2016.

[18] C. Huang and S. S. Narayanan, "Normalization before shaking toward learning symmetrically distributed representation without margin in speech emotion recognition," *CoRR*, vol. abs/1808.00876, 2018. [Online]. Available: http://arxiv.org/abs/1808.00876

[19] X. Gastaldi, "Shake-shake regularization," *arXiv preprint arXiv:1705.07485*, 2017.

[20] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification in DCASE 2019 challenge: Closed and open set classification and data mismatch setups," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019.

[21] http://dcase.community/challenge2019/task-acoustic-scene-classification.

[22] E. Fonseca, M. Plakal, F. Font, D. P. Ellis, and X. Serra, "Audio tagging with noisy labels and minimal supervision," *arXiv preprint arXiv:1906.02975*, 2019.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[24] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[25] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the Effective Receptive Field in Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 29*, 2016, pp. 4898–4906.

SPECAUGMENT FOR SOUND EVENT DETECTION IN DOMESTIC ENVIRONMENTS USING ENSEMBLE OF CONVOLUTIONAL RECURRENT NEURAL NETWORKS

Wootaek Lim, Sangwon Suh, Sooyoung Park and Youngho Jeong

Media Coding Research Section
Electronics and Telecommunications Research Institute
218 Gajeong-ro, Yuseong-gu, Daejeon, Korea
wtlim@etri.re.kr

ABSTRACT

In this paper, we present a method to detect sound events in domestic environments using small weakly labeled data, large unlabeled data, and strongly labeled synthetic data as proposed in the Detection and Classification of Acoustic Scenes and Events 2019 Challenge task 4. To solve the problem, we use a convolutional recurrent neural network composed of stacks of convolutional neural networks and bi-directional gated recurrent units. Moreover, we propose various methods such as SpecAugment, event activity detection, multi-median filtering, mean-teacher model, and an ensemble of neural networks to improve performance. By combining the proposed methods, sound event detection performance can be enhanced, compared with the baseline algorithm. Consequently, performance evaluation shows that the proposed method provides detection results of 40.89% for event-based metrics and 66.17% for segment-based metrics. For the evaluation dataset, the performance was 34.4% for event-based metrics and 66.4% for segment-based metrics.

Index Terms— DCASE 2019, Sound event detection, CRNN, SpecAugment, Model ensemble

1. INTRODUCTION

Sound event detection (SED) is the field of predicting acoustic events in audio signals. In recent years, this field has witnessed growth owing to the release of large datasets, improvements in algorithms, and improved hardware performance [1, 2]. The Detection and Classification of Acoustic Scenes and Events (DCASE) Challenge has been held for several years with the objective of solving the limitations in SED [3-6]. This year, the DCASE Challenge comprised five tasks, and this study proposed a method to solve the DCASE 2019 Challenge task 4. This is the follow-up to DCASE 2018 task 4. The goal of this task is to train the model to detect sound events using the dataset, which has various types of labels, and to find the onset and offset of sound events. According to last year's submissions, various methods have been proposed to solve this problem [7-13], and the mean-teacher model has shown the best performance [13, 14]. Therefore, the baseline system of task 4 in the DCASE 2019 Challenge is based on the idea of the best submission of DCASE 2018 task 4. The method used in the baseline system is similar to that used in [13], but the proposed network architecture has been simplified.

In this study, a SED system based on a convolutional recurrent neural network (CRNN) is proposed. To improve performance, we perform SpecAugment for data augmentation to overcome the small dataset problem, the event activity detection (EAD) method to learn the weakly labeled dataset, the multi-median filtering (MMF) method using a synthetic dataset for more accurate post-processing, and the mean-teacher model to utilize the unlabeled dataset.

2. DATASET

The dataset for the DCASE 2019 Challenge task 4 comprised 10 s audio clips recorded in an indoor environment or synthesized assuming a similar environment. This task also defines 10 sound event classes [6]. The details of the dataset are described in Table 1. First, three types of datasets are provided for training: the weakly labeled training set; an unlabeled, in-domain training set; and a strongly labeled, synthetic set. The weakly labeled training set and the unlabeled in domain training set are based on AudioSet [15], and the strongly labeled synthetic sets are synthesized based on the dataset proposed in [16] and [17]. A validation set is provided for verification of SED performance. This dataset is a combination of the DCASE 2018 task 4 test and evaluation sets. The evaluation dataset is composed of 13190 audio clips, and the details will be released later.

Table 1: Details of DCASE 2019 Challenge task 4 dataset.

Dataset			Descriptions
Development dataset	Training set	Labeled training set	- 1578 clips (2244 class occurrences) - w/ weak labels
		Unlabeled in domain training set	- 14412 clips - w/o labels
		Synthetic strongly labeled set	- 2045 clips (6032 events) - w/ strong labels
	Validation set		- 1168 clips (4093 events) - w/ strong labels
Evaluation dataset			- 13190 clips - w/ strong labels

https://doi.org/10.33682/qacg-8m97

3. PROPOSED METHOD

3.1. Network structure

The proposed method uses a CRNN as a basic network structure inspired by the DCASE 2019 Challenge task 4 baseline system [18]. This network has a more complex structure than the baseline system. First, the convolutional neural networks (CNNs) layer is composed of the 3×3 kernel on all layers, and the number of feature maps increases from the low- to high-level layers. It also has a gated linear unit (GLU), which was originally proposed in [19], and batch normalization. A dropout layer and average pooling layer are stacked after each CNN module. Two bi-directional gated recurrent units (Bi-GRUs) are stacked after the six CNN layers. At the end of the network, strong and weak predictions are estimated, and the attention module is used to help with learning. The detailed network structure is depicted in Figure 1.

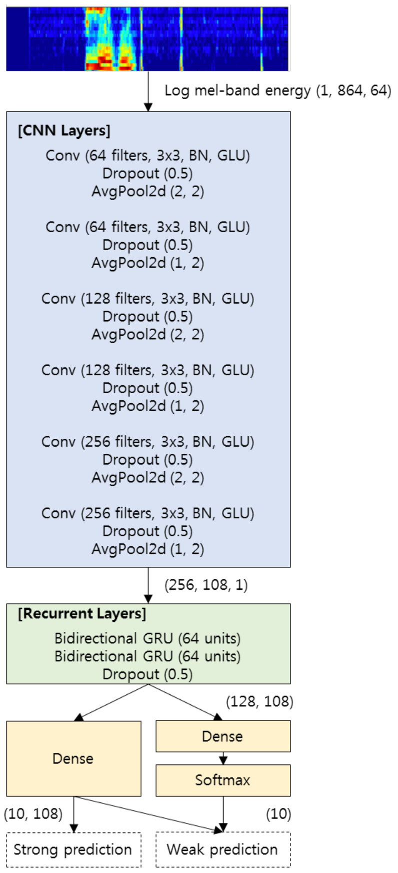


Figure 1: Structure of the CRNN used in our proposed method.

3.2. SpecAugment

When there is insufficient training data, data augmentation can be used to increase the effective size of the existing data, which can greatly improve the neural network performance in many tasks. In audio processing, the conventional data augmentation method transforms waveforms used in learning in the same manner as that for adding time stretching, block mixing, pitch shifting, or background noise. This helps the neural network become more robust by forcing multiple augmented versions of the same audio input into the neural network, which learns the variance during the training process. SpecAugment (SA) is a new data augmentation method for speech recognition that modifies the spectrogram by masking blocks of consecutive frequency channels and time frames [20]. SA applies augmentation directly to the audio spectrogram. Therefore, it is simple and computationally efficient. In this paper, SA was applied directly to the input spectrogram during training. In the frequency domain, the number of masks was 1 with a masking parameter of 10. In addition, in the time domain, the number of masks was 2 with a masking parameter of 50 frames. Time warping was not applied. The dataset used in this process is the weakly labeled dataset, and its robustness is increased by randomly selecting audio clips whether to be augmented or not to be augmented during in each training step.

3.3. Event activity detection

A simple way to realize strong labels from weakly labeled data is to assign a strong label to all time frames. However, assigning a strong label to weakly labeled data is difficult, because there is no information about the existence of the event. Therefore, a pseudo-labeling, created using EAD, was used to learn more accurate labels. A pseudo strong label is assigned when the average frame energy is over a threshold value of 0.7. It assumes that there are no events in the frame if the energy is small [12].

3.4. Multi-median filtering

The output is post-processed by median filtering. Applying median filtering of the same length to various sound event classes is inadequate, because each sound has statistically different characteristics. Therefore, we selected the length of the MMF using the synthetic strongly labeled set. The MMF length for each event class was obtained from the metadata of the synthetic strongly labeled dataset. After calculating the length of all events, the median value of sorted duration was used as the MMF length.

3.5. Mean-teacher model

Semi-supervised learning to utilize the unlabeled in domain training set was done using the mean-teacher model [13, 14]. The mean-teacher model was learned with the same two CRNN structures described in Section 3.1. In the training stage, after the student model is updated, the teacher model is updated using the exponential moving average of the student model weights.

3.6. Model ensemble

A reliable approach to improve the performance of neural networks is to have an ensemble of several trained models. The ensemble technique combines weak learners to create a strong learner. Therefore, the ensemble approach not only improves model diversity but also performance. There are several approaches to forming an ensemble [21]. Our study tested two methods. The first method is an ensemble of different checkpoints in a single model. This method has generally shown limited success, but it is very efficient because it comes from a single training model. The second method is to create an ensemble by learning the same model with different initializations. This method is time-consuming, but simple and powerful. The mean probability of weak learners is used to make the output of the ensemble model.

4. PERFORMANCE EVALUATION

For evaluating the performance of the proposed methods, the dataset described in Section 2 is used. The weakly labeled training set and the synthetic strongly labeled set were used to train the basic CRNN model, and the unlabeled in domain training set was additionally used to train the mean-teacher student model. The audio input was mono channel with a 44.1 kHz sampling rate. To make an input 2D spectrogram, a 10 second audio clip was converted to 64-band log-mel energies with a window size of 2048 and hop length of 511. Consequently, an image with 864 frames and 64 frequency bands was used as a network input. The Adam optimizer was used for network learning, and the learning rate was 0.001. The binary cross-entropy function is used as the criterion for comparing the loss between the target and the output. The early stopping method was not used because the ensemble model could reduce the variance.

Table 2 shows a comparison of performance when using the proposed methods. Training was performed for 500 epochs, and the model was tested every 100 epochs from the 200th epoch onward. The experimental result at the 100th epoch was reported for comparison with the baseline system, but it was not used for the ensemble. The horizontal row denotes the result of the ensemble at different checkpoints. The baseline system showed an F-score of 23.7%, which is improved to 29.52% by using the ensemble of four different checkpoints of a single model. Moreover, when using the network with a deeper structure than the baseline, such as depicted in Figure 1, the performance improved to 32.92%. This system has shown an F-score of 34.70% when applying the MMF as post-processing. Furthermore, the performance was improved to 35.84% by applying EAD, and an F-score of 36.98% was achieved by applying SA as a data augmentation method.

The experimental results of the basic CRNN network which contains all proposed methods are listed in Table 3, and the experimental results based on the mean-teacher model are listed in Table 5. Four experiments (#1-4) were performed for each model for reliable results and model ensemble. As listed in Tables 3 and 5, the mean-teacher model shows slightly better performance on average than the basic CRNN model, although there is a deviation from each training step. Both models outperform the baseline system performance. As previously described, the performance of the ensemble of different checkpoints in a single model and the ensemble of different initializations were evaluated. In Table 3, the horizontal row denotes the result of the ensemble of different checkpoints and the vertical column is the result of the ensemble of different initializations. The ensemble for each row and column was the result of four models combined. The ensemble of different initializations demonstrated better results, and the ensemble of 500th checkpoint models demonstrated an F-score of 38.77%. Finally, the method with an ensemble of 16 models demonstrated the best performance: 39.51% for event-based metrics and 67.29% for segment-based metrics. The detailed results are listed in Table 4. For the evaluation dataset, the performance was 33.2% for event-based metrics and 69.2% for segment-based metrics. The event-based score of this system ranked 16th among the 58 systems. In particular, the segment-based score ranked 3rd among the 58 submitted systems in the DCASE 2019 Challenge task 4.

The results of the mean-teacher model are listed in Table 5. In the mean-teacher model, the ensemble of different checkpoints is unnecessary, but it shows improved performance. Like the basic CRNN model, the ensemble of different initializations shows a better performance in the mean-teacher model. This model demonstrated an F-score of 39.43% when using the ensemble of four models at the 500th checkpoint. Finally, when combining the ensemble composed of 16 models, it showed the best performance: 40.89% for event-based metrics and 66.17% for segment-based metrics. The detailed results are listed in Table 6. The performance on the evaluation dataset was 34.4% for event-based metrics and 66.4% for segment-based metrics. The event-based score of this system ranked 14th among the 58 systems and the segment-based score ranked 8th among the 58 submitted systems in the DCASE 2019 Challenge task 4.

Table 2: Sound event detection performance using proposed methods. (F-score, %)

Model \ Epoch	ep100	ep200	ep300	ep400	ep500	Ensemble
Baseline	*23.70*	-	-	-	-	-
Baseline (Ensemble)	*22.66*	26.55	28.06	27.40	27.32	**29.52**
Proposed CRNN (w/o SA, w/o EAD, w/o MMF)	*26.28*	27.69	29.81	30.86	31.77	**32.92**
Proposed CRNN (w/o SA, w/o EAD, w/ MMF)	*28.77*	30.36	32.26	32.94	33.96	**34.70**
Proposed CRNN (w/o SA, w/ EAD, w/ MMF)	*31.32*	34.16	33.35	34.66	33.99	**35.84**
Proposed CRNN (w/ SA, w/o EAD, w/ MMF)	*31.39*	34.01	34.02	35.88	34.86	**36.98**

Table 3: Sound event detection performance using the basic CRNN model and ensemble. (F-score, %)

Model \ Epoch	ep200	ep300	ep400	ep500	Ensemble
CRNN (# 1)	33.07	34.40	28.94	34.23	36.08
CRNN (# 2)	33.32	36.10	35.46	33.68	37.43
CRNN (# 3)	34.91	33.57	32.72	33.75	36.86
CRNN (# 4)	34.52	34.07	32.75	35.55	36.23
Ensemble	**38.87**	**39.36**	**38.68**	**38.77** (submission-1)	**39.51** (submission-2)

Table 4: Class-wise result of the basic CRNN model ensemble. (submission-2)

Event label	Development dataset (Validation set)				Evaluation dataset
	Event-based metrics		Segment-based metrics		Event-based metrics
	F-score (%)	Error rate	F-score (%)	Error rate	F-score (%)
Alarm/bell/ringing	47.4	0.95	78.6	0.42	26.9
Blender	30.3	1.34	57.4	0.79	36.7
Cat	40.2	1.23	59.6	0.81	53.7
Dishes	19.8	1.30	53.6	0.88	19.3
Dog	21.0	1.29	66.4	0.66	27.1
Electric shaver/toothbrush	42.2	1.31	67.9	0.79	14.0
Frying	39.6	1.36	62.2	0.84	35.9
Running water	40.4	1.05	69.0	0.56	23.0
Speech	51.0	0.86	85.7	0.28	52.4
Vacuum cleaner	63.3	0.78	72.5	0.61	42.9
macro-average	**39.51**	**1.15**	**67.29**	**0.66**	**33.2** (Segment-based 69.2)
micro-average	**40.87**	**1.03**	**72.52**	**0.45**	(not reported)

Table 5: Sound event detection performance using the mean-teacher model and ensemble. (F-score, %)

Model \ Epoch	ep200	ep300	ep400	ep500	Ensemble
Mean-Teacher (# 1)	34.17	34.81	34.86	34.74	36.57
Mean-Teacher (# 2)	33.47	35.59	33.83	34.00	36.29
Mean-Teacher (# 3)	36.83	36.07	36.38	33.51	37.53
Mean-Teacher (# 4)	33.56	36.06	35.57	36.87	38.32
Ensemble	**38.92**	**38.55**	**39.09**	**39.43** (submission-3)	**40.89** (submission-4)

Table 6: Class-wise result of the mean-teacher model ensemble. (submission-4)

Event label	Development dataset (Validation set)				Evaluation dataset
	Event-based metrics		Segment-based metrics		Event-based metrics
	F-score (%)	Error rate	F-score (%)	Error rate	F-score (%)
Alarm/bell/ringing	47.2	0.92	79.4	0.38	26.2
Blender	33.5	1.30	61.0	0.76	35.5
Cat	43.1	1.05	59.4	0.70	57.2
Dishes	22.7	1.17	46.5	0.87	24.1
Dog	27.7	1.21	66.3	0.62	33.1
Electric shaver/toothbrush	42.6	1.32	66.1	0.79	17.4
Frying	40.6	1.26	61.2	0.83	33.3
Running water	32.6	1.11	63.2	0.60	21.5
Speech	57.4	0.80	86.0	0.28	58.5
Vacuum cleaner	61.4	0.76	72.5	0.52	37.1
macro-average	**40.89**	**1.08**	**66.17**	**0.63**	**34.4** (Segment-based 66.4)
micro-average	**44.97**	**0.96**	**72.12**	**0.46**	(not reported)

5. CONCLUSION

The goal of this study was to propose methods for SED in domestic environments using various types of datasets. In this paper, SED performance was improved by using the proposed network structure and various methods such as SA, EAD, MMF, and a mean-teacher student model. Moreover, two ensemble methods and its combination were tested to verify the effectiveness of the ensemble model. According to the experiment, the proposed system achieved an F-score of 40.89% and 66.17% for event-based and segment-based metrics, respectively. For the evaluation dataset, the final performance was 34.4% for event-based metrics and 66.4% for segment-based metrics. In conclusion, the proposed system ranked 6th in event-based metrics and 3rd in segment-based metrics among the 19 teams that submitted in the DCASE 2019 Challenge task 4.

6. ACKNOWLEDGMENT

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT). (No.2017-0-00050, Development of Human Enhancement Technology for auditory and muscle support)

7. REFERENCES

[1] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: outcome of the DCASE 2016 Challenge," IEEE/ACM Transactions on Audio, Speech, and Language Processing, 26(2): 379–393, 2018.

[2] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events," IEEE Transactions on Multimedia, 17(10): 1733–1746, 2015.

[3] "DCASE2016," http://www.cs.tut.fi/sgn/arg/dcase2016/.

[4] "DCASE2017," http://www.cs.tut.fi/sgn/arg/dcase2017/.

[5] "DCASE2018," http://dcase.community/challenge2018/.

[6] "DCASE2019," http://dcase.community/challenge2019/.

[7] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-Scale Weakly Labeled Semi-Supervised Sound Event Detection in Domestic Environments," in Proc. Detection and Classification of Acoustic Scenes and Events Workshop (DCASE), 2018.

[8] R. Serizel, and N. Turpault, "Sound Event Detection from Partially Annotated Data: Trends and Challenges," IcETRAN conference, 2019.

[9] Y. Liu, J. Yan, Y. Song and J. Du, "USTC-NELSLIP System for DCASE 2018 Challenge task 4," Technical Report, DCASE 2018 Challenge.

[10] Q. Kong, T. Iqbal, Y. Xu, W. Wang and M. D. Plumbley, "DCASE 2018 Challenge baseline with convolutional neural networks" Technical Report, DCASE 2018 Challenge.

[11] S. Kothinti, K. Imoto, D. Chakrabarty, G. Sell, S. Watanabe and M. Elhilali, "Joint Acoustic and Class Inference for Weakly Supervised Sound Event Detection," in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019.

[12] W. Lim, S. Suh, and Y. Jeong, "Weakly labeled semi supervised sound event detection using CRNN with inception module," in Proc. Detection and Classification of Acoustic Scenes and Events Workshop (DCASE), 74-77, 2018.

[13] L. JiaKai, "Mean teacher convolution system for dcase 2018 task 4," Technical Report, DCASE 2018 Challenge.

[14] A. Tarvainen, and H. Valpola, "Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results," in advances in neural information processing systems (NIPS), 1195–1204. 2017.

[15] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: an ontology and human-labeled dataset for audio events," in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 776-780, 2017

[16] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra. "Freesound datasets: a platform for the creation of open audio datasets," in Proc. International Society for Music Information Retrieval Conference (ISMIR), 486-493, 2017.

[17] G. Dekkers, S. Lauwereins, B. Thoen, M. W. Adhana, H. Brouckxon, T. V. Waterschoot, B. Vanrumste, M. Verhelst, and P. Karsmakers. "The SINS database for detection of daily activities in a home environment using an acoustic sensor network," in Proc. Detection and Classification of Acoustic Scenes and Events Workshop (DCASE), 32–36, 2017.

[18] N. Turpault, R. Serizel, A. P. Shah, and J. Salamon, "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis," HAL preprint: hal-02160855, 2019.

[19] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modelling with gated convolutional networks," arXiv preprint arXiv: 1612.08083, 2016.

[20] D. S. Park, W. Chan, Y. Zhang, C. C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," arXiv preprint arXiv:1904.08779, 2019.

[21] http://cs231n.github.io/neural-networks-3/#ensemble

GUIDED LEARNING CONVOLUTION SYSTEM FOR DCASE 2019 TASK 4

Liwei Lin[1,2], *Xiangdong Wang*[1], *Hong Liu*[1], *YueLiang Qian*[1],

[1]Bejing Key Laboratory of Mobile Computing and Pervasive Device,
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
{linliwei17g, xdwang, hliu, ylqian}@ict.ac.cn

ABSTRACT

In this paper, we describe in detail the system we submitted to DCASE2019 task 4: sound event detection (SED) in domestic environments. We employ a convolutional neural network (CNN) with an embedding-level attention pooling module to solve it. By considering the interference caused by the co-occurrence of multiple events in the unbalanced dataset, we utilize the disentangled feature to raise the performance of the model. To take advantage of the unlabeled data, we adopt Guided Learning for semi-supervised learning. A group of median filters with adaptive window sizes is utilized in the post-processing of output probabilities of the model. We also analyze the effect of the synthetic data on the performance of the model and finally achieve an event-based F-measure of 45.43% on the validation set and an event-based F-measure of 42.7% on the test set. The system we submitted to the challenge achieves the best performance compared to those of other participates.

***Index Terms*—** Sound event detection, weakly supervised learning, semi-supervised learning, attention, Guided Learning, Disentangled Feature

1. INTRODUCTION

DCASE2019 task 4 is the follow-up to DCASE2018 task 4 [1], which aims at exploring the possibility of the large-scale sound event detection using weakly labeled data (without timestamps) and unlabeled data. Different from DCASE2018 task 4, DCASE2019 task 4 introduces an additional strongly annotated synthetic training set.

Sound event detection (SED) consists in recognizing the presence of sound events in the segment of audio and detecting their onset as well as offset. Due to the high cost of manually labeling data, it is essential to efficiently utilize weakly-labeled data and unlabeled data. Simultaneously, the different physical characteristics of events (such as different duration) and the unbalance of the available training set also increase the difficulty of the multi-class SED in domestic environments. For DCASE2019 task4, there are 5 issues to be resolved:

1) How to learn efficiently with weakly-labeled data?
2) How to learn efficiently with unbalanced training set?
3) How to combine weakly-supervised learning with semi-supervised learning efficiently using weakly-labeled data and unlabeled data?
4) How to design a better post-processing method on the output probabilities of the model to detect more accurate boundaries according to the characteristics of each event category?
5) Does the strongly annotated synthetic training set help?

In this paper, we present a system to solve all these five issues. For issue 1 and 2, we utilize convolutional neural network (CNN) with the embedding-level attention pooling module and disentangled feature [2] to solve them. For issue 3, we adopt a semi-supervised learning method named Guided Learning [3]. For issue 4, according to varied duration of different event categories, we employ a group of median filters with adaptive window sizes in the post-processing of output probabilities of the model. For issue 5, we simply regard the strongly annotated synthetic training set as a weakly annotated training set and conduct a series of ablation experiments to explore its effects on weakly-supervised learning and unsupervised learning separately.

In the rest of this paper, we introduce our methods in Section 2, describe in detail our experiments in Section 3 and draw conclusions in Section 4.

2. METHODS

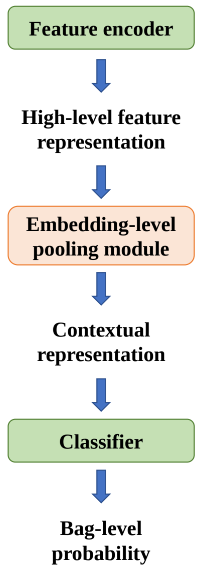
In this section, we discuss the solution for issue 1 in Section 2.1, the solution for issue 2 in Section 2.2, the solution for issue 3 in Section 2.3 and the solution for issue 4 in Section 2.4.

2.1. A CNN model with the embedding-level attention pooling module

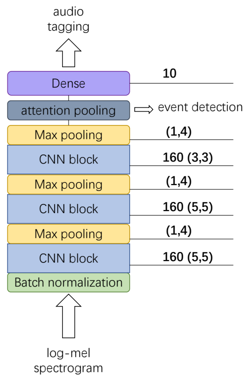
In this section, we describe in detail the model we employ. As shown in Figure 1a, the model comprises 3 parts: a feature encoder, an embedding-level attention pooling module and a classifier. The feature encoder encodes the input feature of an audio clip into high-level feature representations. Assuming that there are C event categories to detect, then the embedding-level attention pooling module integrates these high-level feature representations into C contextual representations. Eventually, the clip-level probabilities can be obtained by passing this C contextual representations through the classifier.

As shown in Figure 1b, the feature encoder we employs is composed of a Batch normalization layer [4], 3 Max pooling layers and 3 CNN blocks, each of which consists of a CNN layer, a Batch normalization layer and a ReLU activation layer as shown in Figure 1c. And the classifier for each contextual representation is a fully-connected layer with a Sigmoid activation layer.

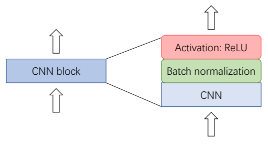
The ability of this model to carry out weakly-supervised learning attributes to its embedding-level attention pooling module. Let $\mathbf{x} = \{x_1, ..., x_T\}$ be the high-level feature representations generated by the feature encoder and $\mathbf{y} = \{y_1, ..., y_C\}$ ($y_c \in \{0, 1\}$) be the groundtruth, where T denotes the number of total frames of the high-level feature representations.


Feature encoder
High-level feature representation
Embedding-level pooling module
Contextual representation
Classifier
Bag-level probability

(a) Model framework



(b) Detailed model architecture



(c) CNN block

Figure 1: CNN model with the embedding-level attention pooling module.

Then for each category c, the embedding-level attention pooling gives different weights $\mathbf{a_c} = \{a_{c1}, ..., a_{cT}\}$ to the corresponding x_t in $\mathbf{x}$. Then the contextual representation $\mathbf{h} = \{h_1, h_2, ..., h_C\}$ can be obtained by the following way:

$$h_c = \sum_t a_{ct} \cdot x_t \tag{1}$$

Such an $\mathbf{a_c}$ enables the model to treat each frame differently. Important frame x_t in $\mathbf{x}$ with larger a_{ct} contributes more to h_c. The embedding-level attention pooling module generates $\mathbf{a_c}$ by the following way:

$$a_{ct} = \frac{\exp\left((w_c^T x_t + b_c)/d\right)}{\sum_k \exp\left((w_c^T x_k + b_c)/d\right)} \tag{2}$$

where d is equal with the dimension of $\mathbf{x}$, w_c^T is a trainable vector, and b_c is the trainable bias.

More importantly, a_{ct} possess the ability to indicate key frames of an audio and is able to generate frame-level probabilities as explained in [2]:

$$\hat{p}\left(y_c \mid x_t\right) = \sigma\left(w_c^T x_t + b_c\right) \tag{3}$$

where σ is Sigmoid function.

Assuming that $\hat{\mathbf{P}}\left(y_c \mid \mathbf{x}\right)$ is the clip-level probabilities for event category c, then the clip-level prediction is:

$$\phi_\mathbf{c}\left(\mathbf{x}\right) = \begin{cases} 1, & \hat{\mathbf{P}}\left(1 \mid \mathbf{x}\right) \geq \alpha \\ 0, & otherwise \end{cases} \tag{4}$$

The frame-level prediction is:

$$\varphi_\mathbf{c}\left(\mathbf{x}, t\right) = \begin{cases} 1, & \hat{p}\left(1 \mid x_t\right) \cdot \phi_\mathbf{c}\left(\mathbf{x}\right) \geq \alpha \\ 0, & otherwise \end{cases} \tag{5}$$

Without loss of generality, we set $\alpha = 0.5$.

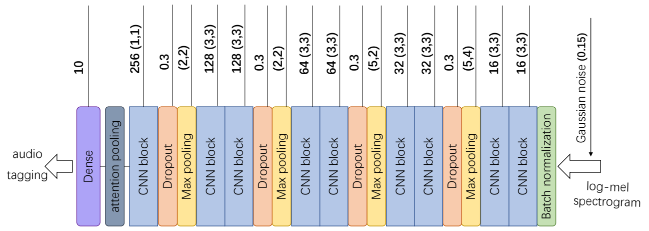


Figure 2: The model architecture of the PT-model.

2.2. Disentangled feature

We take disentangled feature (DF) [2], which re-models the high-level feature subspace of each event category according to the prior information without pre-training, to mitigate the effect of the interference caused by the co-occurrence multiple events.

Assuming that $\chi^\mathbf{d}$ ($\mathbf{x} \subset \chi^\mathbf{d}$) is a d-dimensional space generated by the feature encoder and $\text{ß} = \{e_1, e_2, ..., e_d\}$ is an orthogonal basis of $\chi^\mathbf{d}$ where the element of e_i in i^{th} dimensional is 1. DF selects specific bases of $\chi^\mathbf{d}$ to construct a specific subspace for each category and the basis of the re-modeled feature space $\chi_c^{'}$ of category c is

$$\text{ß}_c^{'} = \{e_1, e_2, ..., e_{k_c}\} \tag{6}$$

$$k_c = \lceil((1 - m) \cdot f_c + m) \cdot d\rceil \tag{7}$$

$$f_c = \sum_i^C \frac{r_i \cdot N_{ci}}{R} \tag{8}$$

$$R = \max_c \sum_{i=1}^C r_i \cdot N_{ci} \tag{9}$$

where m is a constant to avoid too-small k_c and N_{ci} is the number of clips containing i categories including category c in the training set. The constant coefficient r_i denotes the importance these clips:

$$r_i = \begin{cases} 1, & i = 1 \\ 0, & otherwise \end{cases} \tag{10}$$

2.3. Guided Learning

To combine weakly-supervised learning with semi-supervised learning, we utilize Guide Learning (GL) proposed in [3] with a more professional teacher model (PT-model) to guide a more promising student model (PS-model).

The architecture of the PS-model is consistent with the model described in Section 2.1 and we show that of the PT-model in Figure 2. The CNN feature encoder of the PT-model is considered to be better designed than the PS-model on the audio tagging performance with larger sequential sampling size and less trainable parameters. This is because that the larger sequential sampling size allows the CNN feature encoder of the PT-model to have a larger receptive field followed by better exploitation of contextual information.

However, the larger sequential sampling size also disables the PT-model to see finer information due to the compress of sequential information. Therefore, the PS-model is designed with smaller sequential sampling size to see finer information and then achieves better frame-level prediction.

This gap between their ability makes it possible to optimize the PS-model with the guide of the PT-model using unlabeled data. As

Algorithm 1 Guided learning pseudocode.

```
Require: $x_k$ = training input with index $k$
Require: $L$ = set of weakly-labeled training input
Require: $U$ = set of unlabeled training input
Require: $y_k$ = label of weakly-labeled input $x_k \in L$
Require: $S_\theta(x)$ = neural network of the PS-model with trainable parameters $\theta$
Require: $T_{\theta'}(x)$ = neural network of the PT-model model with trainable parameters $\theta'$
Require: $g(x)$ = stochastic input augmentation function
Require: $J(t, z)$ = loss function
Require: $\phi(z)$ = prediction generation function
Ensure: $\theta, \theta'$
  for $i = 1 \to num\_epoches$ do
    if $i > start\_epoch$ then
      $a \leftarrow 1 - \gamma^{i-start\_epoch}$    ▷ calculate the weight of unsupervised loss of the PT-model
    else
      $a \leftarrow 0$
    end if
    for each minibatch $\mathbb{B}$ do
      $s_k \leftarrow S_\theta(x_k \in \mathbb{B})$    ▷ the coarse-level predicted probability of the PS-model
      $t_k \leftarrow T_{\theta'}(g(x_k) \in \mathbb{B})$    ▷ the coarse-level predicted probability of the PT-model
      $\tilde{s_k} \leftarrow \phi(s_k)$    ▷ convert the predicted probability into 0-1 prediction
      $\tilde{t_k} \leftarrow \phi(t_k)$
      if $x_k \in L$ then
        $loss \leftarrow \frac{1}{|\mathbb{B}|}\left\{\sum_{x_k \in J \cap \mathbb{B}}\left[J(y_k, s_k) + J(y_k, t_k)\right]\right\}$
      end if
      if $x_k \in U$ then
        $loss \leftarrow \frac{1}{|\mathbb{B}|}\left\{\sum_{x_k \in U \cap \mathbb{B}}\left[J(\tilde{t_k}, s_k) + a \cdot J(\tilde{s_k}, t_k)\right]\right\}$
      end if
      update $\theta, \theta'$    ▷ update network parameters
    end for
  end for
```

shown in Algorithm 1, an end-to-end process is employed to train these two models.

2.4. Adaptive post-processing

The median filter is utilized for post-processing of the frame-level probabilities output by the model. Instead of determining the window size of the median filter empirically, we adopt a group of median filters with adaptive window sizes for different event categories by the following formulation based on the varying length of different event categories in real life:

$$S_{win} = duration_{ave} \cdot \beta \tag{11}$$

All the frame-level probabilities output by the network are smoothed by a group of median filters with these adaptive window sizes. After smoothed, the probabilities are converted into the 0-1 prediction with a threshold of 0.5 as described in Section 2.1. Then the operation of smoothing is repeated again on the final frame-level prediction.

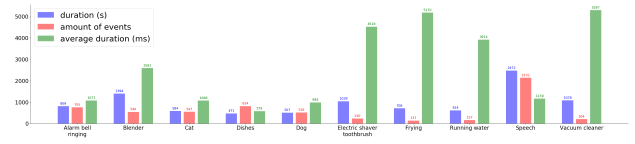


Figure 3: The total duration, number of events and average duration per event category in the synthetic training set.

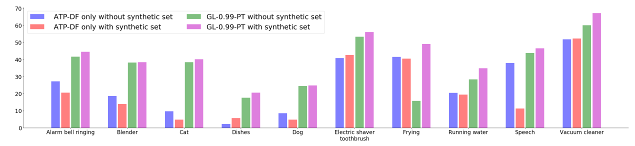


Figure 4: The class-wise F_1 performance

3. EXPERIMENTS

3.1. DCASE 2019 Task 4 Dataset

The dataset [5, 6, 7, 8, 9] of DCASE2019 task 4 is divided into 4 subsets: the weakly annotated training set (1578 clips), the unlabeled training set (14412 clips), the strongly annotated validation set (1168 clips) and the strongly annotated synthetic training set (2045 clips) [10]. We integrate the weakly annotated training set, the unlabeled training set and the strongly annotated synthetic training set (actually we only use weakly labels during training) into a training set and take the validation set as our validation set. The average duration of each event category in the synthetic set is shown in Figure 3.

3.2. Feature exaction and post-processing

We produce 64 log mel-bank magnitudes which are extracted from 40 ms frames with 50% overlap ($n_{FFT} = 2048$) using librosa package [11]. All the 10-second audio clips are extracted to feature vectors with 500 frames. In post-processing, we take $\beta = \frac{1}{3}$ (see details in Section 2.4) in our experiments and the window sizes for different events are shown in Table 1.

3.3. Model architecture

The model architectures of the PS-model and PT-model are described in detail in Section 2. We take $m = 0.04$ for DF. The dimension of DF per category is shown in Table 1. The PS-model has about 2.6 times the number of trainable parameters as the PT-model (877380 / 332364). The start epoch for GL is set to 5. The PS-model with only weakly-supervised learning is named ATP-DF and the co-teaching of the PS-model and the PT-model is named GL-α-PT in the performance report, where α is a hyper-parameter for GL discussed in Algorithm 1.

3.4. Training

The Adam optimizer [12] with learning rate of 0.0018 and mini-batch of 64 10-second patches is utilized to train models. The learning rate is reduced by 20% per 10 epochs. Training early stop if there is no more improvement on clip-level macro F_1 within 20 epochs. All the experiments are repeated 30 times and we report the average results. Event-based measures [13] with a 200ms collar on onsets and a 200ms / 20% of the events length collar on offsets are calculated over the entire test set.

Table 1: The dimension of the disentangled feature when $m = 0.04$ and the window sizes of the median filters when $\beta = \frac{1}{3}$.

Event	**DF dimension**	**Window size (frame)**
Alarm/bell/ringing	137	17
Blender	94	42
Cat	134	17
Dishes	69	9
Dog	132	16
Electric shaver/toothbrush	76	74
Frying	34	85
Running water	160	64
Speech	30	18
Vacuum cleaner	113	87

Table 2: The performance of models from top1 and the ensemble of models.

Model	**Macro F_1 (%)**	
	Event-based	Segment-based
Top1	44.47	66.74
Ensemble (Top1-6)	45.28	**69.06**
Ensemble (Top2-6)	**45.43**	69.02

3.5. Results

As shown in Table 3, GL-0.99-PT (with synthetic set) achieves the best average performance on event-based macro F_1. The class-wise F_1 performance per event category is shown in Figure 3. As shown in Table 2, the ensemble of the models (GL-0.99-PT) from top2 to top6 achieves the best performance, improving the performance by 21.73 percentage points from the baseline. As shown in Table 3, all the models with semi-supervised learning outperform those only with weakly-supervised learning significantly and the model with the best average performance improves the performance by 20.67 percentage points from the weakly-supervised only method. As shown in Figure 5, the performances of all the models without disentangled feature or adaptive window sizes are poorer than those which has.

3.5.1. Does the synthetic training set help?

As shown in Table 3, when learning only with weakly labeled data, the synthetic training set not only does not help improve the results but also brings negative effects. But when combining weakly-supervised learning with semi-supervised learning, the synthetic training set contributes a lot so that the performance is raised by about 5-8 percentage points. We argue that the model tends to be overfitting in the synthetic training set and have difficulty in recognizing the audio clips from the real-life recording since the number of audio clips in the synthetic training set is almost 1.3 times as much as that in the weakly annotated training set. However, the large scale of unlabeled data complements this weakness and enable the synthetic training set to play a positive role during training.

3.5.2. Challenge results

The model (Ensemble Top1-6) achieves an F-measure of 42.7% on the test set and won the first price in the challenge, which is 0.6 percentage point ahead of the second place and 0.7 percentage points ahead of the third place. We note that according to the supplementary metrics released by challenge official, our model achieves the best performance on the Youtube dataset but shows a poorer performance than the second place and the third on the Vimeo dataset. This might be because most of the audios in the dataset are from Youtube. From this point, we guess the data augmentation such as pitch shifting and time stretching might help a lot.

Table 3: The performance of models

Model	**Macro F_1 (%)**	
	Event-based	Segment-based
baseline	23.7	55.2
without the synthetic training set		
ATP-DF	25.95 ± 3.22	56.82 ± 1.34
GL-1-PT	35.19 ± 3.86	61.14 ± 3.14
GL-0.996-PT	$\mathbf{36.50 \pm 3.71}$	$\mathbf{62.03 \pm 3.25}$
GL-0.99-PT	36.21 ± 4.63	61.25 ± 2.77
GL-0.98-PT	33.78 ± 2.95	57.54 ± 3.42
with the synthetic training set		
ATP-DF	21.65 ± 2.55	57.02 ± 1.93
GL-1-PT	41.03 ± 2.98	65.58 ± 2.84
GL-0.996-PT	42.02 ± 3.29	$\mathbf{66.62 \pm 1.82}$
GL-0.99-PT	$\mathbf{42.32 \pm 2.21}$	65.78 ± 2.63
GL-0.98-PT	41.16 ± 2.42	63.89 ± 2.20

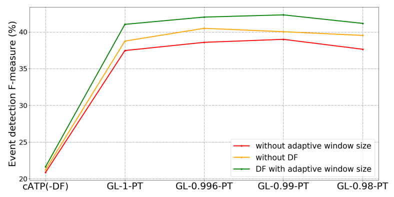


Figure 5: The performance of models without the disentangled feature (without DF) or smoothed by the median filter with a fixed window size of 27 (without adaptive window size).

4. CONCLUSIONS

In this paper, we present a system for DCASE2019 task 4. Actually, we present a complete system for large-scale weakly labeled semi-supervised sound event detection in domestic environments. We broke the task down into 4 small sub-problems and came up with solutions for each. We release the implement to reproduce our system at https://github.com/Kikyo-16/Sound_event_detection. We employ a CNN model with an embedding-level attention module to carry out weakly-supervised learning and utilize GL to carry out semi-supervised learning. DF is employed to raise the performance of the model by reducing the interference caused by the co-occurrence of multiple event categories. In addition, adaptive post-processing is proposed to get more accurate detection boundaries. We also analyze the effect of the synthetic training set. As a result, we achieve state-of-the-art performance on the dataset of DCASE2019 task4.

5. ACKNOWLEDGMENT

This work is partly supported by Beijing Natural Science Foundation (4172058).

6. REFERENCES

[1] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, “Large-scale weakly labeled semi-supervised sound event detection in domestic environments,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 19–23. [Online]. Available: https://hal.inria.fr/hal-01850270

[2] L. Lin, X. Wang, H. Liu, and Y. Qian, “Specialized decision surface and disentangled feature for weakly-supervised polyphonic sound event detection,” *arXiv preprint arXiv:1905.10091*, 2019.

[3] ——, “Guided learning for the combination of weakly-supervised and semi-supervised learning,” *arXiv preprint arXiv:1906.02517*, 2019.

[4] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *arXiv preprint arXiv:1502.03167*, 2015.

[5] N. Turpault, R. Serizel, A. Parag Shah, and J. Salamon, “Sound event detection in domestic environments with weakly labeled data and soundscape synthesis,” June 2019, working paper or preprint. [Online]. Available: https://hal.inria.fr/hal-02160855

[6] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, “Audio set: An ontology and human-labeled dataset for audio events,” in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[7] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, “Freesound datasets: a platform for the creation of open audio datasets,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, Suzhou, China, 2017, pp. 486–493.

[8] F. Font, G. Roma, and X. Serra, “Freesound technical demo,” in *Proceedings of the 21st ACM international conference on Multimedia*. ACM, 2013, pp. 411–412.

[9] G. Dekkers, S. Lauwereins, B. Thoen, M. W. Adhana, H. Brouckxon, T. van Waterschoot, B. Vanrumste, M. Verhelst, and P. Karsmakers, “The SINS database for detection of daily activities in a home environment using an acoustic sensor network,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, November 2017, pp. 32–36.

[10] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, “Scaper: A library for soundscape synthesis and augmentation,” in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 344–348.

[11] Brian McFee, Colin Raffel, Dawen Liang, Daniel P.W. Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto, “librosa: Audio and Music Signal Analysis in Python,” in *Proceedings of the 14th Python in Science Conference*, Kathryn Huff and James Bergstra, Eds., 2015, pp. 18 – 24.

[12] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[13] A. Mesaros, T. Heittola, and T. Virtanen, “Metrics for polyphonic sound event detection,” *Applied Sciences*, vol. 6, no. 6, p. 162, 2016. [Online]. Available: http://www.mdpi.com/2076-3417/6/6/162

CROWDSOURCING A DATASET OF AUDIO CAPTIONS

Samuel Lipping, Konstantinos Drossos, and Tuomas Virtanen

Audio Research Group, Tampere University, Tampere, Finland
{firstname.lastname}@tuni.fi

ABSTRACT

Audio captioning is a novel field of multi-modal translation and it is the task of creating a textual description of the content of an audio signal (e.g. "people talking in a big room"). The creation of a dataset for this task requires a considerable amount of work, rendering the crowdsourcing a very attractive option. In this paper we present a three-step framework for crowdsourcing an audio captioning dataset, based on concepts and practises followed for the creation of widely used image captioning and machine translations datasets. During the first step initial captions are gathered. A grammatically corrected and/or rephrased version of each initial caption is obtained in the second step. Finally, the initial and edited captions are rated, keeping the top ones for the produced dataset. We objectively evaluate the impact of our framework during the process of creating an audio captioning dataset, in terms of diversity and number of typographical errors in the obtained captions. The obtained results show that the resulting dataset has fewer typographical errors than the initial captions, and on average each sound in the produced dataset has captions with a Jaccard similarity of 0.24, roughly equivalent to two ten-word captions having in common four words with the same root, indicating that the captions are dissimilar while they still contain some of the same information.

Index Terms— audio captioning, captioning, amt, crowdsourcing, Amazon Mechanical Turk

1. INTRODUCTION

Multimodal datasets usually have a set of data in one modality and paired set of data in another modality, creating an association of two different forms of media. These datasets differ from a typical classification or regression dataset in the sense that the two modalities convey the same content, but in different form. One example is the captioning task, where the datasets include a form of media (e.g. image) and then a textual description of the perceived content of the media. Two examples of captioning are image [1, 2, 3] and audio captioning [4], where a textual description, i.e. a caption, is generated given an image or an audio file, respectively. Captioning tasks largely use deep learning methods [5, 6] where models are trained and evaluated on datasets, rendering the dataset as an important factor for the quality of the developed methods. Therefore, a good captioning dataset should have captions that are able to represent the differences on the perceived content (i.e. diverse captions). Also, it should have multiple captions per sample in order to represent the different ways of writing the same information (i.e. rephrasing) and allowing for a better assessment of the performance of the captioning method [7].

Different datasets exist for image captioning [2, 3, 8], consisting of images and multiple captions per image. Most (if not all) of image captioning datasets are created by employing crowdsourcing and annotators that are located in an English speaking area (e.g. U.S.A., Australia, U.K., etc). Crowdsourcing provides several benefits, such as having no restrictions on location and the possibility of simultaneous annotation, and using a ready crowdsourcing platform provides the additional benefit of having an established base of users, i.e. potential annotators [9]. One example of an image captioning dataset is the Flickr 8K dataset, which consists of 8092 images with five captions each, and the captions were obtained by crowdsourcing [8]. The dataset images were hand-selected and they depict actions and events to encourage full sentence captions. The annotators were pre-screened (by answering questions regarding grammar and image captioning), and were required to be located in the US and to have an approval rate of 95% on previous tasks on the crowdsourcing platform [10]. Another example of a crowsourced image captioning dataset is the Microsoft COCO Caption dataset, which consists of five captions for over 200 000 images [7]. Annotators captioned the images one at a time and were told to write captions that contain at least eight words [2]. The restriction for the eight words encourages describing the image thoroughly and benefits the diversity of captions.

Audio captioning is a novel recent area of research, introduced in [4]. The first dataset used for audio captioning in [4] is proprietary and consists of textual descriptions of general audio [11]. Recently two other audio captioning datasets have been created, which are not proprietary; the Audio Caption and the AudioCaps datasets. The Audio Caption [12] dataset was partially released with 3710 video clips and their audio tracks, annotated with Mandarin Chinese and English captions. Each video had a duration of about 10 seconds and was annotated with three distinct captions. The video clips were annotated by Chinese university students. Annotators were instructed to focus on the sound of the video clips. The Chinese captions were then translated to English with Baidu Translation [12]. The AudioCaps [13] dataset was created by crowdsourcing. The audio material consists of 46 000 audio files from 527 audio event categories from the AudioSet [14] dataset, and each audio file in AudioCaps has been annotated with one caption. The annotators were pre-screened by discerning those participants who consistently violated the given instructions, such as transcribing speech in the audio or describing the visual stimulus instead of the audio. Additionally, the annotators were required to be located in an English-speaking area, to have an approval rate of 95% on previous tasks, and to have at least 1000 approved submissions on the crowdsourcing platform. Audio clips were selected such that the distribution of their classes was balanced, while ignoring classes that require visuals to be identified (for example the category "inside small room").

There are some considerations regarding the above mentioned datasets. Sound ambiguity is a known and well exploited property (for example in foley sounds) and by providing visual stimuli or word indications there is a high chance of reducing ambiguity and

https://doi.org/10.33682/sezz-vd31

hampering the diversity of the captions. In both datasets, the diversity is hampered by the lack of a minimum number of words in a caption, while in the case of AudioCaps the diversity is further hampered by removing informative sounds about spatial relationships (e.g. "inside small room"), and giving the labels from AudioSet to the annotators to guide the caption. Additionally, Audio Caption is created by annotators who are not located in an English speaking area, increasing the chance for flaws in the acquired English captions. Finally, in the AudioCaps the assessment of the audio captioning methods is hindered by having only one caption per sound.

In this paper we set out to provide a framework for annotating large sets of audio files with multiple captions, which is structured in three steps. We have used this framework in preparing a new and freely available audio captioning dataset, which will be released during Autumn 2019. We employ the Amazon Mechanical Turk (AMT) as our crowdsourcing platform which has been used in numerous previous studies [2, 10, 13, 15, 16]. The rest of the paper is organized as follows. In Section 2 we describe the proposed framework and in Section 3 we describe how we evaluate the effectiveness of the structure of our framework. The results of the evaluation are presented in Section 4. The paper is concluded in Section 5.

2. PROPOSED FRAMEWORK

Our proposed framework consists of three serially executed steps, inspired by practices followed in the creation of image captioning and machine translation datasets [8, 7, 17], and which is implemented on an online crowdsourcing platform. We employ as potential annotators the registered users of the platform (the total number of registered users is not fixed and depends on the platform) who are located in an English speaking area and have at least 3000 and 95% approved (i.e. total and approval rate) submissions on the platform. From the potential annotators, annotators are selected for the steps on a first-come-first-served basis. Again, the number of potential annotators is not known and depends on the platform used. The assignment of audio files and captions to annotators will be explained in each step. The framework employs a set of N_a audio files and produces a set of N_c captions per file. In the first step we gather N_c initial audio captions per file and in the second step the initial N_c captions are edited, resulting to a second set of N_c edited captions per audio file. In the third step we select the best N_c captions per file between the initial and the edited ones. After each step, we screen the answers of the annotators. We permanently exclude annotators that consistently do not follow the given instructions, and we re-assign work that is deemed unacceptable in the screening process to other annotators. The workflow for our proposed framework is visualized in Figure 1.

In more detail, in the first step we solicit annotators for producing captions for all N_a audio files. An annotator is presented with an audio file that is randomly selected from the audio files that have not yet been annotated and asked to write one caption, i.e. a sentence describing the perceived contents of the audio file without assuming any information not present in the audio. No other information is provided to the annotators to aid in describing the audio stimulus (i.e. no access to the name of the audio file, to any tags, or to any visual information is given). This way the caption contains only the perceived information from the audio stimulus and is not based on any prior knowledge about the audio file. To encourage providing descriptive captions with adequate information, we set a minimum

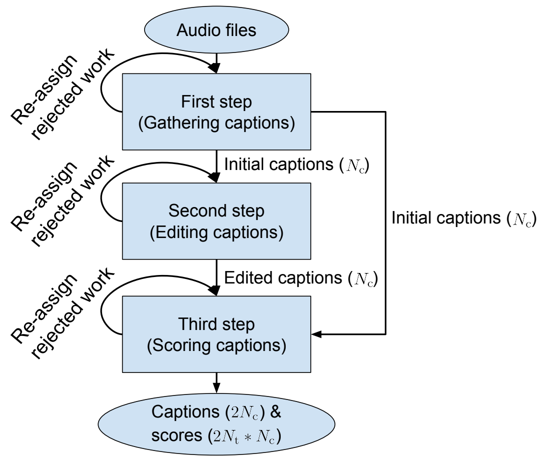

Audio files
Re-assign rejected work
First step (Gathering captions)
Initial captions (N_c)
Re-assign rejected work
Second step (Editing captions)
Initial captions (N_c)
Edited captions (N_c)
Re-assign rejected work
Third step (Scoring captions)
Captions ($2N_\mathrm{c}$) & scores ($2N_\mathrm{t} * N_\mathrm{c}$)

Figure 1: Graph of the task flow of our framework.

caption length of eight words. Additional instructions for annotators in the first task include not to use non-descriptive phrases, such as "There is", "I hear", "sound of", and "sounds like" in the caption to reach the limit of eight words. We present each of our audio files to N_c distinct annotators. From the first step we acquire a set of N_c initial captions for each of the audio files employed.

Some of the initial captions might include grammatical and/or typographical errors (e.g. "*An* car is driving..."), awkward sentence structures (e.g. "An bird swallow-squelches to itself in an small branch"), or similar problems that are easier for humans to detect than for an algorithm. For that reason we introduce a second step for crowdsourcing the correction of any errors in the initial captions, where each initial caption is edited once. In this step an annotator is presented with a random caption that has not yet been edited from the initial captions (i.e. the annotator does not have access to any other information but only the caption). The annotator is instructed to read the given caption and to write an edited caption that fixes the above mentioned problems (e.g. grammatical errors, awkward sentence structures, etc) in the initial one. If there are no errors in the initial caption, the annotator is instructed to only rephrase the initial caption. With this way we acquire a significant number of linguistic corrections on the obtained captions and, in the same time, gather a new set of N_c edited captions per audio file that offer variations in the structure of sentences and association of words. By crowdsourcing the task of the correction and rephrasing, we gain access to significantly large number of workers (i.e. the users of the platform, compared to a non-crowdsourced solution) and diversity (because of the all different workers editing others' captions). It must be noted that if an annotator in the second step has provided a caption in the first step, then this annotator is not presented with his own caption(s) for editing.

After the first two steps each audio file has N_c initial and N_c edited captions, i.e. a total of $2N_\mathrm{c}$ captions. These captions include initial captions with or without grammatical errors and edited captions that fail or succeed to remove errors. To determine which captions describe the audio most accurately and are grammatically most correct, we introduce the third step. In this step an annotator is presented with an audio file that is randomly selected from the audio files that have not yet been annotated with scores and each of the $2N_\mathrm{c}$ captions of that audio file, and is asked to score each caption separately. The annotator scores each caption based on how accurately the caption describes the audio file (i.e. gives an accuracy score to the caption) and how grammatically correct it is (i.e. gives a fluency score to the caption). The annotator gives both scores on a scale of 1 to 4, with 1 meaning "Bad" and 4 meaning "Very good". All the $2N_\text{c}$ captions of each audio file are scored by N_t annotators. As in the previous step, annotators are not presented with their own captions to score. The total accuracy and fluency scores are then calculated by summing the N_t individual scores. Finally, the $2N_\text{c}$ captions for each audio file are sorted in a descending fashion, accounting firstly for the total accuracy and then for the total fluency score. The top N_c captions are selected for each audio file.

3. EVALUATION

We evaluate our framework in the process of creating a new audio captioning dataset that will be released in Autumn 2019, by objectively assessing the impact of the three steps in terms of grammatical correctness and diversity of the gathered captions. We assess the grammatical correctness through the number of typographical errors (the fewer errors, the better) and the diversity by examining the similarity of the captions (the less similar the captions, the more diversity). We use $N_\text{a} = 5000$ audio files with time duration ranging from 15 to 30 seconds, and gathered randomly using the Freesound[1] platform. Audio files in the Freesound platform are tagged, with tags indicating possible categories for the contents of each file. All N_a files do not have tags indicating speech, music, and/or sound effects (e.g. "bass", "glitch", "sci-fi"). All gathered audio files were post-processed to have a maximum absolute value of 1. For each audio file we gather $N_\text{c} = 5$ captions and at the third step, we employ $N_\text{t} = 3$ annotators to rate the $2N_\text{c}$ captions, leading to a range of scores from 3 ($N_\text{t} * 1$) to 12 ($N_\text{t} * 4$). All annotations are gathered using the Amazon Mechanical Turk platform and its registered users as potential annotators. The first, second, and third steps are annotated by 693, 1033, and 1215 annotators with an average of 36, 24, and 12 annotations per annotator respectively. We present the obtained results in Section 4.

We count the typographical errors appearing in the captions for each audio file separately for each of the N_c initial, edited, final, and non-selected (in the third step) captions. To determine typographical errors we use the US and UK libraries of the CyHunspell python library[2], which uses the Hunspell spellchecker[3]. Having edited captions with fewer errors than the initial captions measures the impact of the second step. Additionally, having a set of final selected captions with fewer typographical errors than the ones which are not selected, indicates that in the third step the framework provides a set of final captions that are better (grammatically) than the rest.

To assess the diversity, we use the Jaccard similarity, also known as intersection over union. The Jaccard similarity of two sentences a and b is defined as

$$J(a, b) = \frac{|\mathbb{W}_a \cap \mathbb{W}_b|}{|\mathbb{W}_a \cup \mathbb{W}_b|}, \tag{1}$$

where $\mathbb{W}_a$ is the set of stemmed words (i.e. words reduced to their roots, e.g. "cats" to "cat") in sentence a, $\mathbb{W}_b$ is the set of stemmed words in sentence b, and $0 \leq J(a, b) \leq 1$. When $J(a, b) = 0$, then the sentences a and b have no common (stemmed) words and the sets $\mathbb{W}_a$ and $\mathbb{W}_b$ are disjoint. On the contrary, $J(a, b) = 1$ shows that $\mathbb{W}_a$ and $\mathbb{W}_b$ contain exactly the same stemmed words. For word stemming (i.e. for finding the roots of words) we use the snowball stemmer from the NLTK language toolkit [18]. To measure the amount of rephrasing, we calculate $J(a, b)$ between the initial and edited captions, using as a each of the initial captions and as b the corresponding edited caption. A high $J(a, b)$ will reveal almost no rephrasing and a low one will reveal significant rephrasing. To measure the diversity of the final N_c captions, we firstly calculate the mean $J(a, b)$ for each audio file, using as a and b all the pairs of the final N_c captions. Then, we calculate the mean J across all audio files. We name the mean of the mean Jaccard similarity across all audio files as cross-similarity.

Finally, we evaluate the impact of the proposed framework on the set of words of all captions, that is, the final dictionary or word corpus that will be formed by all captions. We denote the set of words appearing in the N_c captions of an audio file by $\mathbb{S}_a$. We merge all $\mathbb{S}_a$ to the multiset (i.e. bag) $\mathbb{S}_T$. We count the number of appearances of each of the words in $\mathbb{S}_T$, focusing on the rare words, i.e. words that appear up to five times in $\mathbb{S}_T$. For example, if a word in $\mathbb{S}_T$ has a number of appearances equal to two, it means that this word appears in the captions of exactly two audio files. This measure is of importance for the final dataset, because an audio file should not be in both the training and another split (i.e. validation or testing). This means that rare words that appear once in $\mathbb{S}_T$ result in unknown words/tokens to one of the splits. Words that appear twice in $\mathbb{S}_T$ result in audio files that can be used in two, different splits.

[1]https://freesound.org/
[2]https://pypi.org/project/CyHunspell/
[3]http://hunspell.github.io/

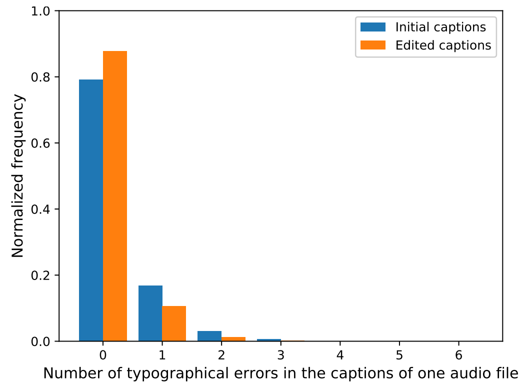


Figure 2: The number of typographical errors in the captions by the normalized frequency of audio files.

4. RESULTS & DISCUSSION

Figure 2 illustrates the frequency of audio files with typographical errors in their captions, for both initial and edited captions. It can be seen that the edited captions are less likely to contain any typographical errors than the initial captions. This means that the second step has a positive impact on the grammatical correctness, managing to produce captions with fewer typographical errors. In total, the edited captions have about 45% fewer typographical errors than the initial captions.

Figure 3 illustrates the histogram of the Jaccard similarities between the initial and the corresponding edited captions. The average similarity is 0.62, which corresponds approximately to, e.g., changing two words in an 8-word caption. Therefore, the second step results in a reasonable amount of added diversity, roughly calculated to changing a fourth of the words in a sentence. Because the number of typographical errors in the edited captions is significantly lower

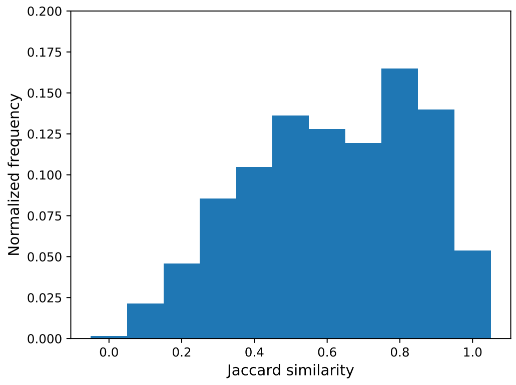


Figure 3: Jaccard similarity between initial and edited captions.

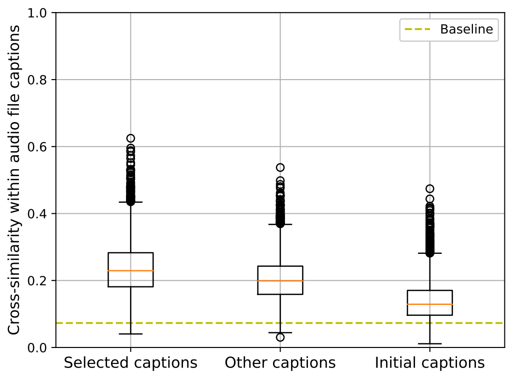


Figure 4: Cross-similarity of selected, non-selected, and initial captions

than that of the initial captions, high frequencies of similarities in the high range might be a result of annotators fixing these errors and therefore not rephrasing the caption.

Figure 4 displays a box plot of the cross-similarity values for the N_c selected (i.e. the final), the other (i.e. the rest N_c not selected at the third step), and the N_c initial captions. The cross-similarity values for the selected, other, and initial captions are 0.24, 0.20, and 0.14 respectively. From the results it can be inferred that from the first step of our framework we indeed get a diverse set of initial captions. Moreover, the results in Figure 4 show that the third step actually managed to control the increased diversity that the initial captions have, producing a lower (but still high) diversity for the captions. The baseline in the figure is calculated by creating random pairs of sentences, from all captions of all audio files.

Figure 5 depicts the percentage of captions versus the number of typographical errors, considering also the total fluency score that is gathered in the third task for each, corresponding caption. It can be seen that the fluency score is inversely related to the number of typographical errors. For example, the captions with a fluency score of 3 have, on average, 18 times more typographical errors than the captions with a fluency score of 12. These results clearly indicate that the fluency score successfully differentiated the levels of fluency within the captions.

Finally, in Figure 6 are the percentages of audio files that have rare words, with numbers of appearances ranging from 1 to 4. The plots show that the selected captions are more likely not to contain any rare words with any number of appearances from one to four. This fact indicates that the resulting diversity imposed by the proposed framework does not hamper the quality of the resulting word corpus and the resulting dataset.

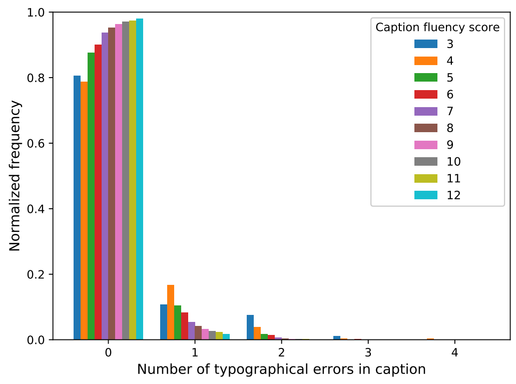


Figure 5: The number of typographical errors appearing in a caption by the total fluency score given in the third step.

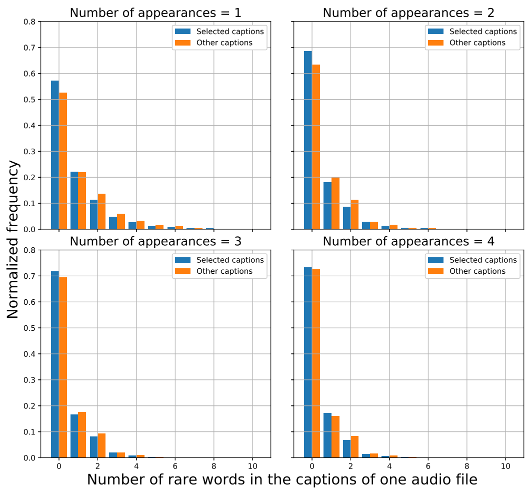


Figure 6: The number of rare words with numbers of appearances of 1 to 4 by the normalized frequency of audio files.

5. CONCLUSIONS & FUTURE WORK

In this paper we presented a framework for the creation of an audio captioning dataset, using a crowdsourcing platform. Our framework is based on three steps of gathering, editing, and scoring the captions. We objectively evaluated the framework during the process of creating a new dataset for audio captioning, and in terms of grammatical correctness and diversity. The results show that the first step of our framework gathers a diverse set of initial captions, the second step gathers a set of edited captions that reduces the number of typographical errors in the initial captions while introducing additional diversity, and the third step extracts from the initial and edited captions a set of final selected captions that maintain a high diversity without introducing many rare words.

Further development of the framework could include prescreening annotators as a way to eliminate manual screening of the annotations and automated processes for the control of more grammatical attributes and the number of rare words.

6. REFERENCES

[1] A. Karpathy and L. Fei-Fei, “Deep visual-semantic alignments for generating image descriptions,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 664–676, Apr. 2017. [Online]. Available: https://doi.org/10.1109/TPAMI.2016.2598339

[2] X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick, “Microsoft coco captions: Data collection and evaluation server,” *arXiv preprint arXiv:1504.00325*, 2015.

[3] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, “From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions,” *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 67–78, 2014. [Online]. Available: https://transacl.org/ojs/index.php/tacl/article/view/229

[4] K. Drossos, S. Adavanne, and T. Virtanen, “Automated audio captioning with recurrent neural networks,” in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, Oct 2017, pp. 374–378.

[5] X. He and L. Deng, “Deep learning for image-to-text generation: A technical overview,” *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 109–116, Nov 2017.

[6] M. Amaresh and S. Chitrakala, “Video captioning using deep learning: An overview of methods, datasets and metrics,” in *2019 International Conference on Communication and Signal Processing (ICCSP)*, April 2019, pp. 0656–0661.

[7] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: common objects in context,” *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[8] M. Hodosh, P. Young, and J. Hockenmaier, “Framing image description as a ranking task: Data, models and evaluation metrics,” in *J. Artif. Intell. Res.*, 2013.

[9] K. Drossos, A. Floros, and A. Giannakoulopoulos, “Beads: A dataset of binaural emotionally annotated digital sounds,” in *IISA 2014, The 5th International Conference on Information, Intelligence, Systems and Applications*, July 2014, pp. 158–163.

[10] C. Rashtchian, P. Young, M. Hodosh, and J. Hockenmaier, “Collecting image annotations using amazon’s mechanical turk,” in *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon’s Mechanical Turk*, ser. CSLDAMT ’10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 139–147. [Online]. Available: http://dl.acm.org/citation.cfm?id=1866696.1866717

[11] ProSoundEffects, “Master Library 2.0,” http://www.prosoundeffects.com/blog/master-library-2-0-nab/, accessed March 2017, 2015. [Online]. Available: http://www.prosoundeffects.com/blog/master-library-2-0-nab/

[12] M. Wu, H. Dinkel, and K. Yu, “Audio caption: Listen and tell,” in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 830–834.

[13] C. D. Kim, B. Kim, H. Lee, and G. Kim, “Audiocaps: Generating captions for audios in the wild,” in *NAACL-HLT*, 2019.

[14] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, “Audio set: An ontology and human-labeled dataset for audio events,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 776–780.

[15] M. Marge, S. Banerjee, and A. I. Rudnicky, “Using the amazon mechanical turk for transcription of spoken language,” in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, March 2010, pp. 5270–5273.

[16] A. Sorokin and D. Forsyth, “Utility data annotation with amazon mechanical turk,” in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, June 2008, pp. 1–8.

[17] O. F. Zaidan and C. Callison-Burch, “Crowdsourcing translation: Professional quality from non-professionals,” in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, ser. HLT ’11. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011, pp. 1220–1229. [Online]. Available: http://dl.acm.org/citation.cfm?id=2002472.2002626

[18] E. Loper and S. Bird, “NLTK: The natural language toolkit,” in *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, 2002.

LONG-DISTANCE DETECTION OF BIOACOUSTIC EVENTS WITH PER-CHANNEL ENERGY NORMALIZATION

Vincent Lostanlen[1,2], *Kaitlin Palmer*[3], *Elly Knight*[4], *Christopher Clark*[1], *Holger Klinck*[1], *Andrew Farnsworth*[1], *Tina Wong*[2], *Jason Cramer*[2], *and Juan Pablo Bello*[2]

[1] Cornell Lab of Ornithology, Ithaca, NY, USA
[2] New York University, Center for Urban Science and Progress, New York, NY, USA
[3] San Diego State University, San Diego, CA, USA
[4] University of Alberta, Edmonton, AB, USA

ABSTRACT

This paper proposes to perform unsupervised detection of bioacoustic events by pooling the magnitudes of spectrogram frames after per-channel energy normalization (PCEN). Although PCEN was originally developed for speech recognition, it also has beneficial effects in enhancing animal vocalizations, despite the presence of atmospheric absorption and intermittent noise. We prove that PCEN generalizes logarithm-based spectral flux, yet with a tunable time scale for background noise estimation. In comparison with pointwise logarithm, PCEN reduces false alarm rate by 50x in the near field and 5x in the far field, both on avian and marine bioacoustic datasets. Such improvements come at moderate computational cost and require no human intervention, thus heralding a promising future for PCEN in bioacoustics.

Index Terms— Acoustic noise, acoustic sensors, acoustic signal detection, spectrogram, underwater acoustics.

1. INTRODUCTION

The deployment of autonomous recording units offers a minimally invasive sampling of acoustic habitats [1], with numerous applications in ecology and conservation biology [2]. In this context, there is an extensive literature on tailoring spectrogram parameters to a specific task of detection or classification: the effects of window size, frequency scale, and discretization are now well understood [3, 4]. However, the important topic of loudness mapping, i.e. representing contrast in the time–frequency domain, has received less attention.

This article investigates the impact of distance between sensor and source on the time–frequency representation of acoustic events. In particular, we point out that measuring local contrast by a difference in pointwise logarithms, as is routinely done in machine learning for bioacoustics, suffers from numerical instabilities in the presence of atmospheric attenuation and intermittent noise. To address this problem, we propose to employ an adaptive gain control technique known as per-channel energy normalization (PCEN) [5].

We deliberately err on the side of design simplicity: rather than training a sophisticated classifier, we apply a constant threshold on the time series of max-pooled PCEN magnitudes. In doing so, our goal is not to achieve the lowest possible false alarm rate, but to argue in favor of replacing the logarithmic mapping of loudness by PCEN in all systems for long-distance sound event detections, including more powerful yet opaque ones such as deep neural networks [6, 7].

Section 2 discusses the theoretical benefits of such a replacement: it proves that PCEN extends temporal context beyond a single temporal frame, thus improving effective detection radius. Sections 3 and 4 present applications to avian and underwater bioacoustics respectively, thereby revealing complementary issues: while bird call detection focuses on mitigating atmospheric absorption at high audible frequencies (1–10 kHz), whale call detection focuses on mitigating the interference of amplitude-modulated noise from near-field passing ships at low audible frequencies (50–500 Hz).

2. SPECTROTEMPORAL MEASURES OF NOVELTY

2.1. Averaged spectral flux

Let $\mathbf{E}(\mathbf{x})[t,f]$ be the magnitude spectrogram of some discrete-time waveform $\mathbf{x}[t]$. In full generality, the ordinal variable f may either represent frequency on a linear scale, a mel scale, or a logarithmic scale. Given $\mathbf{E}$, an implementation of spectral flux composes three operators: loudness mapping, contrast estimation, and feature aggregation. In its most widespread variant, named *averaged spectral flux*, these three operators respectively correspond to pointwise logarithm, rectified differentiation, and frequential averaging:

$$\mathrm{SF}_{\mathrm{avg}}(\mathbf{x})[t] = \sum_{f} \frac{\max\left(\log \mathbf{E}(\mathbf{x})[t,f] - \log \mathbf{E}(\mathbf{x})[t-1,f], 0\right)}{N_{\mathrm{fr}}} \tag{1}$$

where N_{fr} is the number of frequency bands f in $\mathbf{E}$. The motivation underlying this design choice finds its roots in psychoacoustics, and notably the Weber-Fechner law, which states that the relationship between stimulus and sensation is logarithmic [8]. We may also remark that Equation 1 is invariant to gain. Indeed, multiplying the waveform $\mathbf{x}[t]$ by some constant $K \neq 0$ incurs a multiplication by K in each frequency band of $\mathbf{E}(\mathbf{x})$, and thus an additive bias of $\log K$ in $\log \mathbf{E}(x)$, which eventually cancels after first-order differentiation. In the case of a single point source at some distance d, the relative change in acoustic pressure K caused by a spherical wave propagation is proportional to $\frac{1}{d}$. Therefore, in a lossless medium without reflections, logarithm-based spectral flux is invariant to geometric spreading insofar as acoustic sources do not overlap in the time–frequency domain.

This work is partially supported by NSF awards 1633206 and 1633259, the Leon Levy Foundation, and the Pinkerton Foundation.

The source code to reproduce experiments and figures is available at: https://www.github.com/BirdVox/lostanlen2019dcase

https://doi.org/10.33682/ts6e-sn53

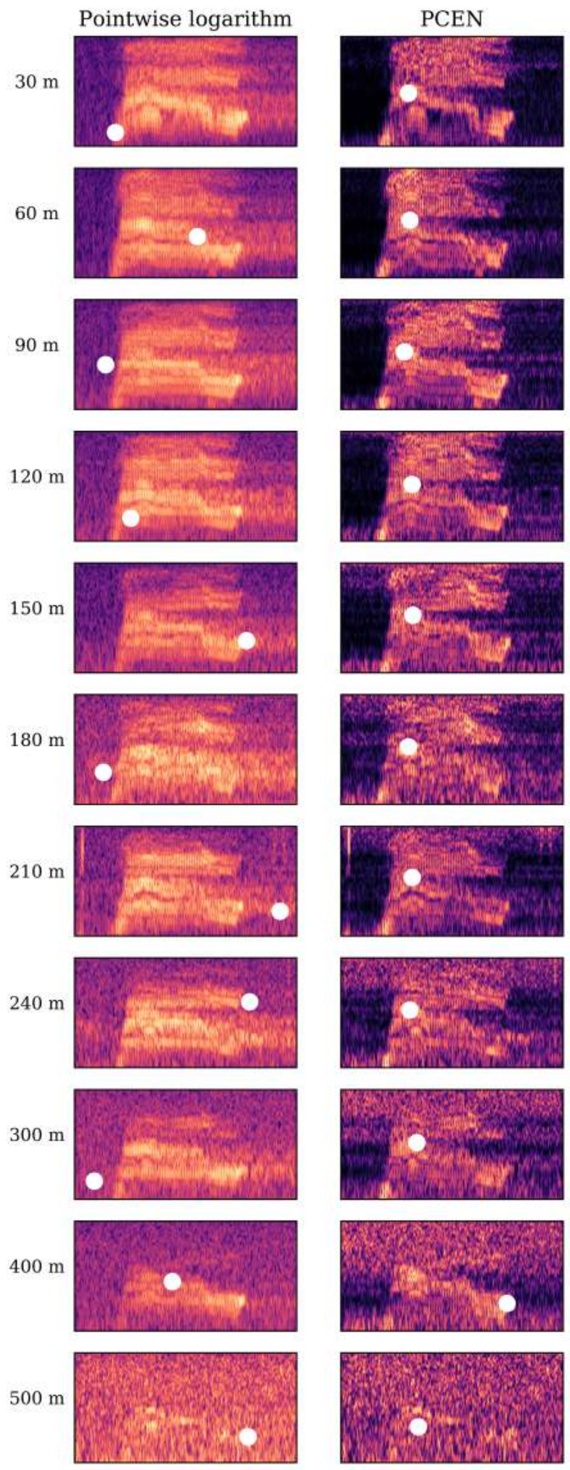


Figure 1: Effect of pointwise logarithm (left) and per-channel energy normalization (PCEN, right) on the same Common Nighthawk vocalization, as recorded from various distances. White dots depict the time–frequency locations of maximal spectral flux (left) or maximal PCEN magnitude (right). The spectrogram covers a duration of 700 ms and a frequency range between 2 and 10 kHz.

2.2. Max-pooled spectral flux

The situation is different in an absorbing medium. Indeed, heat conduction and shear viscosity, in conjunction with molecular relaxation processes, attenuate sine waves in quadratic proportion to their frequency [9]. Under standard atmospheric conditions, this attenuation is below $5\,\mathrm{dB\,km}^{-1}$ at 1 kHz, yet of the order of $100\,\mathrm{dB\,km}^{-1}$ at 10 kHz. As a result, bird calls spanning multiple octaves lose in bandwidth as they travel through air. A simple workaround is to replace the frequential averaging in Equation 1 by a max-pooling operator. This replacement yields the *max-pooled spectral flux*

$$\mathrm{SF}_{\max}(\mathbf{x})[t] = \max_f \big(\log \mathbf{E}(\mathbf{x})[t,f] - \log \mathbf{E}(\mathbf{x})[t-1,f]\big), \quad (2)$$

which performs differentiation on a single frequency band, and is thus invariant to the low-pass filtering effect induced by absorption. However, as illustrated in Figure 1, the definition above suffers from numerical instabilities. Indeed, $\mathrm{SF}_{\max}(\mathbf{x})$ discards all but two scalar values, corresponding to neighboring time–frequency bins in the spectrogram $\mathbf{E}(\mathbf{x})$.

2.3. Max-pooled per-channel energy normalization

In order to associate invariance and stability, this article proposes to increase the time scale of contrast estimation beyond a single spectrogram frame. To this end, we replace both the logarithmic mapping of loudness and the first-order differentiation by a procedure of per-channel energy normalization (PCEN). PCEN was recently introduced as a trainable acoustic frontend for far-field automatic speech recognition [5]. In full generality, PCEN results from an equation of the form

$$\mathrm{PCEN}(\mathbf{x})[t,f] = \frac{1}{r}\left(\frac{\mathbf{E}(\mathbf{x})[t,f]}{\big(\varepsilon + \mathbf{M}(\mathbf{x})[t,f]\big)^{\alpha}} + \delta\right)^{r} - \frac{\delta^r}{r}, \quad (3)$$

where the gain control matrix $\mathbf{M}(\mathbf{x})$ proceeds from $\mathbf{E}(\mathbf{x})$ by first-order IIR filtering:

$$\mathbf{M}(\mathbf{x})[t,f] = s \times \mathbf{E}(\mathbf{x})[t-1,f] + (1-s) \times \mathbf{M}(\mathbf{x})[t-1,f]. \quad (4)$$

Note that the definition in Equation 3 differs from the original definition [5] by a factor of $\frac{1}{r}$. This is in order to allow the limit case $r \to 0$ to remain nonzero. Investigating the role of all parameters in PCEN is beyond the scope of this paper; we refer to the asymptotic analysis of [10] in this regard. Rather, we focus on the smoothing parameter $0 < s < 1$ as striking a tradeoff between numerical stability ($s \to 0$) and rapid adaptation to nonstationary in background noise ($s \to 1$). The following proposition, proven in Section 6, asserts that PCEN is essentially a generalization of spectral flux.

Proposition 2.1. *At the limit* $(s,\varepsilon,\alpha,r) \to (1,0,1,0)$ *in Equations 3 and 4, and for any finite value of* δ, $\mathrm{PCEN}(\mathbf{x})(t,f)$ *tends towards*

$$\log\big(\mathbf{E}(\mathbf{x})[t,f] + \mathbf{E}(\mathbf{x})[t-1,f]\big) - \log \mathbf{E}(\mathbf{x})[t-1,f], \quad (5)$$

which is a smooth approximation of the summand in Equation 1.

For the sake of simplicity, we adopt the PCEN parametrization that is prescribed by Proposition 2.1: we set $\varepsilon = 0$, $\alpha = 1$, $\delta = 1$, and $r = 0$. Derecursifying the autoregressive dependency in 4 and summarizing across frequencies yields the *max-pooled PCEN* detection function

$$\mathrm{PCEN}_{\max}(\mathbf{x})[t] = \log\left(1 + \max_f \frac{\mathbf{E}(\mathbf{x})[t,f]}{s\sum_{\tau=0}^{+\infty}(1-s)^{\tau}\mathbf{E}(\mathbf{x})[t-\tau-1,f]}\right). \quad (6)$$

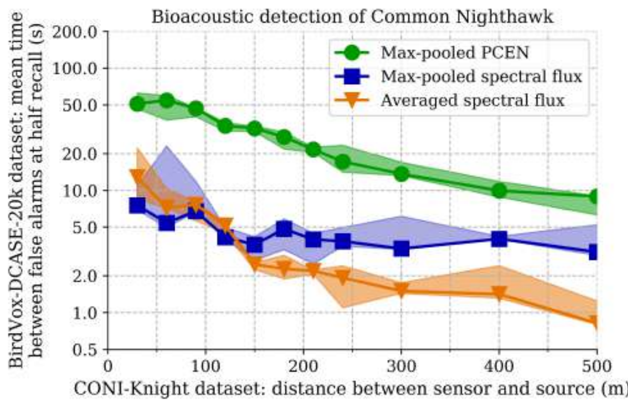


Figure 2: Detection of Common Nighthawk calls: evolution of mean time between false alarms at half recall (MTBFA@50) as a function of distance between sensor of source. Shaded areas denote interquartile variations across individual birds. See Section 3 for details.

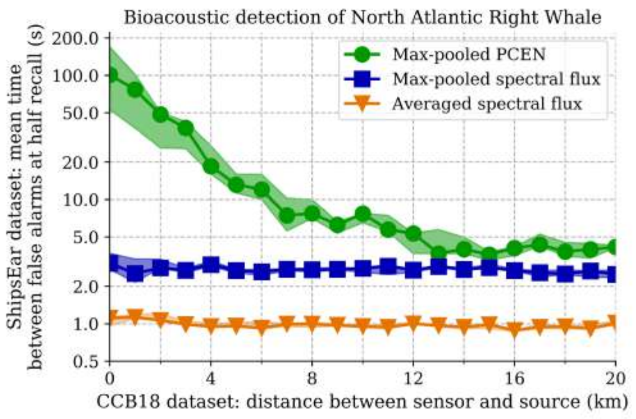


Figure 3: Detection of North Atlantic Right Whale calls: evolution of mean time between false alarms at half recall (MTBFA@50) as a function of distance between sensor of source. Shaded areas denote interquartile variations across days. See Section 4 for details.

3. APPLICATION TO AVIAN BIOACOUSTICS

3.1. CONI-Knight dataset of Common Nighthawk calls

We consider the problem of detecting isolated calls from breeding birds in a moderately cluttered habitat. To this end, we use the CONI-Knight dataset [11], which contains 64 vocalizations from five different adult male Common Nighthawks (*Chordeiles minor*), as recorded by 11 autonomous recording units in a regenerating pine forest north of Fort McMurray, AB, CA. The acoustic sensor network forms a linear transect, in which the distance between each microphone and the vocalizing individual varies from 30 m to 500 m. The dataset contains $11 \times 64 = 704$ positive audio clips in total, each lasting 700 ms. These clips were annotated by an expert, as part of a larger collection of continuous recordings which lasts seven hours in total. We represent each of these clips by their mel-frequency magnitude spectrograms, consisting of 128 bands between 2 kHz and 11.025 kHz, and computed with a Hann window of duration 12 ms (256 samples) and hop 1.5 ms (32 samples). These parameters are identical as in the state-of-the-art deep learning model for bird species recognition from flight calls [12]. We use the librosa implementation of PCEN [13] with $s = 0.09$, i.e. an averaging time scale of about $T = 100$ ms.

Figure 1 displays the mel-frequency spectrogram of one call at various distances, after processing them with either pointwise logarithm (left) or PCEN (right). Atmospheric absorption is particularly noticeable above 200 m, especially in the highest frequencies. Furthermore, we observe that max-pooled spectral flux is numerically unstable, because it triggers at different time–frequency bins from one sensor to the next. In comparison, PCEN is more consistent in reaching maximal magnitude at the onset of the call, and at the same frequency band.

3.2. Evaluation: mean time between false alarms at half recall

Our evaluation procedure consists in two stages: distance-specific threshold calibration and estimation of false alarm rate. In the first stage, we split the dataset of positive clips (i.e. containing one vocalization) into disjoint subsets of increasing average distance; sort the values of the detection function over this subset in decreasing order; and set the detection threshold at the median value, thus yielding a detection recall of 50%. In the second stage, we run the detector on an external dataset of negative recordings, i.e. containing no vocalizations from the species of interest; apply the detection thresholds that were prescribed by the first stage; and count the number of false alarms, i.e. values of the detection function that are above threshold. Dividing the total duration of the dataset of negative recordings by this number of peaks above threshold yields the *mean time between false alarms at half recall* (MTBFA@50) of the detector, which grows in inverse proportion to false alarm rate. We repeat this operation over all available subsets to obtain a curve that decreases with distance, and which reflects the ability of the detection curve to generalize from near-field to far-field events.

3.3. Results and discussion

In the case of the Common Nighthawk, we choose the BirdVox-DCASE-20k dataset [14] as a source of negative recordings. A derivative of BirdVox-full-night [15], this dataset has been divided into 20k ten-second soundscapes from six autonomous recording units in Ithaca, NY, USA, and annotated by an expert for presence of bird calls. Among these 20k soundscapes, 9983 are guaranteed to contain no bird call, and *a fortiori* no Common Nighthawk call. These 9983 recordings amount to 27 hours of audio, i.e. over 30M spectrogram frames. For each detection function, we subtract the minimum value over each 10-second scene to the frame-wise value, in order to account for the nonstationarity in background noise at the scale of multiple hours.

Figure 2 summarizes our results. We find that max-pooled PCEN enjoys a five-fold reduction in false alarm rate with respect to average spectral flux. In addition, the false alarm rate at 300 m of max-pooled PCEN is comparable with the false alarm rate of averaged spectral flux at 30 m. As a post hoc qualitative analysis, we compute novelty curves for 200 recordings of outdoor noise from the ESC-50 dataset [16]: geophony (rain, wind), biophony (crickets), and anthropophony (helicopter, chainsaw). For max-pooled spectral flux, we find that the main causes of false alarms are pouring water (30% of total amount), crackling fire (17%), and water drops (10%).

4. APPLICATION TO MARINE BIOACOUSTICS

4.1. CCB18 dataset of North Atlantic Right Whale calls

We consider the problem of detecting isolated calls from whales in a noisy environment. To this end, we use the CCB18 dataset, which contains vocalizations from about 80 North Atlantic Right Whales (*Eubalaena glacialis*), as recorded by nine underwater sensors during five days in Cape Cod Bay, MA, USA. The distance between sensor and source is estimated by acoustic beamforming, similarly as in [17]. The dataset contains 40k clips in total, each lasting two seconds. These clips were annotated by an expert, as part of a larger collection of continuous recordings which lasts 1k hours in total. We represent each of these clips by their short-term Fourier transform (STFT) magnitude spectrograms, consisting of 128 bands between 8 Hz and 1 kHz, and computed with a Hann window of duration 128 ms and hop of 64 ms. We set $s = 0.33$, i.e. an averaging time scale of about $T = 1\,\mathrm{s}$. We choose the ShipsEar dataset as a source of negative recordings [18]. This dataset contains 90 ship underwater noise recordings from vessels of various sizes, most of them acquired at a distance of 50 m or less. These 90 recordings amount to 189 minutes of audio, i.e. 177k spectrogram frames.

4.2. Results and discussion

Figure 3 summarizes our results. First, we find that averaged spectral flux leads to poor false alarm rates, even in the near field. We postulate that this is because, in the CCB18 dataset, ship passage events occasionally introduce high received levels of noise. In other words, distance sets an upper bound, but no lower bound, on signal-to-noise ratio. Therefore, achieving 50% recall with averaged spectral flux requires to employ a low detection threshold, which in turn triggers numerous false alarms.

Secondly, we find that, across the board, replacing averaged spectral flux by max-pooled spectral flux allows a two-fold reduction in false alarm rate. We postulate that this improvement is due to the fact that whale calls are locally sinusoidal whereas near-field ship noise is broadband. Indeed, the max-pooled spectral flux of a chirp is above its averaged spectral flux, with a ratio of the order of N_{fr}; whereas the averaged and max-pooled spectral fluxes of a Dirac impulse are the same. Therefore, maximum pooling is particularly well suited to the extraction of chirps in noise [19].

Thirdly, we find that, in the near field, replacing spectral flux by PCEN leads to a 50-fold reduction in false alarm rate. We postulate that this is because ship noise has rapid amplitude modulations, at typical periods of 50 to 500 ms (i.e. engine speeds of 120 to 1200 rotations per minute). If this period approaches twice the hop duration (i.e. 128 ms in our case), short-term magnitudes $\log \mathbf{E}(x)[t-1,f]$ and $\mathbf{E}(x)[t,f]$ may correspond precisely to intake and expansion in the two-stroke cycle of the ship, thus eliciting large values of spectral flux. Nevertheless, in the case of PCEN, the periodic activation of one every other frame causes $\mathbf{M}(\mathbf{x})[t,f]$ to be of the order of $\frac{1}{2}\mathbf{E}(\mathbf{x})[t,f]$, assuming that the parameter T is large enough to encompass multiple periods. Therefore, $\mathrm{PCEN}_{\max}(x)[t]$ peaks at $\log(\frac{3}{2})$ in the absence of any transient signal. This peak value is relatively low in comparison with the max-pooled PCEN of a near- or mid-field whale call.

Fourthly, we find that the false alarm rate of max-pooled PCEN increases exponentially with distance, until reaching comparable values as max-pooled spectral flux at a distance of 12 km. This decay is due, in part, to geometric spreading, but also to more complex acoustic phenomena, such as reflections and scattering with the surface as well as the ocean floor [20]. At these large distances, a successful detector should not only denoise, but also dereverberate whale calls. Max-pooled PCEN does not have any mechanism for dereverberation, and thus falls short of that objective. Thus, an ad hoc detection function is no longer sufficient, and the resort to advanced machine learning techniques appears as necessary. We must note, however, that deep convolutional networks in the time–frequency domain rely on the same functional blocks as max-pooled PCEN — i.e. rectified extraction of local contrast and max-pooling — albeit in a more sophisticated, data-driven fashion. Consequently, we believe that PCEN, whether parametrized by feature learning or by domain-specific knowledge, has a promising future in deep learning for environmental bioacoustics.

5. CONCLUSION

An adequate representation of loudness in the time–frequency domain is paramount to efficient sound event detection. This is particularly true in bioacoustic monitoring applications, where the source of interest may vocalize at a large distance to the microphone. Our experiments on the Common Nighthawk and the North Atlantic Right Whale demonstrate that, given a simple maximum pooling procedure across frequencies, per-channel energy normalization (PCEN) outperforms conventional (logarithm-based) spectral flux. Beyond the direct comparison between ad hoc detection functions at various distances, this study illustrates the appeal in replacing pointwise logarithm by PCEN in time–frequency representations of mid- and far-field audio signals. In the future, PCEN could be used, for example, as a similarity measure for spectrotemporal template matching; as an input to deep convolutional networks in the time–frequency domain [21]; or as a frequency-dependent acoustic complexity index for visualizing nonstationary effects in “false color spectrograms” [22] of open soundscapes.

6. APPENDIX: PROOF OF PROPOSITION 2.1

Proof. Applying Taylor's theorem to the exponential function yields

$$\frac{\delta^r}{r}\left[\Big(1+\frac{\mathbf{E}(\mathbf{x})[t,f])}{\mathbf{M}(\mathbf{x})[t,f]}\Big)^r - 1\right] \approx \delta^r \log\Big(1+\frac{\mathbf{E}(\mathbf{x})[t,f]}{\mathbf{M}(\mathbf{x})[t,f]}\Big) \qquad (7)$$

with an error term proportional to $r\delta^r \log(1+\frac{\mathbf{E}(\mathbf{x})[t,f]}{\mathbf{M}(\mathbf{x})[t,f]})^2$, which vanishes at the limit $r \to 0$ as long as $\mathbf{M}(\mathbf{x})[t,f]$ remains nonzero. On the left-hand side, we recognize $\mathbf{PCEN}(\mathbf{x})[t,f]$ with $\varepsilon = 0$ and $\alpha = 1$. On the right-hand side, the finite factor δ^r tends towards 1 for $r \to 0$. The limit $s \to 1$ allows to replace $\mathbf{M}(\mathbf{x})[t,f]$ by $\mathbf{E}(\mathbf{x})[t-1,f]$. We conclude with

$$\begin{aligned}\log\Big(1+\frac{\mathbf{E}(\mathbf{x})[t,f]}{\mathbf{E}(\mathbf{x})[t-1,f]}\Big) &= \log\Big(\frac{\mathbf{E}(\mathbf{x})[t-1,f]+\mathbf{E}(\mathbf{x})[t,f]}{\mathbf{E}(\mathbf{x})[t-1,f]}\Big)\\ &= \log\big(\mathbf{E}(\mathbf{x})[t,f]+\mathbf{E}(\mathbf{x})[t-1,f]\big) - \log \mathbf{E}(\mathbf{x})[t-1,f]. \qquad (8)\end{aligned}$$

Interestingly, the distinction between Equation 1 and Equation 5 mirrors the distinction between the rectified linear unit (ReLU) $y \mapsto \max(y,0)$ and the softplus $y \mapsto \log(1+\exp(y))$ in deep learning. ■

7. ACKNOWLEDGMENT

We wish to thank D. Santos-Domínguez for sharing his dataset.

8. REFERENCES

[1] J. Shonfield and E. Bayne, “Autonomous recording units in avian ecological research: current use and future applications,” *Avian Conservation and Ecology*, vol. 12, no. 1, pp. 42–54, 2017.

[2] M. G. Efford, D. K. Dawson, and D. L. Borchers, “Population density estimated from locations of individuals on a passive detector array,” *Ecology*, vol. 90, no. 10, pp. 2676–2682, 2009.

[3] J. S. Ulloa, A. Gasc, P. Gaucher, T. Aubin, M. Réjou-Méchain, and J. Sueur, “Screening large audio datasets to determine the time and space distribution of screaming piha birds in a tropical forest,” *Ecological informatics*, vol. 31, pp. 91–99, 2016.

[4] E. C. Knight, S. P. Hernandez, E. M. Bayne, V. Bulitko, and B. V. Tucker, “Pre-processing spectrogram parameters improve the accuracy of bioacoustic classification using convolutional neural networks,” *Bioacoustics*, pp. 1–19, 2019. [Online]. Available: https://doi.org/10.1080/09524622.2019.1606734

[5] Y. Wang, P. Getreuer, T. Hughes, R. F. Lyon, and R. A. Saurous, “Trainable frontend for robust and far-field keyword spotting,” in *Proc. IEEE ICASSP*, 2017, pp. 5670–5674.

[6] P. Zinemanas, P. Cancela, and M. Rocamora, “End-to-end convolutional neural networks for sound event detection in urban environments,” in *Proceedings of the Conference of Open Innovations Association (FRUCT)*, April 2019, pp. 533–539.

[7] M. Cartwright, J. Cramer, J. Salamon, and J. P. Bello, “TriCycle: Audio representation learning from sensor network data using self-supervision,” in *Proc. IEEE WASPAA*. IEEE, Oct 2019.

[8] A. Klapuri, “Sound onset detection by applying psychoacoustic knowledge,” in *Proc. IEEE ICASSP*, vol. 6. IEEE, 1999, pp. 3089–3092.

[9] L. C. Sutherland and G. A. Daigle, “Atmospheric sound propagation,” *Handbook of Acoustics*, vol. 28, pp. 305–329, 1998.

[10] V. Lostanlen, J. Salamon, M. Cartwright, B. McFee, A. Farnsworth, S. Kelling, and J. P. Bello, “Per-channel energy normalization: Why and how,” *Signal Processing Letters*, vol. 26, no. 1, pp. 39–43, Jan 2019.

[11] E. C. Knight and E. M. Bayne, “Classification threshold and training data affect the quality and utility of focal species data processed with automated audio-recognition software,” *Bioacoustics*, pp. 1–16, 2018. [Online]. Available: https://doi.org/10.1080/09524622.2018.1503971

[12] J. Salamon, J. P. Bello, A. Farnsworth, and S. Kelling, “Fusing shallow and deep learning for bioacoustic bird species classification,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 141–145.

[13] B. McFee, M. McVicar, S. Balke, V. Lostanlen, C. Thomé, C. Raffel, D. Lee, K. Lee, O. Nieto, E. Battenberg, D. Ellis, R. Yamamoto, J. Moore, Z. Wei, R. Bittner, K. Choi, P. Friesch, F.-R. Stöter, S. K. Golu, S. Waloschek, S. Kranzler, R. Naktinis, D. Repetto, C. F. Hawthorne, C. Carr, W. Pimenta, P. Viktorin, P. Brossier, J. F. Santos, J. Wu, E. Peterson, and A. Holovaty, “librosa/librosa: 0.6.3,” 2018. [Online]. Available: https://doi.org/10.5281/zenodo.1252297

[14] V. Lostanlen, J. Salamon, A. Farnsworth, S. Kelling, and J. P. Bello, “Birdvox-DCASE-20k: A dataset for bird audio detection in 10-second clips,” 2018. [Online]. Available: https://zenodo.org/record/1208080

[15] ——, “BirdVox-full-night: A dataset and benchmark for avian flight call detection,” in *Proc. IEEE ICASSP*, April 2018.

[16] K. J. Piczak, “ESC: Dataset for environmental sound classification,” in *Proceedings of the 23rd ACM international conference on Multimedia*. ACM, 2015, pp. 1015–1018.

[17] C. Clark, R. Charif, D. Hawthorne, A. Rahaman, G. Givens, J. George, and C. Muirhead, “Acoustic data from the Spring 2011 bowhead whale census at Point Barrow, Alaska,” *Journal of Cetacean Research and Management*, 2019.

[18] D. Santos-Domínguez, S. Torres-Guijarro, A. Cardenal-López, and A. Pena-Gimenez, “ShipsEar: An underwater vessel noise database,” *Applied Acoustics*, vol. 113, pp. 64–69, 2016.

[19] S. Böck and G. Widmer, “Maximum filter vibrato suppression for onset detection,” in *Proc. DAFx*, 2013.

[20] R. P. Hodges, *Underwater acoustics: analysis, design, and performance of sonar*. Wiley, 2010.

[21] V. Lostanlen, J. Salamon, A. Farnsworth, S. Kelling, and J. P. Bello, “Robust sound event detection in bioacoustic sensor networks,” *Under review*, 2019.

[22] M. Towsey, L. Zhang, M. Cottman-Fields, J. Wimmer, J. Zhang, and P. Roe, “Visualization of long-duration acoustic recordings of the environment,” *Procedia Computer Science*, vol. 29, pp. 703–712, 2014.

ACOUSTIC SCENE CLASSIFICATION FROM BINAURAL SIGNALS USING CONVOLUTIONAL NEURAL NETWORKS

Rohith Mars, Pranay Pratik, Srikanth Nagisetty, Chong Soon Lim

Panasonic R&D Center Singapore
{rohith.mars, pranay.pratik, srikanth.nagisetty, chongsoon.lim}@sg.panasonic.com

ABSTRACT

In this paper, we present the details of our proposed framework and solution for the DCASE 2019 Task 1A - Acoustic Scene Classification challenge. We describe the audio pre-processing, feature extraction steps and the time-frequency (TF) representations employed for acoustic scene classification using binaural recordings. We propose two distinct and light-weight architectures of convolutional neural networks (CNNs) for processing the extracted audio features and classification. The performance of both these architectures are compared in terms of classification accuracy as well as model complexity. Using an ensemble of the predictions from the subset of models based on the above CNNs, we achieved an average classification accuracy of **79.35%** on the test split of the development dataset for this task. In the Kaggle's private leaderboard, our solution was ranked 4th with a system score of **83.16%** — an improvement of $\approx 20\%$ over the baseline system.

Index Terms— DCASE 2019, acoustic scene classification, convolutional neural networks, binaural signals, mixup.

1. INTRODUCTION

Humans perceive their surroundings primarily through the visual and audio cues presented to their eyes and ears, respectively. Though visual stimuli provide a substantial amount of information regarding the scene, it is inarguable that audio cues also play a vital role in determining the type of the environment we are immersed in. For example, an immersive experience through virtual reality (VR) is deemed satisfactory only when the associated audio aligns with the visual scene. In a simpler scenario, a person standing near a beach with eyes closed can easily infer that they are near the shore from the repetitive sound pattern of the waves crashing on the rocks or from the sound of the seagulls. It is easy to conclude that acoustic characteristics of certain environments have their own unique signature, which aids humans in distinguishing an audio scene from another.

The objective of acoustic scene classification is to empower a machine to automatically recognize the audio scene from the audio signals they are provided with. Such "machine listening" tasks fall under the broader umbrella of computational auditory scene analysis (CASA) [1, 2]. Over the past few years, advancement of deep learning algorithms along with availability of large datasets and increase in computational power has helped to further push the performance of such machine listening systems.

The Detection and Classification of Acoustic Scenes and Events (DCASE) challenge, has played a major role in providing common datasets for development, setting algorithmic benchmarks and furthering the research in deep learning for audio signals, especially for tasks such as scene classification, event detection and audio tagging. For the scene classification task introduced in DCASE 2013 challenge, the best performing algorithm used a machine learning approach, more specifically, a treebagger classifier using hand-crafted features extracted from the audio recordings [3]. In the 2016 edition, most of the solutions involved deep learning approach, with the top performance achieved by a fusion of convolutional neural network (CNN) and binaural I-vectors [4]. Continuing a similar trend from the previous year, the top performing algorithms for the DCASE 2017 scene classification task employed CNNs for the audio spectrogram representations [5] and used generative adversarial network (GAN) for data augmentation [6]. In the 2018 edition of the DCASE challenge, the best performance was achieved by use of CNNs with adaptive division of multiple spectrogram respresentations [7].

Similar to the previous editions, the DCASE 2019 challenge [8] consists of separate challenge tasks, with Acoustic Scene Classification being one among them. This task is further divided into three subtasks, wherein we participate in the DCASE 2019 Task 1A. In this subtask, the development data and evaluation data are obtained from the same recording device. We built our proposed solution framework inspired by the success of utilizing 2-D time-frequency (TF) representations of binaural recordings with CNNs for classification. However, instead of using computationally expensive audio feature extraction steps and CNN models with large number of parameters, we utilize audio feature extraction with minimal computation and light-weight CNN models. In addition, we also explore the effect of using rectangular kernels and non-uniform pooling operations in CNN architecture as opposed to conventional square kernels and uniform pooling for achieving the same task and compare these distinct architectures in terms of accuracy as well as complexity. We obtain the final prediction by ensembling the outputs from the best-performing models identified using the test split from the development set.

We begin this paper by describing our audio pre-processing, feature extraction and data augmentation steps in Section 2. In Section 3, we provide details on the two separate CNN architectures used in our solution. The details of the database provided for the challenge, the accuracy achieved by our solution on the test split of the development dataset as well as the Kaggle's private leaderboard are provided in Section 4. Finally, the conclusions are presented in Section 5.

2. FEATURE EXTRACTION & DATA AUGMENTATION

In this section, we describe the audio pre-processing steps as well as the binaural audio feature extraction process. The extracted features are then provided as input to the CNN for predicting the acoustic scene class. In addition, we also discuss the data augmentation step

https://doi.org/10.33682/6c9z-gd15

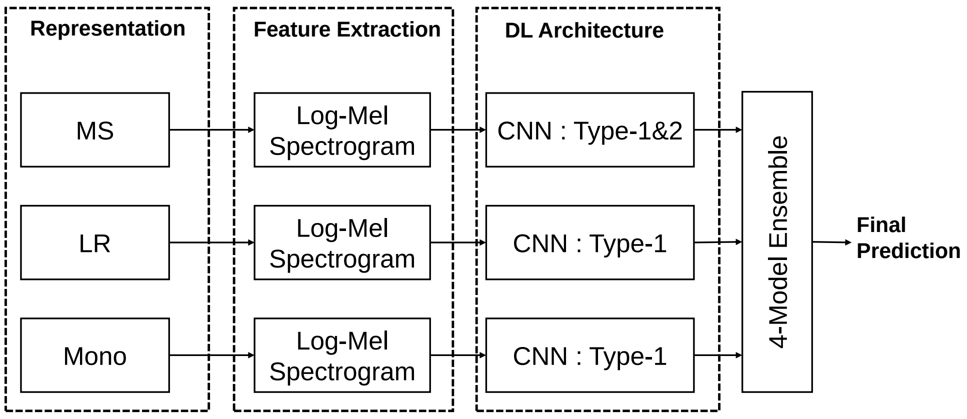


Figure 1: Our overall solution framework consisting of binaural audio representations, feature extraction steps, multiple CNNs & model ensembling for final prediction of the acoustic scene.

used to improve the model's generalization to unseen data.

2.1. Binaural audio feature extraction

In our proposed system, we use the originally provided audio recordings sampled at 48 kHz without down sampling. The time domain audio signals are then normalized by amplitude and then converted to the TF representation to extract the temporal and spectral characteristics. As such, we first compute the short-time Fourier transform (STFT) of the normalized time-domain audio signal. The frame size for the STFT operation is fixed at 2048, with a frame overlap of 50% and hanning window. Due to the large dimension of the linear STFT representation, we further compute the corresponding Mel-spectrogram representation using 128 Mel-bands. The use of Mel-scale is more close to the human auditory system and provides additional advantage of having smaller dimensionality than conventional linear STFT. As the final step, we compute the $\mathrm{Log}(\cdot)$ of the Mel-spectrogram to reduce the dynamic range of the values and make the feature space more Gaussian in distribution as reported in [9]. On the computed Log Mel-spectrograms, we performed feature normalization to achieve zero mean with unit variance. This mean and standard deviation was computed using the training data and the same were used on the validation/test split.

Since the recorded data in this task is binaural in nature, the Log Mel-spectrograms are computed separately for the Left (L), Right (R), Mid (M), Side (S) representation. In addition, we also use the conventional mono representation of the binaural signal for computing the Log Mel-spectrogram. The MS representation is obtained from the LR representation as follows

$$\begin{aligned} M &= (L+R)/2 \\ S &= (L-R)/2. \end{aligned} \tag{1}$$

The use of LR, MS & mono representations for audio classification have been explored in earlier editions of DCASE audio scene classification task and has been reported to achieve superior performance [5]. However, our framework differs from [5] in few aspects. Firstly, we do not split the 10 second audio clips to smaller audio chunks. In other words, the entire 2-D Log Mel-spectrogram of size 128×469 per channel is provided as input to the CNN. Secondly, instead of using each channel as a separate input to the CNN and concatenating the corresponding CNN layers at a later stage in the network, we combine the individual channels at the first layer of convolution itself. Finally, we do not perform the background subtraction (BS) method as well as the harmonic percussive source separation (HPSS) on the mono representation used in [5] as they involve further processing after the downmix operation and are thus computationally more expensive. The entire framework of our solution depicting the audio representations, feature extraction, multiple CNNs and ensembling step is shown in Figure 1.

2.2. Data Augmentation

It is well-known that deep learning algorithms perform well when they are trained using large amounts of data. However, depending on the task, the amount of labelled data for training maybe limited or constrained. As a result, deep learning algorithms may not fully capture the intra-class and inter-class variations in the data. In such situations, data augmentation plays a crucial role by increasing the amount and variance in the training data. For acoustic signals, conventional augmentation techniques include pitch shifting, time stretching, adding background noise and dynamic range modulation [10]. Another approach for augmentation is to mix the clips of same acoustic class by splitting and shuffling [11]. Recently, the use of GANs for data augmentation has also been explored in [6].

In our proposed method, to ensure a better generalization capability for the neural network, we perform the augmentation method proposed in [12], termed as *mixup*. The use of *mixup* for improving the performance of acoustic scene classification task has been explored in [13, 14]. In *mixup*, two random training examples (x_i, x_j) are weighted and mixed together along with their class labels (y_i, y_j) to form virtual training examples $(\tilde{x}, \tilde{y})$ as

$$\begin{aligned} \tilde{x} &= \lambda x_i + (1-\lambda)x_j \\ \tilde{y} &= \lambda y_i + (1-\lambda)y_j, \end{aligned} \tag{2}$$

with $\lambda \in [0,1]$ acquired by sampling from the β distribution $\beta(\alpha, \alpha)$, with α being a hyper parameter.

Input: $128 \times 469 \times 2$ or $128 \times 469 \times 1$
Conv2D (64, $\{3 \times 3\}$), BatchNormalization, ReLU MaxPooling2D $\{2 \times 2\}$
Conv2D (128, $\{3 \times 3\}$), BatchNormalization, ReLU MaxPooling2D $\{2 \times 2\}$
Conv2D (256, $\{3 \times 3\}$), BatchNormalization, ReLU MaxPooling2D $\{2 \times 2\}$
Conv2D (512, $\{3 \times 3\}$), BatchNormalization, ReLU
Conv2D (512, $\{3 \times 3\}$), BatchNormalization, ReLU
GlobalMaxPool2D
Dense(256, ReLU)
Dense(10, Softmax)

Table 1: CNN: Type-1 Architecture. Conv2D($n, \{p{\times}q\}$) represents 2D convolution operation with n filters of kernel size $p \times q$.

3. CNN ARCHITECTURE

The audio features extracted by the pre-processing and data augmentation steps explained in Section 2 are provided as input to a CNN. In this work, we experimented with two distinct architectures of CNN. The first CNN architecture is similar to the VGG-style architecture, which uses a constant $\{3 \times 3\}$ square-shaped kernels. However, we use significantly less number of convolution and dense layers as compared to the original VGG-16 architecture. In the second CNN architecture, we employ rectangular kernels for convolution and non-uniform pooling operation for the frequency and temporal dimension of the audio spectrogram. CNNs with rectangular kernels have been previously used for a variety of tasks such as scene classification [15, 16], keyword spotting [17] and music genre classification [18, 19]. The use of such rectangular kernels help to treat the spectral and temporal components of the audio with different context sizes as compared to square-shaped kernels. In the following subsections, we further elaborate on the above two CNN architectures.

3.1. CNN: Type-1

This CNN architecture is similar to the VGG-style architecture. It consists of 5 convolutions with increasing number of filters, i.e., $(64, 128, 256, 512, 512)$. The kernel size is chosen as $\{3 \times 3\}$ and is kept constant for all the convolution layers. We also apply batch normalization [20] and ReLU non-linear activation for each convolution layers. Max pooling $\{2 \times 2\}$ operation is performed at the first three convolution layers to reduce the dimensionality. Finally we perform global max pool operation to gather all the components, which is then connected to a dense layer of 256 units with ReLU activation. The output layer consists of 10 units corresponding to the number of scene classes and undergoes softmax operation to obtain the prediction probabilities. The CNN: Type-1 has ≈ 4 million parameters.

Input: $128 \times 469 \times 2$ or $128 \times 469 \times 1$
Conv2D (64, $\{3 \times 7\}$), BatchNormalization, ReLU MaxPooling2D $\{3 \times 1\}$
Conv2D (128, $\{3 \times 1\}$), BatchNormalization, ReLU MaxPooling2D $\{4 \times 1\}$
Conv2D (256, $\{11 \times 1\}$, padding= "valid") BatchNormalization, ReLU
Conv2D (512, $\{1 \times 7\}$), BatchNormalization, ReLU
GlobalMaxPool2D
Dense(256, ReLU)
Dense(10, Softmax)

Table 2: CNN: Type-2 Architecture. Note that we use rectangular kernels and non-uniform pooling operation throughout the network.

3.2. CNN: Type-2

In this CNN architecture, we employ rectangular kernels instead of square kernels. It consists of 4 convolutions with increasing number of filters, i.e., $(64, 128, 256, 512)$. For the convolution layer-1, we apply a kernel of size $\{3 \times 7\}$ for low-level feature extraction, followed by a max pooling with size $\{3 \times 1\}$. After reducing the dimension in the frequency axis, convolutions with kernel size $\{3 \times 1\}$ is applied in convolution layer-2 to extract frequency patterns for each time-frame. We further reduce the dimension in the frequency axis by using $\{4 \times 1\}$ max pooling. In the convolution layer-3, we use a kernel size of $\{11 \times 1\}$ and perform "valid" convolutions. This step ensures that spectral patterns are learnt with entire frequency dimension being compressed. We do not perform pooling across time dimension and the last convolution layer uses filter size of $\{1 \times 7\}$ to learn only the temporal characteristics. Similar to CNN-1, we apply batch normalization and ReLU non-linear activation for each convolution layers. After the convolution layers, all the components are collected using the global max pooling operation, which is further input to a fully connected layer of 256 units with ReLU activation. The output layer consists of 10 units corresponding to the number of scene classes and undergoes softmax operation to obtain the prediction probabilities. The CNN: Type-2 uses $\approx$ 1.4 million parameters. This is 3 times lower number of parameters as compared to CNN: Type-1 and therefore, a comparatively less-complex architecture.

By employing the above distinct architectures of CNNs, we expect each of them to learn different low-level and high-level features of the audio spectrogram. While CNN: Type-1 treats the frequency and temporal dimension equally using square kernels, CNN:Type-2 treats these dimensions with different context sizes using rectangular kernels. Note that for LR & MS representations, the input size is $128 \times 469 \times 2$, with each channel arranged back to back and we combine the individual channels at the first layer of convolution itself. For the case of mono representation, the input size is $128 \times 469 \times 1$.

Methods	Mean accuracy (%)
Baseline	62.5
CNN:Type-1 - Mono	73.04
CNN:Type-1 - LR	74.20
CNN:Type-1 - MS	75.19
CNN:Type-2 - Mono	69.86
CNN:Type-2 - LR	69.79
CNN:Type-2 - MS	72.90
Ensemble	**79.35**

Table 3: Mean accuracy on the test split from the development set using the baseline system, the proposed CNN architectures for each representation and after ensembling. For ensembling, the top-4 best performing models are selected.

Scene Label	Baseline system Accuracy (%)	Proposed system Accuracy (%)
Airport	48.4	90.2
Bus	62.3	92.0
Metro	65.1	74.7
Metro station	54.5	80.2
Park	83.1	65.1
Public square	40.7	80.5
Shopping mall	59.4	75.5
Street pedestrian	60.9	87.5
Street traffic	86.7	80.2
Tram	64.0	79.4
Average	**62.5**	**79.3**

Table 4: Comparison of class-wise accuracy on the test split from the development set using the baseline system and the proposed system after ensembling.

4. DATABASE & RESULTS

The TAU Urban Acoustic Scenes 2019 dataset [21] for this task is the extension of the 2018 TUT Urban Acoustic Dataset, consisting of binaural audio recordings from various acoustic scenes in different cities. The recordings were made using the Soundman OKM II Klassik/studio A3, electret binaural microphone and a Zoom F8 audio recorder using 48 kHz sampling rate and 24 bit resolution. For each acoustic scene class, such recordings were collected from different locations in the city. Each original recordings were split into segments with a length of 10 seconds as development and evaluation set. The audio scenes are namely {"Airport", "Indoor shopping mall", "Metro station", "Pedestrian street", "Public square", "Street with medium level of traffic", "Travelling by a tram", "Travelling by a bus", "Travelling by an underground metro" & "Urban park"}.

From the training split of the development set, we use a random split of 15% as the hold-out validation set for hyperparameter tuning. We do not utilize any external data or pre-trained models for training our system. The optimization is performed using the Adam optimizer [22], with an initial learning rate of 0.001 and a maximum epoch of 200 with a batch size of 32 samples. We reduce the learning rate by a factor of 0.1 if the validation loss does not decrease after 5 epochs. We use early stopping method to stop the training if the validation loss does not decrease after 10 epochs. The categorical cross-entropy is chosen as the loss function. For the data augmentation step using *mixup*, we kept $\alpha = 0.3$ for all the models. The baseline system [23] also used a CNN based approach on Log Mel spectrogram of 40 bands, consisting of two CNN layers and one fully connected layer. We chose Keras [24] as our deep learning framework for all experiments.

For the CNN: Type-1, using the mono representation, we get an average accuracy of 73.04%. In comparison, the MS and LR representation, we achieve a mean accuracy of 75.19% and 74.2%, respectively. For CNN: Type-2, the mean accuracy are 69.86%, 72.9% & 69.79% using the mono, MS and LR representation, respectively. From both these results, we conclude that the MS representation is best suited for this task. The performance drop in CNN: Type-2 can be attributed to the low-complex architecture used. We also note that better tuning of the parameters may be required to enable this CNN to better capture the spectral and temporal patterns which can lead to performance improvement as well.

Based on performance on the test split, we select the top-4 best performing models (CNN: Type-1 : MS, LR, Mono, CNN: Type-2: MS) for the final ensembling. We ensemble the output predictions from each of the 4 models by computing the geometric mean of the predictions. The final prediction is done by selecting the class with maximum probability on the ensembled prediction. After this ensembing step, we obtain a mean accuracy of **79.35%** on the test split of the development set.

The classification results for all the proposed models and ensembled solution compared with the baseline system are shown in Table 3. The class-wise accuracy of the proposed system after ensembling for the test split is compared with the baseline system is shown in Table 4. It can be seen that the proposed system achieves better accuracy for all classes except for "Park" and "Street traffic". For the evaluation on Kaggle leaderboard set, we used the entire development set for training the proposed system. In the Kaggle's private leaderboard [25], the baseline system achieved a system score of 63.00%. In comparison, our solution was ranked 4th with system score of **83.16%**, thereby achieving an improvement of $\approx 20\%$ over the baseline system.

5. CONCLUSIONS

In this paper, we provided the details of our solution to the DCASE2019 Task1A - Acoustic Scene Classification. We described the audio pre-processing, feature extraction steps and the various binaural representations used as input the neural network. The architecture of two distinct and light-weight CNNs used for the classification are described. We compared the performance of these CNNs on each binaural representations in terms of classification accuracy as well as their complexity. After ensembling multiple models, our system achieves an average accuracy of **79.35%** on the test split from the development set. The solution was ranked 4th with system score of **83.16%** in the Kaggle's private leaderboard.

6. REFERENCES

[1] D. Wang and G. J. Brown, *Computational auditory scene analysis: Principles, algorithms, and applications*. Wiley-IEEE press, 2006.

[2] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational analysis of sound scenes and events*. Springer, 2018.

[3] D. Li, J. Tam, and D. Toub, "Auditory scene classification using machine learning techniques," in *AASP Challenge on Detection and Classification of Acoustic Scenes and Events*. IEEE, 2013.

[4] H. Eghbal-Zadeh, B. Lehner, M. Dorfer, and G. Widmer, "CP-JKU submissions for DCASE-2016: a hybrid approach using binaural i-vectors and deep convolutional neural networks," DCASE2016 Challenge, Tech. Rep., September 2016.

[5] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *the Detection and Classification of Acoustic Scenes and Events (DCASE)*, pp. 1–5, 2017.

[6] S. Mun, S. Park, D. K. Han, and H. Ko, "Generative adversarial network based acoustic scene training set augmentation and selection using svm hyper-plane," *Proc. DCASE*, pp. 93–97, 2017.

[7] Y. Sakashita and M. Aono, "Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divisions," *IEEE AASP Challenge on DCASE 2018 technical reports*, 2018.

[8] http://dcase.community/challenge2019/.

[9] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "A comparison of audio signal preprocessing methods for deep neural networks on music tagging," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 1870–1874.

[10] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[11] T. Inoue, P. Vinayavekhin, S. Wang, D. Wood, N. Greco, and R. Tachibana, "Domestic activities classification based on cnn using shuffling and mixing data augmentation," *DCASE 2018 Challenge*, 2018.

[12] H. Zhang and et.al, "mixup: Beyond empirical risk minimization," *International Conference on Learning Representations*, 2018.

[13] K. Xu, D. Feng, H. Mi, B. Zhu, D. Wang, L. Zhang, H. Cai, and S. Liu, "Mixup-based acoustic scene classification using multi-channel convolutional neural network," in *Pacific Rim Conference on Multimedia*. Springer, 2018, pp. 14–23.

[14] J. J. Huang and J. J. A. Leanos, "Aclnet: efficient end-to-end audio classification cnn," *arXiv preprint arXiv:1811.06669*, 2018.

[15] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2015, pp. 1–6.

[16] D. Battaglino, L. Lepauloux, N. Evans, F. Mougins, and F. Biot, "Acoustic scene classification using convolutional neural networks," *IEEE AASP Challenge on Detec*, 2016.

[17] T. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," *Proc. Interspeech*, 2015.

[18] A. Schindler, T. Lidy, and A. Rauber, "Comparing shallow versus deep neural network architectures for automatic music genre classification," in *9th Forum Media Technology (FMT2016)*, vol. 1734, 2016, pp. 17–21.

[19] J. Pons, T. Lidy, and X. Serra, "Experimenting with musically motivated convolutional neural networks," in *2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI)*. IEEE, 2016, pp. 1–6.

[20] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[21] T. Heittola, A. Mesaros, and T. Virtanen, "TAU Urban Acoustic Scenes 2019, Development dataset," Mar. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2589280

[22] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[23] http://dcase.community/challenge2019/task-acoustic-scene-classification#baseline-system.

[24] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[25] https://www.kaggle.com/c/dcase2019-task1a-leaderboard/leaderboard.

FIRST ORDER AMBISONICS DOMAIN SPATIAL AUGMENTATION FOR DNN-BASED DIRECTION OF ARRIVAL ESTIMATION

Luca Mazzon[1,2], *Yuma Koizumi*[1], *Masahiro Yasuda*[1], *and Noboru Harada*[1]

[1]NTT Media Intelligence Laboratories, Tokyo, Japan
[2]University of Padova, Padua, Italy

ABSTRACT

In this paper, we propose a novel data augmentation method for training neural networks for Direction of Arrival (DOA) estimation. This method focuses on expanding the representation of the DOA subspace of a dataset. Given some input data, it applies a transformation to it in order to change its DOA information and simulate new potentially unseen one. Such transformation, in general, is a combination of a rotation and a reflection. It is possible to apply such transformation due to a well-known property of First Order Ambisonics (FOA). The same transformation is applied also to the labels, in order to maintain consistency between input data and target labels. Three methods with different level of generality are proposed for applying this augmentation principle. Experiments are conducted on two different DOA networks. Results of both experiments demonstrate the effectiveness of the novel augmentation strategy by improving the DOA error by around 40%.

Index Terms— First Order Ambisonics, direction of arrival, deep learning, data augmentation

1. INTRODUCTION

Direction of arrival (DOA) estimation is the task of detecting the spatial position of a sound source with respect to a listener. The approaches that has been adopted to solve this problem can be classified in two main categories: parametric-based methods, like multiple signal classification (MUSIC) [1] and others [2–4], and deep neural network (DNN)-based methods [5–17]. DNN-based models often combine DOA estimation with other tasks such as sound activity detection (SAD), estimation of number of active sources and sound event detection (SED) [11–13]. In particular, Sound Event Localization and Detection was the task 3 of Detection and Classification of Acoustic Scenes and Events 2019 Challenge (DCASE2019 Challenge) [18].

In machine learning, *data augmentation* is an effective strategy to overcome the lack of data in the training set and prevent overfitting. For example SpecAugment [20], a recently published augmentation method based on time warping and time and frequency block masking of the spectrogram, achieved state of the art performance on the Speech recognition task. DCASE2018 Task2 Challenge (about audio tagging) winner [21] used mixup augmentation [22].

While data augmentation is effective for sound event detection and similar tasks, none of the documented strategies is capable of effectively increasing the spatial representativeness of a dataset, i.e. increasing the number of DOAs represented in the dataset. The critical point of the problem is that when the observed signals are modified by a data augmentation method, it must be guaranteed that the relationship between the DOA information carried by the signal and the corresponding labels is maintained. For example, augmentation techniques such as SpecAugment, phase-shifting and mixup can indeed influence DOA, although it's hard to analytically compute the new true DOA labels. In fact, according to the technical reports of DCASE 2019 task3, SpecAugment has affected adversely for DOA estimation even though it is effective for SED [19, 23].

In this paper, we propose *FOA Domain Spatial Augmentation*, a novel augmentation method based on the well-known rotational property of First Order Ambisonics (FOA) sound encoding. The basic idea of the method is to apply some transformations to the FOA channels (and corresponding labels) to modify and simulate a new DOA of the recorded sounds in a predictable way. Such transformations are: channel swapping and inversion, application of a rotation formula (i.e. Rodrigues' rotation formula) and multiplication by an orthonormal matrix, which correspond to rotations and reflections of the sound sources positions with respect to a reference system centered on the listener.

2. FIRST ORDER AMBISONICS

First-Order Ambisonic (FOA) is a digital audio encoding which describes a soundfield [24]. It has origin in the B-Format, which encodes the directional information on four channels W, X, Y and Z [24]. W carries omnidirectional information, while channels X, Y and Z carry the directional information of the sound field along the Cartesian axes of a reference system centered on the listener [24].

Adopting the same notation and convention of the dataset used for the following experiments [25], the spatial responses (steering vectors) of the FOA channels are $H_1(\phi, \theta, f) = 1$, $H_2(\phi, \theta, f) = \sqrt{3} * \sin\phi * \cos\theta$, $H_3(\phi, \theta, f) = \sqrt{3} * \sin\theta$, and $H_4(\phi, \theta, f) = \sqrt{3} * \cos\phi * \cos\theta$, where ϕ and θ are the azimuth and elevation angles of a sound source, f is frequency and $*$ is used for the multiplication operation. As it is noticeable from the expressions, FOA channels can be seen as the projections of the sound sources to the three dimensional Cartesian axes, with H_1 corresponding to channel W, H_2 to channel Y, H_3 to channel Z and H_4 to channel X. Thus, indicating with $\mathbf{S} = \{S_1, ..., S_n\}$ a set of sound sources in their STFT domain, FOA channels can be written as a sum of each source and its steering vector, that is $X = \frac{1}{N}\sum_{n=1}^{N} H_4(\phi_n, \theta_n, f) * S_n$, where $N = |\mathbf{S}|$, and ϕ_n and θ_n are the azimuth and elevation of S_n, respectively.

3. FOA DOMAIN SPATIAL AUGMENTATION

The goal of the method is, from the audio recordings in the dataset, to generate new ones with different DOA information. More specif-

https://doi.org/10.33682/3qgs-e216

ically, the problem consists in simulating a new set of spatial responses $\{H_i(\phi'_n, \theta'_n, f)\}_{i=2}^{4}$ corresponding to new DOA labels $\{\phi'_n, \theta'_n\}_{n=1}^{N}$ for the audio recordings by applying a transformation directly to the FOA channels. It is a known property of FOA that, since it encodes a soundfield rather than the sources themselves, it is possible to apply some operations directly to the channels [24], such as rotations and reflections. There are several ways to apply these transformations, leading to different augmentation strategies with different pros and cons. In the following, three strategies are proposed and compared.

3.1. First method: 16 patterns

The *16 patterns* method simply consists in applying to the data one of the 16 prefixed channel transformations summarized in Table 1, where $\leftarrow$ indicates an assignment. The basic operations used in this method are channel swapping (e.g. $X' \leftarrow Y, Y' \leftarrow X$) and channel sign inversion (e.g. $Z' \leftarrow -Z$) or a combination between the two. Using this set of operations, it is possible to obtain 8 rotations about the z axis and 2 reflections with respect to the $xy : z = 0$ plane, for a total of 16 augmentation patterns (i.e. 15 new patterns plus the original one). The corresponding transformations for the labels are also reported in Table 1. In particular, the listed transformations correspond to the translations of $+0$, $+\pi$, $+\frac{\pi}{2}$ and $-\frac{\pi}{2}$ of the azimuth angles ϕ and $-\phi$ and to the pair of opposites ϕ and $-\phi$.

The main advantage of this algorithm, other than it's simplicity and straightforward implementation, is the possibility of it being applied to many pre-computed features, such as logmel magnitude spectrogram or phase spectrogram, since the corresponding transformations in the feature-domain are straightforward to compute (channel swapping maps to the same channel swapping, channel sign inversion maps to identity for magnitude and to a 180 degrees difference for phase). Another advantage is that it is easy to control that mapped angles remain in the same domain as the original ones. For example, in the dataset in use for DCASE2019 Challenge task3, all angles are multiples of 10 degrees and elevation angles range from -40 to $+40$ degrees. It is easy to see that the augmented angles maintain the same domain. One more important advantage of this method is that it can be applied independently on the number of the maximum number of overlapping sound sources, which is a complication for the next proposed method.

3.2. Second method: Labels First

In *Labels First* method, the basic idea is to first decide the target augmented labels, than to apply a transformation to the data accordingly. The critical aspect of this method is that while for azimuth this is always possible independently on the number of overlapping sources, it isn't the same for elevation. The reason is that when modifying the azimuth coordinates by a fixed amount by means of a rotation, *z-axis* is the common rotational axis for all the sources, while for modifying only the elevation coordinate by a fixed amount by means of a rotation, for each source, an appropriate rotation axis must be selected.

Keeping into consideration this critical aspect, assuming at first to have sound files with non-overlapping sound events, the proposed algorithm for this method is follows. For convenience, it is divided in two steps in which azimuth and elevation are augmented separately. In the first step, at first a random angle α is selected and used to translate, at each time step t with arbitrary range, the azimuth labels:

```
$\alpha \leftarrow \texttt{random}(0, 2\pi)$
$\phi'_t \leftarrow \phi_t \oplus \alpha$
```

where $\oplus$ here indicates an addition with a wrap-around on the domain $(-\pi, \pi)$, i.e. $(\phi_t + \alpha + \pi) \bmod 2\pi - \pi$. At this point, the rotation matrix around *z-axis* R_z is computed:

$$R_z = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{1}$$

and applied to the channels at each time step t:

$$\mathbf{v}'_t = R_z \mathbf{v}_t, \tag{2}$$

where $\mathbf{v}_t = (X_t, Y_t, Z_t)^\top$ denotes original channels and $\mathbf{v}'_t = (X'_t, Y'_t, Z'_t)^\top$ denotes azimuth-augmented channels. In the second step, the elevation coordinate is augmented. At first, a random augmentation angle β is selected. To do so, elevation labels in the selected time range (e.g. a batch) is inspected and maximum and minimum values M_e and m_e are extracted. The elevation angle, by definition, has range $\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, but in some datasets like the one in use for DCASE2019 Challenge task3, it might have a custom range (m_{er}, M_{er}). In order not to go out of this range, the augmentation angle β is extracted randomly in the range $(m_{er} - m_e, M_{er} - M_e)$[1]. At this point, elevation labels are updated:

```
$\beta \leftarrow \texttt{random}(M_{er} - M_e, m_{er} - m_e)$
$\theta'_t \leftarrow \theta_t + \beta$
```

Secondly, at each time step t, the rotation axis for augmenting elevation is computed. This axis is defined by the unit vector perpendicular to the one along the azimuthal axis, oriented properly so that a rotation of the audio channels by an angle β corresponds to the same increment of the elevation label. It can be easily verified with the right-hand rule that this unit vector corresponds to the azimuthal one rotated by $-\frac{\pi}{2}$ about the *z-axis*:

$$\mathbf{u}_t = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\phi'_t \\ \sin\phi'_t \\ 0 \end{pmatrix} = \begin{pmatrix} \sin\phi'_t \\ -\cos\phi'_t \\ 0 \end{pmatrix}, \tag{3}$$

where the first term means $R_z(-\pi/2)$. Now, Rodrigues' rotation formula is applied to the $\mathbf{v}'_t$, we obtain full-augmented channels:

$$\mathbf{v}''_t = \mathbf{v}'_t \cos\beta + (\mathbf{u}_t \times \mathbf{v}'_t) \sin\beta + \mathbf{u}_t\, (\mathbf{u}_t \cdot \mathbf{v}'_t)(1 - \cos\beta), \tag{4}$$

where $\times$ and $\cdot$ denote the cross-product and the inner-product, respectively.

The main advantage of this method is the high control over the augmented labels. For example, it allows for generating new labels which belong to the same domain of the original ones (e.g only multiple of $10°$ and elevation restricted to the range $(-40°, 40°)$, such as in the dataset used for the experiments [25]. The main disadvantage is that it is best suitable for non-overlapping sound events. There are some workarounds to adapt it to sound recordings with multiple overlapping sound events, though. Some possibilities are to apply it only to time frames with one event, to check for the somewhat rare cases in which all of time events share the same azimuth coordinates or to apply a hybrid strategy such as applying the 16 patterns method only for elevation augmentation or considering only one of the overlapping sources and computing labels for the others sources as in Channels First.

[1]In order to maximize the augmentation range, one could segment the audio recordings in frames containing the single sources. Alternatively, one could accept to extend the elevation domain of the dataset and agnostically select a fixed range for the augmentation angle β, which is convenient when augmenting an entire audio file altogether, as done in experiment 2.

Table 1: Sixteen patterns of simple spatial augmentation. Swap(X, Y) denotes $X' \leftarrow Y$ and $Y' \leftarrow X$.

	$\phi - \pi/2$	ϕ	$\phi + \pi/2$	$\phi + \pi$
θ	Swap$(-X, Y)$	original	Swap$(X, -Y)$	Swap$(-X, -Y)$
$-\theta$	Swap$(-X, Y)$, $Z' \leftarrow -Z$	$Z' \leftarrow -Z$	Swap$(X, -Y)$, $Z' \leftarrow -Z$	Swap$(-X, -Y)$, $Z' \leftarrow -Z$
	$-\phi - \pi/2$	$-\phi$	$-\phi + \pi/2$	$-\phi + \pi$
θ	Swap$(X, -Y)$	$Y' \leftarrow -Y$	Swap(X, Y)	$X' \leftarrow -X$
$-\theta$	Swap$(-X, -Y)$, $Z' \leftarrow -Z$	$Y' \leftarrow -Y, Z' \leftarrow -Z$	Swap(X, Y), $Z' \leftarrow -Z$	$X' \leftarrow -X, Z' \leftarrow -Z$

3.3. Third method: Channels First

Channels first is the most general case of FOA Domain Spatial Augmentation. This method doesn't depend on the number of overlapping sources, but the control over labels is almost completely lost.

The procedure is as follows. A random (3×3) orthonarmal matrix R is selected. An orthonormal matrix R is a matrix such that $HH^\top = I$ and $det(H) = \pm 1$. This can be done by selecting a random (3×3) matrix and then orthonormalizing it with the Graham-Schmidt method. Augmented channels $\mathbf{v}'$ are then computed as:

$$\mathbf{v}' = R\,\mathbf{v}. \quad (5)$$

The same transformation is also applied to the labels $y = (\phi, \theta)^\top$, in Cartesian coordinates[2]:

$y_c \leftarrow \mathrm{to\_cartesian}(y)$
$y'_c \leftarrow R\, y_c$
$y' \leftarrow \mathrm{to\_spherical}(y'_c)$

An orthonormal matrix expresses a general rotoreflection. This method allows generating the most number of augmentation patterns for any number of sources, but, since there is few to none control over the labels, it is recommended to use only with datasets without any restrictions on the labels' domain, as justified by the results of experiment 2.

4. EXPERIMENT

4.1. Experimental setup

We conducted our experiments, referred to as *Experiment 1* and *Experiment 2*, using two different DOA networks, one simpler, one more sophisticated, here referred to as *Simple DOAnet* and *Sophisticated DOAnet*. Both networks give as output a single pair of azimuth and elevation angles computed in a regression fashion and a sound activity detection value computed in a classification fashion. Both networks are trained using the maximum overlapping 1 audio files of the DCASE2019 Challenge dataset [25], and evaluated on DOA error (Er) and Frame-recall (FR). We used only overlap 1 files in order to be able to evaluate the effectiveness of FOA Domain Spatial Augmentation specifically for the DOA estimation task. In systems that are able to localize more than one overlapping source, such as SELDnet [11], other tasks, such as Sound Event Detection (SED), might be influenced by the augmentation strategy and at the same time influence the performance on DOA estimation.

4.1.1. Experiment 1

Simple DOAnet has a convolutional recurrent neural network (CRNN) as a core structure, as in [10–12, 19, 26]. Input features are logscale Mel-magnitude spectrogram (logmels) and Generalized Cross-Correlation Phase Transform (GCC-PHAT) of the mutual channels, as in [12, 26]. All wav-files were downsampled at a sampling rate of 32 kHz. The length of the short-time-Fourier-transform (STFT) and its shift length were 1024 and 640 (20 ms) points, respectively. The dimension of Mel bins for logmels and GCC-PHAT was 96. The DNN structure is a CRNN, similar to a SELDnet [11] without the SED branch and with a single class DOA output. The CRNN consists of 3 convolutional neural network (CNN) layers, 2 gated recurrent unit layers, and 2 fully-connected (FCN) layers, with the total number of parameters of 545K.

As a loss function, we compute the mean average errors (MAE) between true and predicted labels for both azimuth and elevation and mask them with the true sound activity labels, then sum them to the binary cross-entropy loss of the sound activity output. The model is trained adopting the four cross-validation folds defined in [25] for 400 epochs each and selecting the best model among the epochs according to the best validation loss. The conducted experiments on this model are 3: the first is without using FOA Domain Spatial Augmentation (No Aug), the second is applying the Labels First method on 50% of the input data (LF Half) and the third is applying the Labels First method on all of the input data (LF Full). Augmentation is applied on minibatches of 100 STFT frames (2s).

4.1.2. Experiment 2

Experiment 2 is conducted on *Sophisticated DOAnet*. *Sophisticated DOAnet* is a combination method of parametric-based and DNN-based DOA estimation [27]. Sound intensity vector (IV)-based DOA estimation is used as a base method and two CRNNs are used for denoising and dereverberation of IVs. Each CRNN consists of 5 CNN layers, 2 FCN layers, and 1 bidirectional long short-term memory layer, and the total number of parameters of *Sophisticated DOAnet* is 2.79M. The details of *Sophisticated DOAnet* are described in [27].

Training was performed on the standard cross-validation folds, and selecting the best model among the epochs according to the best validation loss. Four different runs of the training are performed on this network, one without augmentation (No Aug) and one for each of the methods described in Section 3: 16 Patterns (16P), Labels First (LF) and Channels First (ChF). Based on the results of *Experiment 1*, all data augmentation was performed on 50% of the input data directly on the full length wav files. For the Labels First method, elevation augmentation angle β was selected randomly between -20° and 20°, extending the range of elevation to $(-60^\circ, 60^\circ)$.

4.2. Results

Experiment 1 has mainly two purposes: demonstrate the effectiveness of FOA Domain Spatial Augmentation on training a simple

[2]Since distance from the listener is not relevant for the task, when converting to and from cartesian coordinates, we always assume the norm $r = 1$, that is we consider direction of arrivals as points on the unit sphere.

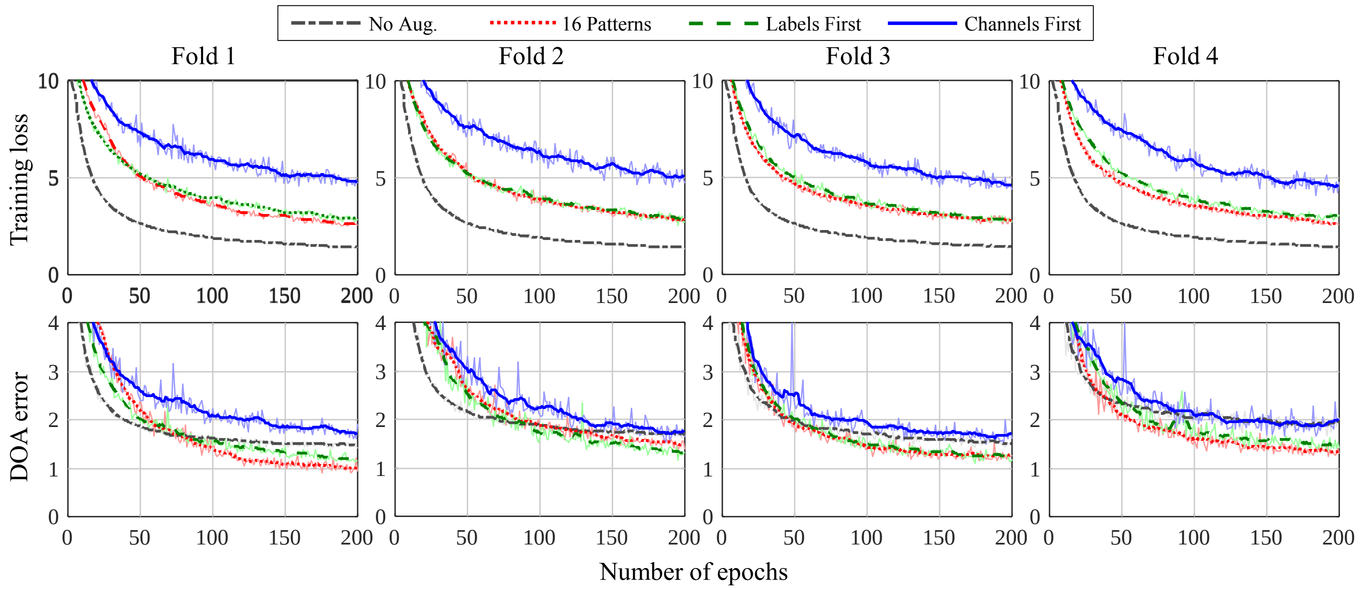


Figure 1: Training progress graph of experiment 2; training loss (top) and DOA error of validation set (bottom). It is apparent that the 16 Patterns method and the Labels First method performed better than without augmentation. The Channels First method lead to worse results, supposedly due to the over-extension of the labels domain and the consequent complication of the problem.

Table 2: Results of *experiment 1* on *Simple DOAnet*

		Fold 1	Fold 2	Fold 3	Fold 4	Ave.
No	Er	5.32	4.85	5.56	5.07	5.22
Aug.	FR(%)	97.78	98.46	97.79	97.67	97.93
LF	Er	**3.34**	**3.28**	**3.27**	**3.07**	**3.22**
Half	FR(%)	98.16	**98.89**	98.28	98.14	98.37
LF	Er	3.53	3.64	3.53	3.21	3.48
Full	FR(%)	**98.18**	98.74	**98.41**	**98.38**	**98.43**

Table 3: Results of *experiment 2* on *Sophisticated DOAnet*

		Fold 1	Fold 2	Fold 3	Fold 4	Ave.
No	Er	1.69	1.53	1.81	1.60	1.66
Aug.	FR(%)	96.91	96.46	97.14	97.50	97.00
16P	Er	**0.96**	**1.30**	**1.45**	**1.21**	**1.21**
	FR(%)	97.30	97.00	97.53	**97.70**	**97.39**
LF	Er	1.31	1.39	1.49	1.43	1.40
	FR(%)	**97.34**	94.16	**97.61**	97.14	96.56
ChF	Er	1.99	1.35	1.98	1.61	1.73
	FR(%)	96.45	**97.28**	97.24	96.96	96.98

DOA network and discover whether it is best to apply augmentation to all the data or, heuristically, to only half of the data. As reported in Table 2, both runs with the use of augmentation outperformed the run without using augmentation on all the cross-validation folds, decreasing the DOA error by 2° on average and increasing the Frame Recall of 0.5% on average. DOA error achieved the best results by augmenting 50% of the input data (0.24° better on average with respect to 100%), while augmenting all of the input data achieved the best result in terms of Frame Recall (0.06% better on average), although the results of these two runs were very close to each other.

Experiment 2 has the purpose of comparing the different FOA Domain Spatial Augmentation methods illustrated in Table 3 with each other as well as with non augmented data. The results reported in Table 3 show that in this case the 16 Patterns one was the best performing method, followed by the Labels First method, also scoring better than without augmentation in terms of DOA Error. As we expected Labels First to be the method achieving the best scores, we believe the penalty with respect to the expectation is due to the expansion of the labels domain, which means a more difficult problem to solve. As expected, the Channels First method was the least effective on this dataset, scoring worse with respect to non augmented data. It is safe to say that the determining factor for the underperformance is the too big of a difference in the labels domains of the augmented data and of the original data. In terms of frame recall, again 16 Patterns achieve the best score, although there aren't any particularly noticeable differences. In Figure 1, the training progress graphs of experiment 2 are reported. It can be clearly seen that in all the cross-validation folds there is a point since which DOA error on validation split is better with the 16 Patterns and Labels First methods rather than without augmentation.

5. CONCLUSIONS

In this paper, FOA Domain Spatial Augmentation, a novel data augmentation strategy, has been proposed. The basic idea of the method is to apply rotational transformations to the FOA channels and corresponding labels. We proposed three types of such transform: channel swapping and inversion, application of a rotation formula, and multiplication by an orthonormal matrix. It has been proven effective for training two different neural networks for the task of DOA estimation, improving the DOA error by 40%. Future research will be to further investigate the effectiveness of this augmentation strategy in different scenarios, for example with a dataset including all the possible DOAs or with overlapping sound events.

6. REFERENCES

[1] R. O. Schmidt, "Multiple Emitter Location and Signal Parameter Estimation" in *IEEE Transactions on Antennas and Propagation*, vol. AP-34, pp. 276–280, Mar. 1986.

[2] Y. Huang, J. Benesty, G. Elko, and R. Mersereati, "Real-time passive source localization: a practical linear-correction least-squares approach" in *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 8, 2001.

[3] M. S. Brandstein and H. F. Silverman, "A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1997.

[4] R. Roy and T. Kailath, "ESPRIT-estimation of signal parameters via rotational invariance techniques in *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 37, no. 7, 1989.

[5] M. Yiwere and E. J. Rhee, "Distance estimation and localization of sound sources in reverberant conditions using deep neural networks in *International Journal of Applied Engineering Research*, vol. 12, no. 22, 2017.

[6] E. L. Ferguson, S. B. Williams, and C. T. Jin, "Sound source localization in a multipath environment using convolutional neural networks" in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[7] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza, "A neural network based algorithm for speaker localization in a multi-room environment" in *IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2016.

[8] Y. Sun, J. Chen, C. Yuen, and S. Rahardja, "Indoor sound source localization with probabilisitic neural network" in *IEEE Transactions on Industrial Electronics*, vol. 29, no. 1, 2017.

[9] R. Roden, N. Moritz, S. Gerlach, S. Weinzierl, and S. Goetze, "On sound source localization of speech signals using deep neural networks" in *Deutsche Jahrestagung für Akustik (DAGA)*, 2015.

[10] S. Adavanne, A. Politis, and T. Virtanen, "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network" in *European Signal Processing Conference (EUSIPCO)*, 2018.

[11] S. Adavanne, A. Politis, J. Nikuunen, and T. Virtanen, "Sound Event Localization and Detection of Overlapping Sources Using Convolutional Recurrent Neural Networks" in *Journal of Selected Topics in Signal Processing*, 2018.

[12] Y. Cao, Q. Kong, T. Iqbal, F. An, W. Wang, and M. D. Plumbley, "Polyphonic sound event detection and localization using a two-stage strategy" *arXiv preprint, arXiv:1905.00268*, 2019.

[13] T. Hirvonen, "Classification of spatial audio location and content using convolutional neural networks in *Audio Engineering Society Convention 138*, 2015.

[14] J. H. DiBiase, H. F. Silverman, and M. S. Brandstein, "Robust localization in reverberant rooms" in *Microphone Arrays*, Springer, 2001.

[15] R. Takeda and K. Komatani, "Sound source localization based on deep neural networks with directional activate function exploiting phase information" in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[16] N. Yalta, K. Nakadai, and T. Ogata, "Sound source localization using deep learning models" in *Journal of Robotics and Mechatronics*, vol. 29, no. 1, 2017.

[17] W. He, P. Motlicek, and J. M. Odobez, "Deep neural networks for multiple speaker detection and localization" in *International Conference on Robotics and Automation (ICRA)*, 2018.

[18] `http://dcase.community/challenge2019/task-sound-event-localization-and-detection`.

[19] S. Kapka and M. Lewandowski, "Sound source detection, localization and classification using consecutive ensemble of CRNN models" in *Tech. Report of Detection and Classification of Acoustic Scenes and Events 2019 Challenge (DCASE2019 Challenge)*, Jun. 2019

[20] D. S. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, Ekin D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition" in *Interspeech 2019*, Apr. 2019

[21] I. Jeong and H. Lim, "Audio tagging system using densely connected convolutional networks" in *Proc. of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, Nov. 2018

[22] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: beyond empirical risk minimization" in *International Conference on Learning Representations (ICLR)*, Apr. 2018

[23] J. Zhang, W. Ding, and L. He, "Data augmentation and prior knowledge-based regularization for sound event localization and detection," in *Tech. Report of Detection and Classification of Acoustic Scenes and Events 2019 Challenge (DCASE2019 Challenge)*, Jun. 2019

[24] F. Hollerweger, "An introduction to higher order ambisonics", `https://pdfs.semanticscholar.org/40b6/8e33d74953b9d9fe1b7cf50368db492c898c.pdf` (last access, July 17, 2019), Oct. 2008

[25] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and detection" in *Submitted to Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019.

[26] L. Mazzon, M. Yasuda, Y. Koizumi, and N. Harada, "Sound event localization and detection using FOA domain spatial augmentation", in *Tech. Report of Detection and Classification of Acoustic Scenes and Events 2019 Challenge (DCASE2019 Challenge)*, Jun. 2019

[27] M. Yasuda, Y. Koizumi, L. Mazzon, S. Saito, and H. Uematsu, "DOA estimation via DNN-based dnoising and dereverberation from sound intensity vector" *arXiv preprint*, 2019.

THE IMPACT OF MISSING LABELS AND OVERLAPPING SOUND EVENTS ON MULTI-LABEL MULTI-INSTANCE LEARNING FOR SOUND EVENT CLASSIFICATION

Maarten Meire[1], *Lode Vuegen*[1], *Peter Karsmakers*[1]

[1] KU Leuven, Dpt. of Computer Science, TC CS-ADVISE,
Kleinhoefstraat 4, B-2440 GEEL, Belgium, maarten.meire@kuleuven.be

ABSTRACT

Automated analysis of complex scenes of everyday sounds might help us navigate within the enormous amount of data and help us make better decisions based on the sounds around us. For this purpose classification models are required that translate raw audio to meaningful event labels. The specific task that this paper targets is that of learning sound event classifier models by a set of example sound segments that contain multiple potentially overlapping sound events and that are labeled with multiple weak sound event class names. This involves a combination of both multi-label and multi-instance learning. This paper investigates two state-of-the-art methodologies that allow this type of learning, low-resolution multi-label non-negative matrix deconvolution (LRM-NMD) and CNN. Besides comparing the accuracy in terms of correct sound event classifications, also the robustness to missing labels and to overlap of the sound events in the sound segments is evaluated. For small training set sizes LRM-NMD clearly outperforms CNN with an accuracy that is 40 to 50% higher. LRM-NMD does only minorly suffer from overlapping sound events during training while CNN suffers a substantial drop in classification accuracy, in the order of 10 to 20%, when sound events have a 100% overlap. Both methods show good robustness to missing labels. No matter how many labels are missing in a single segment (that contains multiple sound events) CNN converges to 97% accuracy when enough training data is available. LRM-NMD on the other hand shows a slight performance drop when the amount of missing labels increases.

Index Terms**—** Multi-label learning, multi-instance learning, weak labels, non-negative matrix deconvolution, convolutional neural networks, overlapping sound events, polyphonic classification, sound event classification.

1. INTRODUCTION

We are surrounded by complex acoustic scenes made up of many potentially overlapping sound events. For example, in a busy street we may hear engine sounds in cars, footsteps of people walking, doors opening, or people talking. Large amounts of recorded sound clips are also being uploaded into audio and multimedia collections of clips on the internet, creating an explosive growth in this audio data. In recent years, research into content analysis for complex audio has found increasing attention in the research community [1]. It has lead to algorithms based on machine learning that automate the analysis of the complex audio of everyday sounds, to help us navigate within the enormous amount of data and help us make better decisions based on the sounds around us.

In this work we specifically focus on the task of learning sound event classifier models by a set of sound segments that contain multiple potentially overlapping sound events and that are labeled with multiple weak sound event class names. Such setup involves a combination of multi-label and multi-instance learning. Multi-label refers to the fact that a single sound segment has multiple labels. When strong labels are used the labelled sound events are all active for the full (small, e.g. 50ms) sound segment. Weak labelled sound events are active on undefined positions in the considered (larger, e.g. 10s) audio segment. Hence, the learning strategy should have the ability to identify multiple instances that are present within an audio fragment. For example learning sound event models based on YouTube movies that have meta information (that could be automatically transformed into some predefined set of class labels) could be considered as a multi-label multi-instance learning task.

The literature concerning the classification of overlapping sounds is mainly divided into two separate streams. Either the overlapping events are separated first using source separation methods [2, 3] or region extraction methods such as [4] prior to detection or the overlapping events are directly classified via a unifying classfication scheme [5, 6], with the latter being the most successful. In this work we compare two methods that belong to this category. Particularly convolutional neural networks (CNN), potentially with some recurrent layers added, have shown good performance with respect to the considered task [5, 7]. In [8] the authors developed a convolutive modeling technique that combines a sound source separation strategy based on non-negative matrix devonvolution (NMD) with weak supervision to enable the option to perform classification. In this paper we will compare both methods not only in terms of classification accuracy but also in terms of robustness to missing labels and the amount of overlap of sound events that are present in the sound segments used during the learning stage.

The literature concerning missing labels in weak labelled data is rather limited. While research into noisy labels has recently been growing [9, 10], the impact of missing labels specifically has not yet been investigated to the best of our knowledge. This is backed up by the statement in [11] that the theme of noisy labels was completely missing from the literature.

This paper is organized as follows. In Section 2 we briefly introduce the two methods that are being compared. Section 3 describes the data set that was used and the experimental setup. The results are discussed in Section 4. Final conclusions and future directions are given in Section 5.

2. METHODS

2.1. Non-negative Matrix Deconvolution

NMD is an extension of non-negative matrix factorization (NMF) and is capable of identifying components with a temporal structure [12, 13]. The main objective of NMD is to decompose an all-positive observation data matrix $\mathbf{Y}^{[o]} \in \mathbb{R}_{\geq 0}^{B \times F}$, e.g. a time-frequency magnitude spectrogram in case of acoustic processing, into the convolution between a set of temporal basis matrices $\mathbf{A}_t^{[o]} \in \mathbb{R}_{\geq 0}^{B \times L}$, with $t \in [1, T]$, and its activation pattern over time $\mathbf{X}_{\geq 0}^{L \times F}$. The general form of NMD is expressed by

$$\mathbf{Y}^{[o]} \approx \mathbf{\Psi}^{[o]} = \sum_{t=1}^{T} \mathbf{A}_t^{[o]} \overset{(t-1)}{\overrightarrow{\mathbf{X}}}, \tag{1}$$

where $\mathbf{\Psi}^{[o]} \in \mathbb{R}_{\geq 0}^{B \times F}$ denotes the reconstructed data and $\overset{t}{\overrightarrow{(\cdot)}}$ a matrix shift of t entries to the right. Columns that are shifted out at the right are discarded, while zeros are shifted in from the left. The complete set of basis data, i.e. also called '*dictionary*' or '*soundbook*' is described by combining all temporal basis matrices $\mathbf{A}_t^{[o]}$ into a global three-way tensor $\mathbf{A}^{[o]} \in \mathbb{R}_{\geq 0}^{B \times L \times T}$. Each l-th slice of $\mathbf{A}^{[o]}$ then contains the temporal basis data of the l^{th}-component over time t and can be interpreted as one of the additive time-frequency elements describing the underlying structure in $\mathbf{Y}^{[o]}$.

The general form of NMD given by Equation 1 factorizes the observations in a blind fashion. In [8] we have proposed an extension to NMD, called low-resolution multi-label non-negative matrix deconvolution (LRM-NMD), where both the observation data and the available labelling information are used during the factorization process. More specifically, let us assume that $\mathbf{Y}^{[o]}$ is supported by a multi-label vector $\mathbf{y}^{[s]} \in \{0, 1\}^C$, with C denoting the number of classes, indicating the sound events that have occurred without describing beginnings nor endings. In the other words weak labels are employed. The objective of LRM-NMD is still to decompose $\mathbf{Y}^{[o]}$ as is given in Equation 1 but with respect to

$$\mathbf{y}^{[s]} \approx \boldsymbol{\psi}^{[s]} = \mathbf{A}^{[s]}\mathbf{X}\mathbf{1}, \tag{2}$$

with $\mathbf{A}^{[s]} \in \{0, 1\}^{C \times L}$ acting as a labelling matrix for $\mathbf{A}^{[o]}$ and $\mathbf{1}$ being an all-one column vector of length F. Hence, the cost function of LRM-NMD is expressed by

$$\min_{\mathbf{A}^{[o]}, \mathbf{A}^{[s]}, \mathbf{X}} \sum_{j=1}^{J} \Big[D(\mathbf{Y}_j^{[o]} \| \mathbf{\Psi}_j^{[o]}) + \lambda \|\mathbf{X}_j\|_1 + \eta D(\mathbf{y}_j^{[s]} \| \boldsymbol{\psi}_j^{[s]}) \Big], \tag{3}$$

where $D(\mathbf{v}\|\mathbf{w})$ denotes the Kullback-Leibler divergence between $\mathbf{v}$ and $\mathbf{w}$, λ being the sparsity penalty parameter and η the trade-off parameter between the observation data and the labelling information. The cost function can be minimised using the method of multiplicative updates as discussed in [8]. LRM-NMD favours decompositions that have a balanced performance in terms of reconstruction error and classification performance. More specifically, LRM-NMD encourages that the sound events in segment $\mathbf{Y}^{[o]}$, labelled by $\mathbf{y}^{[s]}$, are described by a linear combination of a subset of sound book elements in $\mathbf{A}^{[0]}$ each assigned to a specific sound event class by the labelling matrix $\mathbf{A}^{[s]}$. Two crucial advantages of LRM-NMD are: a) that it can deal directly with overlapping sound events in the observation data, i.e. because of the additive behaviour due to the non-negativity constraint, and b) that not all events in an acoustic segment must be labelled and thus that it can cope with missing labels enabling a semi-supervised learning strategy that learns the model parameters from both labelled and unlabelled data.

Classifying an unseen sample is done by decomposing the test data under the fixed learned basis data $\mathbf{A}^{[o]}$ and performing a global average pooling on the corresponding activations $\mathbf{X}$.

2.2. Convolutional Neural Network

Convolutional neural networks (CNNs) have become the current state-of-the-art solution for many different machine learning tasks and are already widely discussed in the literature. CNNs usually consist of several pairs of convolutional and pooling layers as a feature extractor. The extracted features are usually flattened using a flatten layer and are then followed by one or more fully connected layers that act as a classifier.

In this study we used a basic CNN architecture, using the aforementioned layers. To accommodate for variable sized input frames, we changed the flatten layer to a global average pooling layer. This change allows training on segments with a different size compared to those that are being used in the testing phase. While we could train on different length segments, we padded all segments to the length of the longest segment during training for batching purposes. This padding is done, for each mel band, by sampling from a normal distribution with the mean and standard deviation derived from the considered mel band values from the training data.

3. EXPERIMENTAL SETUP

3.1. Dataset

Both methods are validated using the publicly available NAR-dataset. This dataset contains a set of real-life isolated domestic audio events, collected with a humanoid robot Nao, and is recorded specifically for acoustic classification benchmarking in domestic environments [14, 15]. In total 42 different sound classes were recorded and can be categorised into '*kitchen related events*', '*office related events*', '*non-verbal events*' and '*verbal events*'. The verbal events are not used in this research which reduces the dataset to a total of 20 sound classes each containing 20 or 21 recordings.

The training, validation and test sets are created by randomly sampling instances from the NAR-dataset with a ratio of 50% for training, 25% for validation and 25% for testing. The training and validation sets are further processed into so-called acoustic observation segments for the multi-instance multi-label learning task. More specifically, the acoustic segments are generated by randomly drawing five events, sorting them with increasing time duration, and combining them into a single stream with 0%, 25%, 50%, 75% and 100% overlap[1]. In total 10000 training and 2000 validation segments were generated per degree of overlap. The test set was not altered since the envisioned task of the experiments later is simply a classification problem. The previous process was repeated four times resulting in a final dataset containing 4 folds each made up of 10000 training segments, 2000 validation segments, and 100 isolated test samples (5 per sound class) for classification.

3.2. Features

The so-called mel-magnitude spectrograms [16] have shown to be a good choice of acoustic features having the properties of non-negativity and approximate additivity. The mel-magnitude spectrograms that span 40 bands are computed using a Hamming window with a frame length of 25 ms and a frame shift of 10 ms as proposed in [17]. The used filter bank is constructed such that the begin frequency of the first mel-filter and the end frequency of the last mel-

[1]The amount of overlap is defined by the amount of overlapping samples between two successive events, based on the first event. Special case is 100% where all events in the acoustic segment have the same onset time.

filter correspond to the frequency range of the microphone, i.e. 300 Hz and 18 kHz.

3.3. Experiments

In this paper two main experiments were carried out. The first experiment investigates the influence of the number of training segments (n_{tr}) on the classification performance of LRM-NMD and CNN for different degrees of assigned labels (n_{lbl}). The number of training segments are increased offline from 50 to 10,000 and the amount of assigned labels varies between $n_{lbl} = 1$ (4 missing labels per segment) and $n_{lbl} = 5$ (no missing labels). The second experiment investigates the influence of overlapping sound events on the classification performance of LRM-NMD and CNN for a fixed number of training segments. The investigated degrees of overlap (n_{ovl}) vary between $n_{ovl} = 0\%$ and $n_{ovl} = 100\%$. The amount of assigned labels varies again in the range $n_{lbl} = \{1, 2, 3, 4, 5\}$. In both experiments, the objective is to predict a single label for an event while training on multi-label segments.

The set of dictionary elements in $\mathbf{A}^{[o]}$ for LRM-NMD was initialized with one example per sound class and one additional dictionary element with small positive noise for acoustic background modelling. Hence, the overall dimensions of $\mathbf{A}^{[o]}$ are $B = 40$, $L = 21$, and $T = 40$. The labelling matrix $\mathbf{A}^{[s]}$ was initialised by an identity matrix augmented with an all zero C long column vector for the background dictionary element. The used hyperparameters are $\lambda = 5$ and $\eta = 5$ and where selected from the results in [8].

The network used for CNN starts with 3 convolutional layers using (5,5) filter shapes and 64 filters, similar to what was proposed in [5], each convolutional layer is followed by a batchnormalization [18] layer, a relu activation and a pooling layer. The pooling layers used maxpooling with (5,1), (3,1), (2,1) as shapes respectively. After these layers a globalaveragepooling layer is added, followed by a hidden fully connected layer of 64 neurons, with a relu activation, and an output layer of 20 neurons, the same amount as the number of output classes. Between all convolutional layers dropout [19] is used with a drop rate of 0.5. During training the output layer has a sigmoid activation, due to the multi-label nature of this problem, and during inference this activation is changed to a softmax since a single label is required. Since this is a multi-label problem with binarized labels, binary-crossentropy is used as the loss function. Adam was used as optimizer with a learning rate of 0.001.

4. RESULTS

In this section the results of both experiments are discussed. Firstly, we will discuss the effect of missing labels and the amount of training samples on the performance of both methods. Secondly, we will discuss the impact overlap in training segments has in both methods. Finally, while this is not the main focus of this study, we will do a short comparison of our results to the results of other studies which used the NAR-dataset.

4.1. Missing labels

The results of this experiment for CNN and LRM-NMD are presented in Figure 1. The LRM-NMD model was trained with at most $n_{tr} = 2000$, while the CNN was trained up to $n_{tr} = 10000$, however we assume that the results of LRM-NMD will not have a large improvement with a higher n_{tr} based on the trend in the results.

From these results we can see that LRM-NMD substantially outperforms CNN in cases where there is little data available. The latter is mainly the result of the exemplar based initialization of LRM-NMD resulting in a bootstrapped model structure. At $n_{tr} = 50$, CNN achieves accuracies ranging from $53.8 \pm 6.1\%$ to $11.2 \pm 8.2\%$ for 5 labels and 1 label, respectively. In comparison, for the same n_{tr}, LRM-NMD achieves accuracies ranging from $89.0 \pm 3.4\%$ to $66.5 \pm 4.4\%$.

When considering the results when more training segments are added, we can see that CNN begins to achieve similar accuracies as LRM-NMD when a few labels are missing. At $n_{tr} = 1000$, CNN achieves accuracies ranging from $95.5 \pm 1.1\%$ to $64.5 \pm 9.1\%$ for 5 labels and 1 label, respectively. For the same n_{tr} LRM-NMD achieves accuracies ranging from $94.5 \pm 2.5\%$ to $76.0 \pm 2.4\%$. These numbers confirm our statement that CNN achieves similar accuracies as LRM-NMD when few labels are missing, if more training data is added.

If further training data is added, so $n_{tr} = 10000$, we see that the achieved accuracies converge around 97% across all n_{lbl}. From this we can state that CNN sligthly outperforms LRM-NMD, if a large amount of training data is available. Note that compared to CNN, LRM-NMD uses less model parameters and has no non-linear modeling option available.

Another observation is that while LRM-NMD has a good performance for a small n_{tr}, it does not improve as much compared to CNN when n_{tr} increases. Note that the LRM-NMD hyperparameters i.e. η and λ, were selected and kept fixed in a model selection procedure in which all labels were present for each training segment (hence no missing labels). This choice is probably sub-optimal since the amount of provided labels influences the balance between the supervision and reconstruction error terms in 3.

4.2. Overlap of events in segments

In this experiment we used $n_{tr} = 10000$ for CNN and $n_{tr} = 500$ for LRM-NMD. These amounts were chosen based on the classification accuracies and the time needed to train the models. For CNN the accuracies converged for $n_{tr} = 10000$ and for LRM-NMD they started stagnating for $n_{tr} = 500$. The results of the experiment are presented in Figure 2.

For $n_{ovl} = 0\%$ we can see that CNN outperforms LRM-NMD, this can be attributed to the increase in n_{tr}, as described in 4.1. CNN is converged around 97% accuracy, while the accuracies achieved by LRM-NMD range from $93.5 \pm 3.1\%$ to $75.5 \pm 3.9\%$.

At $n_{ovl} = 50\%$ the accuracies achieved by CNN start to diverge slightly, ranging from $96.8 \pm 1.3\%$ to $94.5 \pm 2.7\%$. In comparison, LRM-NMD achieves accuracies ranging from $92.5 \pm 3.7\%$ to $66.3 \pm 1.5\%$. At this point CNN still outperforms LRM-NMD.

However, when we look at $n_{ovl} = 100\%$, we see that CNN has a drop in classification accuracy. The classification accuracy ranges from $80.2 \pm 1.3\%$ to $67.5 \pm 5.7\%$, while the classification accuracy of LRM-NMD ranges from $90.0 \pm 3.2\%$ to $67.3 \pm 7.5\%$.

From these results, we conclude that for up to $n_{ovl} = 75\%$ CNN outperforms LRM-NMD. However, if n_{ovl} gets closer to 100%, LRM-NMD starts to achieve higher accuracies than CNN. A possible explanation for this could be that, due to the nature of the generation of the overlap in the segments, the filters of CNN are smaller than the non-overlapping part of the events for less than 100% overlap. This could lead to the CNN still being able to recognize the events in this non-overlapping area, while the rest of the event is overlapped with the next event.

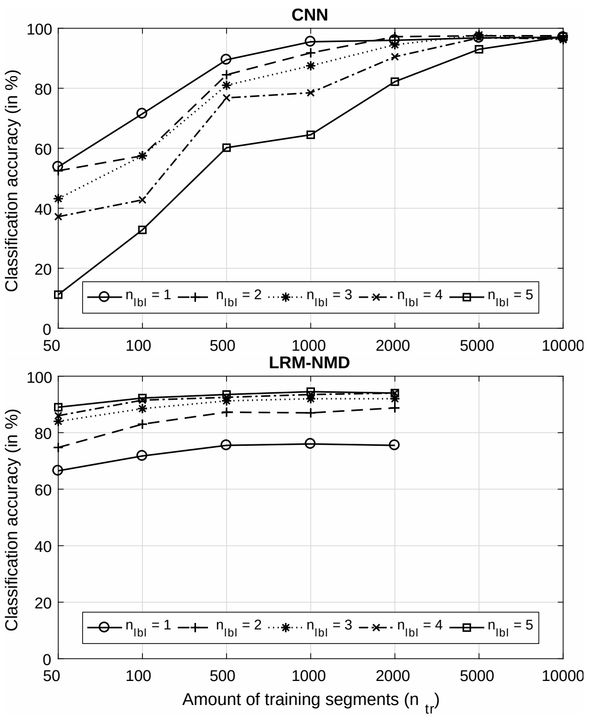


Figure 1: The obtained classification results for CNN and LRM-NMD in function of the number of training segments (n_{tr}) and the number of labelled events per segment (n_{lbl}). Note that $n_{tr} = 5000$ and $n_{tr} = 10000$ are not evaluated for LRM-NMD due to the computational complexity of the multiplicative updates and the stagnation of the results.

An important aspect to note here is that the CNN had more training data. With a smaller training set size, the performance of CNN is worse.

4.3. Comparison with other papers

This paragraph compares the results of this study with other studies using the NAR-dataset. For this comparison we used the best results achieved in the studies, i.e. 96.0% in [14], 97.0% in [15], 98.36% in [20], and 100.0% in [21], and for LRM-NMD and CNN we used the best results with no missing labels and 0% overlap, 94.5% and 97.0% respectively. Note that in the other studies the learning is done using single strong labels, while in this study multi weak labels were used. This makes a direct comparison unfair due to the different natures of learning, however, based on the results, we can cautiously state that we approach state-of-the-art performance.

5. CONCLUSION

In this work two experiments that compare the classification performance of a CNN-based and a LRM-NMD-based approach for acoustic event classification using weakly multi-labelled data were performed.

The first experiment was done to examine the influence of the amount of training data on the classification performance for different amounts of missing labels. In this experiment we observed that for a low amount of data LRM-NMD clearly outperforms CNN

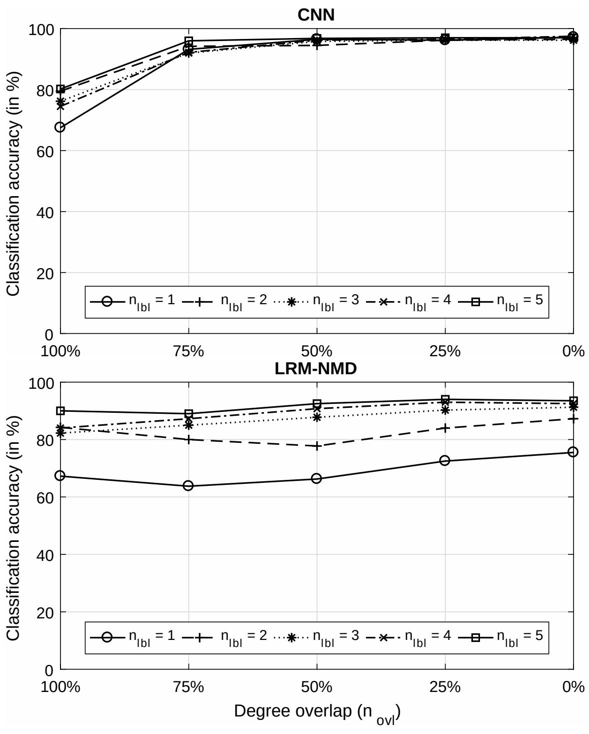


Figure 2: The obtained classification results for CNN and LRM-NMD in function of the degree of overlap (n_{ovl}) and the number of labelled events per segment (n_{lbl}). Note that CNN uses the setting of $n_{tr} = 10000$ and LRM-NMD $n_{tr} = 500$.

with an accuracy that is 40 to 50% higher, on each amount of missing labels. However, if enough training data is added, CNN slightly outperforms LRM-NMD and converges to 97% accuracy for all amounts of missing labels. Results from this experiment indicate that, with large training set sizes and with a uniform probability of a label being absent over classes, missing labels have a very limited effect on the classification performance of a CNN.

In the second experiment we examined the impact of overlap on the classification performance. This experiment was done using $n_{tr} = 500$ and $n_{tr} = 10000$ for LRM-NMD and CNN respectively which gave the best models in the former experiment for 0% overlap of both approaches. We conclude that for up to 75% overlap CNN outperforms LRM-NMD and converges to 97% while LRM-NMD reaches 95% accuracy. However, if the amount of overlap increases further, LRM-NMD starts to outperform CNN, with up to 10% higher accuracies for different amounts of missing labels. In this experiment we have also seen that overlap has a relatively limited impact on LRM-NMD.

In future work we also target to develop a neural network alternative to the LRM-NMD algorithm that we proposed in [8]. In this way we can benefit from the modelling flexibility (e.g. the ability to include non-linearity in the modelling process) that comes with neural networks allowing for several extensions and generalizations, while also keeping the capabilities of LRM-NMD (e.g. being able to use unlabelled data in addition to weak labelled data and the robustness to overlap). Moreover, a more detailed benchmarking of the considered methods will be performed on other publicly available data sets.

6. REFERENCES

[1] “DCASE: Detection and classification of acoustic scenes and events,” http://dcase.community/.

[2] T. Heittola, A. Mesaros, T. Virtanen, and M. Gabbouj, “Supervised model training for overlapping sound events based on unsupervised source separation,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, May 2013, pp. 8677–8681.

[3] A. Mesaros, T. Heittola, O. Dikmen, and T. Virtanen, “Sound event detection in real life recordings using coupled matrix factorization of spectral representations and class activity annotations,” in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 151–155.

[4] I. McLoughlin, H. Zhang, Z. Xie, Y. Song, and W. Xiao, “Robust sound event classification using deep neural networks,” *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 23, pp. 540–552, 2015.

[5] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, “Convolutional recurrent neural networks for polyphonic sound event detection,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, June 2017.

[6] H. Phan, O. Y. Chén, P. Koch, L. Pham, I. McLoughlin, A. Mertins, and M. D. Vos, “Unifying isolated and overlapping audio event detection with multi-label multi-task convolutional recurrent neural networks,” in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 51–55.

[7] C.-C. Kao, W. Wang, M. Sun, and C. Wang, “R-CRNN: Region-based convolutional recurrent neural network for audio event detection,” in *Proc. Interspeech 2018*, 2018, pp. 1358–1362. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-2323

[8] L. Vuegen, P. Karsmakers, B. Vanrumste, and H. Van hamme, “Acoustic event classification using low-resolution multi-label non-negative matrix deconvolution,” *Audio Engineering Society (AES)*, vol. 66, pp. 369–384, 5 2018.

[9] E. Fonseca, M. Plakal, D. P. W. Ellis, F. Font, X. Favory, and X. Serra, “Learning sound event classifiers from web audio with noisy labels,” in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 21–25.

[10] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, and X. Serra, “Audio tagging with noisy labels and minimal supervision,” *arXiv e-prints*, p. arXiv:1906.02975, Jun 2019.

[11] A. Shah, A. Kumar, A. G. Hauptmann, and B. Raj, “A Closer Look at Weak Label Learning for Audio Events,” *arXiv e-prints*, p. arXiv:1804.09288, Apr 2018.

[12] D. Lee and H. Sebastian Seung, “Learning the parts of objects by non-negative matrix factorization,” *Nature*, vol. 401, pp. 788–91, 11 1999.

[13] P. Smaragdis, “Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs,” in *Independent Component Analysis and Blind Signal Separation*, C. G. Puntonet and A. Prieto, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 494–499.

[14] M. Janvier, X. Alameda-Pineda, L. Girinz, and R. Horaud, “Sound-event recognition with a companion humanoid,” in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, Nov 2012, pp. 104–111.

[15] J. Maxime, X. Alameda-Pineda, L. Girin, and R. Horaud, “Sound representation and classification benchmark for domestic robots,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 6285–6292.

[16] J. F. Gemmeke, L. Vuegen, P. Karsmakers, B. Vanrumste, and H. Van hamme, “An exemplar-based NMF approach to audio event detection,” in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, Oct 2013, pp. 1–4.

[17] B. Gold, N. Morgan, and D. Ellis, *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*, 2nd ed. New York, NY, USA: Wiley-Interscience, 2011.

[18] S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” *arXiv e-prints*, p. arXiv:1502.03167, Feb 2015.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[20] H. Phan, L. Hertel, M. Maass, R. Mazur, and A. Mertins, “Learning representations for nonspeech audio events through their similarities to speech patterns,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 4, pp. 807–822, April 2016.

[21] P. M. Baggenstoss, “Acoustic event classification using multi-resolution HMM,” in *2018 26th European Signal Processing Conference (EUSIPCO)*, Sep. 2018, pp. 972–976.

ACOUSTIC SCENE CLASSIFICATION IN DCASE 2019 CHALLENGE: CLOSED AND OPEN SET CLASSIFICATION AND DATA MISMATCH SETUPS

Annamaria Mesaros, Toni Heittola, Tuomas Virtanen

Tampere University
Computing Sciences
Korkeakoulunkatu 10, Tampere, Finland
name.surname@tuni.fi

ABSTRACT

Acoustic Scene Classification is a regular task in the DCASE Challenge, with each edition having it as a task. Throughout the years, modifications to the task have included mostly changing the dataset and increasing its size, but recently also more realistic setups have been introduced. In DCASE 2019 Challenge, the Acoustic Scene Classification task includes three subtasks: Subtask A, a closed-set typical supervised classification where all data is recorded with the same device; Subtask B, a closed-set classification setup with mismatched recording devices between training and evaluation data, and Subtask C, an open-set classification setup in which evaluation data could contain acoustic scenes not encountered in the training. In all subtasks, the provided baseline system was significantly outperformed, with top performance being 85.2% for Subtask A, 75.5% for Subtask B, and 67.4% for Subtask C. This paper presents the outcome of DCASE 2019 Challenge Task 1 in terms of submitted systems performance and analysis.

Index Terms— Acoustic Scene Classification, DCASE 2019 Challenge, open set classification

1. INTRODUCTION

Acoustic scene classification is a task of widespread interest in the general topic of environmental audio analysis, and refers to the specific case of classifying environments based on their general acoustic characteristics [1, 2, 3]. Other closely related and popular directions of research include classification of individual sound events from the environment, sound event detection, localization and tagging. Specific applications for acoustic scene classification include services and devices that can benefit of context awareness [4], services or applications for indexing audio content [5], documentary and archival of everyday experience [6], wearable technology, navigation systems for robotics, etc.

As a research area acoustic scene classification is not novel, but has gained traction in recent years due to the wide availability of user devices and applications. However, is not plausible to be able to record training data with all devices or all types of scenes that may be encountered in use conditions. In such situation, the classifiers require methods to handle device mismatch through e.g. domain adaptation, and the ability to detect acoustic scenes unseen in training.

This work has received funding from the European Research Council, grant agreement 637422 EVERYSOUND

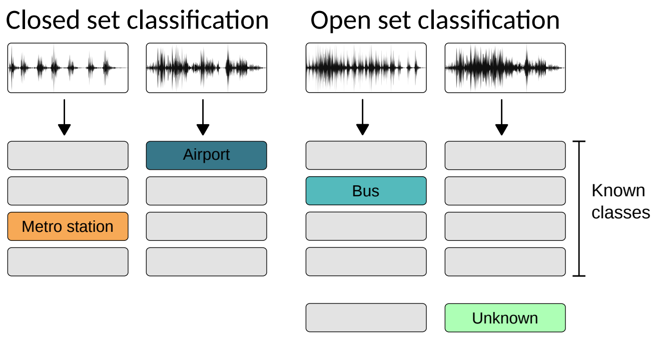


Figure 1: Closed and open-set acoustic scene classification

In DCASE 2019 Challenge, the Acoustic Scene Classification Task includes three subtasks, among which two represent realistic usage cases. Subtask A is a closed-set supervised classification problem where all data is recorded with the same device; Subtask B is a closed-set classification problem with mismatched recording devices between training and evaluation data, and Subtask C is an open-set classification problem in which evaluation data could contain acoustic scenes not encountered in the training.

In this paper we present the task setup and submissions of DCASE 2019 Challenge Task 1. We introduce the three different setups used for the three subtasks, describe the datasets provided for each, and present the challenge submissions. Evaluation and analysis of submitted systems includes general statistics on systems and performance and system characteristics.

The paper is organized as follows: Section 2 presents the task setup including data, rules and baseline system, Section 3 presents the main statistics about the received submissions, while Section 4 presents an analysis of the main trends in the submissions, with details about selected systems. Finally, Section 6 presents the conclusions and ideas for future editions.

2. TASK DESCRIPTION

The goal of acoustic scene classification is to classify a test recording into one of the provided predefined classes that characterizes the environment in which it was recorded. In DCASE 2019 challenge, the Acoustic Scene Classification task presented participants with three different subtasks that required system development for three different situations:

- Subtask A: Acoustic Scene Classification. Classification of data from the same device as the available training data.

https://doi.org/10.33682/m5kp-fa97

- Subtask B: Acoustic Scene Classification with mismatched recording devices. Classification of data recorded with devices different than the training data.
- Subtask C: Open set Acoustic Scene Classification. Classification on data that includes classes not encountered in the training data.

2.1. Dataset

The dataset for this task is the TAU Urban Acoustic Scenes 2019 dataset, consisting of recordings from the following acoustic scenes: airport, indoor shopping mall, metro station, pedestrian street, public square, street with medium level of traffic, travelling by tram, travelling by bus, travelling by underground metro, and urban park. The dataset used for the task is an extension of the TUT 2018 Urban Acoustic Scenes dataset, recorded in multiple cities in Europe. TUT 2018 Urban Acoustic Scenes dataset contains recordings from Barcelona, Helsinki, London, Paris, Stockholm and Vienna, to which TAU 2019 Urban Acoustic Scenes dataset adds Lisbon, Amsterdam, Lyon, Madrid, Milan, and Prague. The recordings were done with four devices simultaneously, denoted in the data as device A (Soundman OKM II Klassik/studio A3 electret binaural microphone), device B (Samsung Galaxy S7), device C (IPhone SE), and device D (GoPro Hero5 Session). The data recording procedure is explained in detail in [13].

Different versions of the dataset are provided for each subtask, together with a training/test partitioning for system development. Generalization properties of systems were tested by presenting in the evaluation set data recorded in cities unseen in training (10 cities in development data, 12 in evaluation). As a special situation, in Subtask C additional data is provided for the open set classification; this consists of the ”beach” and ”office” classes of TUT Acoustic Scenes 2017 dataset, and other material recorded in 2019. Similarly, data from acoustic scenes other than the 10 mentioned above were present in the evaluation data.

Table 1 summarizes the information about datasets. For each subtask, the development set is split into training/test subsets, created based on the recording location such that the training subset contains approximately 70% of recording locations from each city. The evaluation set was released as audio only, two weeks before the challenge submission deadline; reference annotation is available only to task coordinators for evaluating the systems' performance.

Use of external data was allowed in all subtasks under the conditions that the data is freely accessible and available before the release of the Evaluation dataset. A list of external data sources was provided, and participants had the option to suggest others.

2.2. Evaluation

The submissions were evaluated using classification accuracy calculated as average of the class-wise accuracy, with each segment considered as an independent test sample. Ranking of submissions was done as follows:

- Subtask A used the average accuracy on all evaluation data.
- Subtask B used the average accuracy on devices B and C.
- Subtask C used the weighted average of the accuracy of known classes ACC_{kn} and accuracy of the unknown class ACC_{unk}, with a weight of 0.5 for each:

$$ACC_w = wACC_{kn} + (1 - w)ACC_{unk} \qquad (1)$$

During the challenge, public leaderboards were provided for each task through Kaggle InClass competitions. Leaderboards were meant to serve as a development tool for participants, and did not have an official role in the challenge.

2.3. Baseline system

The baseline system implements a convolutional neural network (CNN) based approach using two CNN layers and one fully connected layer, trained using log mel-band energies extracted for the 10-second audio examples. The system is identical to the baseline provided in Task 1 of DCASE 2018 Challenge, and detailed system parameters can be found in [13]. Model selection is done using a validation set of approximately 30% of the original training data. Model performance is evaluated on this validation set after each epoch, and the best performing model is selected.

Specific modifications for subtasks include the use of different training data for the different subtasks and the decision making process for the output. Training of the system for Subtask B was done such that all available audio material (devices A, B and C) was used, with no specific way of treating parallel data. For Subtask C, the system was trained using only the known classes audio material.

The activation function in the output layer for Subtasks A and B is softmax, allowing selection of the most likely class in the closed-set classification problem. For Subtask C, the activation function in the output layer is sigmoid, to allow making the open-set decision based on a threshold; if at least one of the class values is over the threshold of 0.5, the most probable target scene class is chosen, if all values are under the threshold, the unknown scene class is selected.

3. CHALLENGE SUBMISSIONS

The task has received a total number of 146 submissions from 46 teams (maximum 4 submissions per team allowed). Subtask A was the most popular, as expected, with 98 submissions; Subtask B has received 29 submisions, and Subtask C 19.

Subtask A had the best performance of 85.2%, with a 95% confidence interval (CI) between 84.4 and 86.0. Zhang et al. [14] are authors of the four best systems. Koutini et al. [15] ranked 5th - 8th, with their best system having an accuracy of 83.8% (CI 82.9 - 84.6). The McNemar test between the top system of Zhang et al. and Koutini et al. shows that they are significantly different, establishing Zhang et al. as the top system. The baseline system with a performance of 63.3% ranks very low, with only 5 of the 98 submitted systems performing lower.

In Subtask B, Kosmider et al. [16] obtained the highest performance of 75.3% (74.3 - 76.3) on data from devices B and C, and submitted the four best systems. Tied on 4th rank, the system by McDonnell et al. [17] obtained an accuracy of 74.9% (73.9 - 75.9), while on rank 5, the system by Eghbal-zadeh et al. [15] has an accuracy of 74.5% (73.5 - 75.5). McNemar's test shows that the top system by Kosmider et.al and the system by McDonnell et al. make significantly different errors; also McDonnell et al. and Eghbal-zadeh et al. are significantly different according to the same test, therefore even though their confidence intervals overlap, their order in ranking is justified. The baseline system ranks last with a significant gap to the second last, as no effort was made in it to deal with the device mismatch.

The top system in Subtask C, by Zhu et al. [18], has an accuracy of 67.4% (66.8 - 68.1) calculated according to (1), and again the four best systems were submitted by the same team. On rank 5 is

	Subset	Hours	Devices	Observations
Subtask A dataset:	Dev [7]	40	A	Binaural audio, data balanced between classes
TAU Urban Acoustic Scenes 2019	Eval [8]	20	A	Introduced two unseen cities
Subtask B dataset:	Dev [9]	46	A, B, C	Single channel audio, 3h of parallel data provided
TAU Urban Acoustic Scenes 2019 Mobile	Eval [10]	30	A, B, C, D	Introduced two unseen cities, unseen device D
Subtask C dataset:	Dev [11]	44	A	Single channel audio, 4h ”unknown” class data
TAU Urban Acoustic Scenes 2019 Open set	Eval [12]	20	A	Introduced two unseen cities, unknown class data

Table 1: Summary of datasets. Complete details for each are included with the data package.

a system by Rakowski et al. [19] with a performance of 64.4% (CI 63.8 - 65.1); this is outside of the confidence interval of the top system, therefore the top system performs significantly better. In this subtask too, the baseline system ranks last.

Figure 2 presents the performance of the top ten teams for Subtasks A and B, and top 5 for Subtask C. Best system per team is selected for the illustration. In the bottom panel, additional details on the system performance are presented: seen vs unseen cities in Subtask A, other devices including unseen device D in Subtask B, and the known vs unknown scenes in Subtask C.

4. ANALYSIS OF SUBMISSIONS

A large majority of submissions for all subtasks used as features log mel energies and used classifiers based on convolutional neural networks. The following statistics are based on the information reported by participants.

4.1. Acoustic Scene Classification

Subtask A includes 85 systems of the 99 (including baseline) that use log mel energies, as standalone features or in combination with other features; the other most common preprocessing technique was harmonic/percussive source separation [20] used by 8 systems of 3 teams. From the 99 systems, 58 reported using mixup [21] as a data augmentation method.

CNNs were part of 82 systems, in many cases as ensemble. Ensembles were very common, with 75 systems reporting use 2 to 40 subsystems. Many ensembles are just multiple CNNs, while in some cases combinations of specific architectures like VGG, Inception and ResNet were used. The most number of systems in an ensemble is 40, with one billion parameters [22]. The system uses a model pre-trained on AudioSet [23] and obtains an accuracy of 80.5%, ranking 15th among the 99 systems.

The top system by Zhang et al. used log mel energies and CQT as feature representations, generative neural network-based augmentation, and an ensemble of 7 CNNs having in total 48M parameters [14]. Among the 7 subsystems, one uses an adversary city adaptation branch that classifies the test samples into the target city, and a gradient reverse layer that makes the output of convolutional layers similar for the same scene class over various city domains. The system has a performance of 77.9% on data from unknown cities, compared to 86.7% on the cities enocuntered in training.

The runner-up team proposed a variety of Receptive-Field-Regularized CNN classifiers, among which one submission was single-model. Separate predictions of the model (snapshots) taken every 5 epochs after 300 epochs in training were used as a way to incorporate more opinions on the test data,resulting in a system with 35M parameters. The best system of the team uses among others a new convolutional layer to create Frequency-Aware Convolutional Neural Networks [15], with filters more specialized in certain frequencies. This ensemble of 7 subsystems has 71M parameters, and its confidence intervals overlap with the single model system.

4.2. Acoustic Scene Classification with mismatched devices

Subtask B has a total of 30 systems including the baseline, of which 29 are CNN-based, the other one using support vector machines. Among all, 25 systems use log mel energies, four use perceptually weighted power spectrogram (one team), and one uses mel-frequency discrete wavelet coefficients; 20 systems use mixup, and 14 have a parameter count over 10M. The most number of systems in an ensemble is 124, belonging to the top system, while the highest parameter count is 727M for an ensemble of 11 systems using 20 snapshots each [15]. Methods for dealing with the device mismatch include domain adaptation and transfer learning, feature transform, spectrum correction, and regularization.

The top systems from Kosmider et al. [16] are based on large ensembles and use a spectrum correction method to account for different frequency responses of the devices in the dataset. The method uses the special feature of the provided development data, namely the temporally aligned recordings from different devices and calculated correction coefficients for devices, using as a reference the average spectrum of devices B and C. The method obtains an accuracy of 75.3 on the data from devices B and C (ranking metric), 80.8% on device A, and 38.6% on the unseen device D.

Also in the top is a simple two-system CNN ensemble by McDonnell et al. that uses multiple forms of regularization that involves aggressively large value for weight decay and not learning batch normalization scale and offset, along with mixup and temporal crop augmentation [17]. The two CNNs use deep residual networks with two pathways, one for high frequencies and one for low frequencies, that were fused prior to the network output. Authors point out that the temporal and frequency axes in spectrograms represent fundamentally different information than for images, and choose no to downsample the frequency axis within the networks. No specific processing of the parallel data from different devices is reported, but the system obtains nevertheless a very balanced performance of the four different devices: 74.9% (73.9 - 75.9) on devices B and C, performs with 79.8% on device A, and 65.2% on device D, which is the highest accuracy obtained on device D among all systems.

4.3. Open set Acoustic Scene Classification

Subtask C has a total of 20 entries, all using log mel energies, with all but the winner team using CNNs. Only six systems have better accuracy for the known classes than the unknown class, indicating a tendency towards optimization for detection of the unknown class.

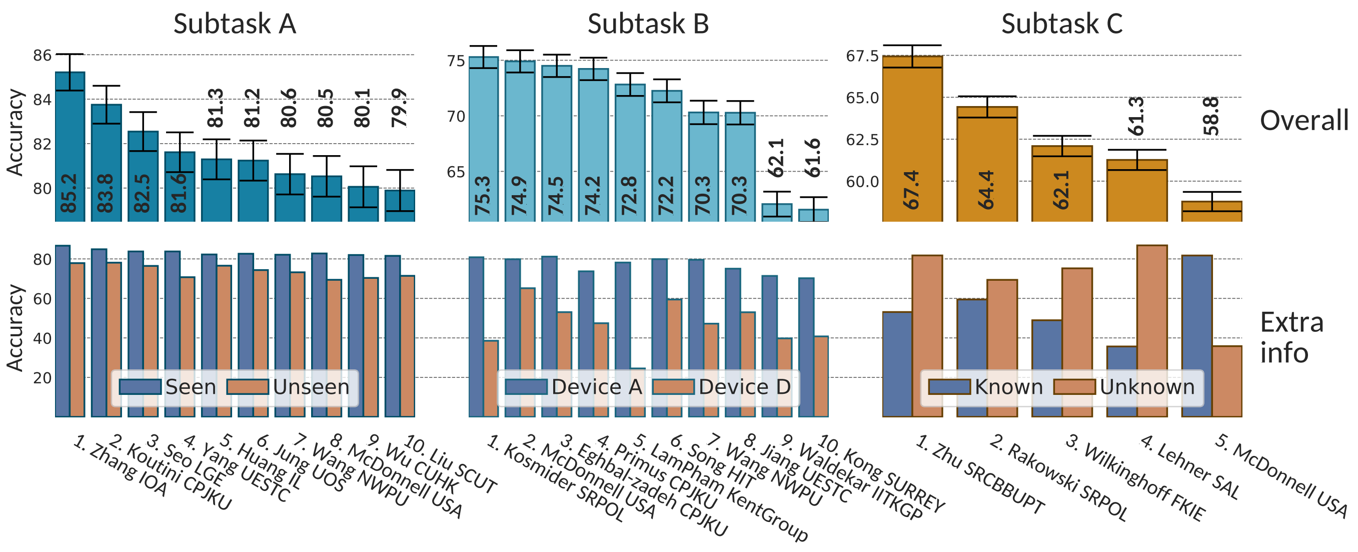


Figure 2: Performance of top teams in each subtask, including confidence intervals

The highest reported number of subsystems in an ensemble for this subtask is 17, but most have only 2 to 4 subsystems.

The top ranked system by Zhu et al. uses CRNNs with self-attention mechanism which are trained on different time divisions of the mel spectrogram. The decision for the unknown class is guided by a threshold of 0.4 on the output layer probabilities for the classes [18]. Their choice of threshold results in a 81.8% accuracy on the unknown class, with 53.1% on the known 10 classes.

Rakowski et al. [19] employed a frequency-aware CNN that preserves the location of features on the frequency axis by applying global pooling only across the temporal dimension, similar to the observations of [17] in Subtask B. The approach resulted in a relatively balanced performance on the known and unknown classes, 59.5% and 69.4% respectively. Lehner et al. [24] used a rejection option for the identification of unknown class, based on the most likely of the ten known classes. They note that the weighted average accuracy used for ranking Subtask C favors aggressive rejection, and for this reason chose the threshold for rejection as the maximum score on the validation data. As a result, they obtained an accuracy of up to 91% on the unknown class, but considerably lower performance on the ten known classes, only 30%.

A notably different approach was proposed by Wilkinghoff et al. [25], which treats the open set classification problem as a combination of convolutional neural networks for closed-set classification and deep convolutional autoencoders for unknown class detection. The method results in a high accuracy on the unknown class at the expense of low accuracy in the closed set (75.2% vs 48.9%).

5. DISCUSSION

One immediate observation about the submissions is that there was little use of external data, with only the four mentioned systems of one team using pretrained models [22]. This is contrary to the feedback of previous challenges that indicated participants wanted to use external data. It is possible that the datasets provided for the task are considered large enough to warrant robust modeling, and therefore use of external data is not necessary.

Compared to 2018 Challenge, novel approaches tailored to use of parallel data have emerged for solving the device mismatch. Among all, the spectrum correction has provided the best performance on the target devices [16], but the best generalization over the four devices was obtained by extensive regularization procedures [17]. The open set classification was tackled by participants in few different ways, with most systems using a threshold. The more distinct approaches treated the unknown class as a separate class [17] or as a subproblem [25]. In most cases, the optimization resulted in emphasis on getting good performance on the unknown class, at the expense of the performance on the ten known classes.

We also want to highlight two approaches to cross-task solutions, one very basic and another one including many techniques to achieve robustness. [26] consists of a generic CNN architecture, similar to the baseline but with more layers, and obtains average performance in Subtask A (53th with 70.5%) but is only better than the baseline system in Subtask B and Subtask C. In contrast, [17] uses more specialized networks and extensive regularization and data augmentation techniques, resulting in a system highly robust to device mismatch (4th in subtask B), having a performance 10 to 15% higher than [26] in all subtasks. These results show that a generic approach, while generally appropriate for straightforward tasks such as the closed set classification, is not suitable for dealing with the same problem in realistic settings, but requires additional techniques to obtain satisfactory performance.

6. CONCLUSIONS

The 2019 Challenge has introduced realistic problems that included evaluation data that contained unseen cities in Subtask A, unseen devices in Subtask B and unseen acoustic scene classes in Subtask C. Acoustic Scene Classification remains the favorite task in the DCASE Challenge, as it offers a textbook problem for audio classification, suitable for beginners in the field. With multiple subtasks of different complexity, the task has also attracted the attention of experienced researchers, and state of the art methods for audio classification are continuously developed within the framework of acoustic scene classification.

7. REFERENCES

[1] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley, "Detection and classification of acoustic scenes and events," *IEEE Transactions on Multimedia*, vol. 17, no. 10, pp. 1733–1746, Oct 2015.

[2] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 16–34, May 2015.

[3] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and Classification of Acoustic Scenes and Events: Outcome of the DCASE 2016 Challenge," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 2, pp. 379–393, Feb 2018.

[4] B. N. Schilit, N. Adams, and R. Want, "Context-aware computing applications," in *IN PROCEEDINGS OF THE WORKSHOP ON MOBILE COMPUTING SYSTEMS AND APPLICATIONS*. IEEE Computer Society, 1994, pp. 85–90.

[5] Y. F. Phillips, M. Towsey, and P. Roe, "Revealing the ecological content of long-duration audio-recordings of the environment through clustering and visualisation," *PLOS ONE*, vol. 13, no. 3, pp. 1–27, 03 2018. [Online]. Available: https://doi.org/10.1371/journal.pone.0193345

[6] M. Droumeva, "Curating everyday life: Approaches to documenting everyday soundscapes," *M/C Journal*, vol. 18, no. 4, 2015.

[7] T. Heittola, A. Mesaros, and T. Virtanen, "TAU Urban Acoustic Scenes 2019, Development dataset," Mar. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2589280

[8] ——, "Tau urban acoustic scenes 2019, evaluation dataset," June 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3063822

[9] ——, "TAU Urban Acoustic Scenes 2019 Mobile, Development dataset," March 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2589332

[10] ——, "TAU Urban Acoustic Scenes 2019 Mobile, Evaluation dataset," June 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3063980

[11] ——, "TAU Urban Acoustic Scenes 2019 Openset, Development dataset," March 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2591503

[12] ——, "TAU Urban Acoustic Scenes 2019 Openset, Evaluation dataset," May 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3064132

[13] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 9–13.

[14] H. Chen, Z. Liu, Z. Liu, P. Zhang, and Y. Yan, "Integrating the data augmentation scheme with various classifiers for acoustic scene modeling," DCASE2019 Challenge, Tech. Rep., June 2019.

[15] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Acoustic scene classification and audio tagging with receptive-field-regularized CNNs," DCASE2019 Challenge, Tech. Rep., June 2019.

[16] M. Kośmider, "Calibrating neural networks for secondary recording devices," DCASE2019 Challenge, Tech. Rep., June 2019.

[17] W. Gao and M. McDonnell, "Acoustic scene classification using deep residual networks with late fusion of separated high and low frequency paths," DCASE2019 Challenge, Tech. Rep., June 2019.

[18] H. Zhu, C. Ren, J. Wang, S. Li, L. Wang, and L. Yang, "DCASE 2019 challenge task1 technical report," DCASE2019 Challenge, Tech. Rep., June 2019.

[19] A. Rakowski and M. Kośmider, "Frequency-aware CNN for open set acoustic scene classification," DCASE2019 Challenge, Tech. Rep., June 2019.

[20] N. Ono, K. Miyamoto, J. Le Roux, H. Kameoka, and S. Sagayama, "Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram," in *2008 16th European Signal Processing Conference*, Aug 2008, pp. 1–4.

[21] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

[22] J. Huang, P. Lopez Meyer, H. Lu, H. Cordourier Maruri, and J. Del Hoyo, "Acoustic scene classification using deep learning-based ensemble averaging," DCASE2019 Challenge, Tech. Rep., June 2019.

[23] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 776–780.

[24] B. Lehner and K. Koutini, "Acoustic scene classification with reject option based on resnets," DCASE2019 Challenge, Tech. Rep., June 2019.

[25] K. Wilkinghoff and F. Kurth, "Open-set acoustic scene classification with deep convolutional autoencoders," DCASE2019 Challenge, Tech. Rep., June 2019.

[26] Q. Kong, Y. Cao, T. Iqbal, W. Wang, and M. D. Plumbley, "Cross-task learning for audio tagging, sound event detection and spatial localization: DCASE 2019 baseline systems," DCASE2019 Challenge, Tech. Rep., June 2019.

OTOMECHANIC: AUDITORY AUTOMOBILE DIAGNOSTICS VIA QUERY-BY-EXAMPLE

Max Morrison

Northwestern University
Computer Science Deptartment
Evanston, IL, USA
morrimax@u.northwestern.edu

Bryan Pardo

Northwestern University
Computer Science Department
Evanston, IL, USA
pardo@northwestern.edu

ABSTRACT

Early detection and repair of failing components in automobiles reduces the risk of vehicle failure in life-threatening situations. Many automobile components in need of repair produce characteristic sounds. For example, loose drive belts emit a high-pitched squeaking sound, and bad starter motors have a characteristic whirring or clicking noise. Often drivers can tell that the sound of their car is not normal, but may not be able to identify the cause. To mitigate this knowledge gap, we have developed OtoMechanic, a web application to detect and diagnose vehicle component issues from their corresponding sounds. It compares a user's recording of a problematic sound to a database of annotated sounds caused by failing automobile components. OtoMechanic returns the most similar sounds, and provides weblinks for more information on the diagnosis associated with each sound, along with an estimate of the similarity of each retrieved sound. In user studies, we find that OtoMechanic significantly increases diagnostic accuracy relative to a baseline accuracy of consumer performance.

Index Terms— Audio retrieval, human-computer interfaces (HCI), public safety, transfer learning, vehicle diagnosis

1. INTRODUCTION

The timely maintenance of personal automobiles is vitally important to passenger safety. Foregoing important vehicle maintenance can cause a vehicle to behave unexpectedly and poses a danger to both occupants and nearby pedestrians. Failing to fix specific vehicle issues in a timely manner may also result in significantly more expensive repairs (e.g., engine damage due to a lack of oil). Because it is the consumer's decision to take their vehicle in for repair, it is of significant public interest to empower drivers with knowledge regarding the status of their vehicle, and whether any urgent repairs are needed.

The owner is typically alerted to a vehicle issue by either notifications from on-board computers or a change in the sensory experience of driving (e.g., a strange sound, smell, or vibration). On-Board Diagnostic (OBD) computer systems have been ubiquitous in consumer vehicles sold in the United States since 1996. While OBD systems provide some basic information about vehicle status directly to the driver (e.g., via the "Check Engine" light), a large majority of the diagnostic information from these systems must be retrieved via an external, specialized computer most consumers do not own.

Many automobile components in need of repair produce characteristic sounds. For example, loose drive belts emit a high-pitched squeaking sound, and bad starter motors have a characteristic whirring or clicking noise. Often drivers can tell that the car does not sound normal, but may not be able to identify the failing component. Consumer guides have been released to help drivers identify these sounds [1]. However, these guides assume that the consumer possesses significant knowledge, such as being able to locate and identify the vehicle components for power-steering or engine cooling. Many do not have this knowledge.

In this work, we present the OtoMechanic ("oto-" meaning "ear") web application. OtoMechanic is designed for drivers who can hear a strange sound coming from their vehicle, but may be uncertain of the underlying issue and want information about it, such as the urgency and cost to repair. To use OtoMechanic, one uploads a recording of the sound a car is making and answers questions about when and where the sound happens. The system diagnoses the problem by measuring the similarity of the uploaded recording to reference recordings in a database of sounds produced by a variety of known vehicle issues. In the remainder of this article we describe related work, the methods used in OtoMechanic, the collection of labeled recording of automotive problems and a user study to evaluate the effectiveness of OtoMechanic.

2. RELATED WORK

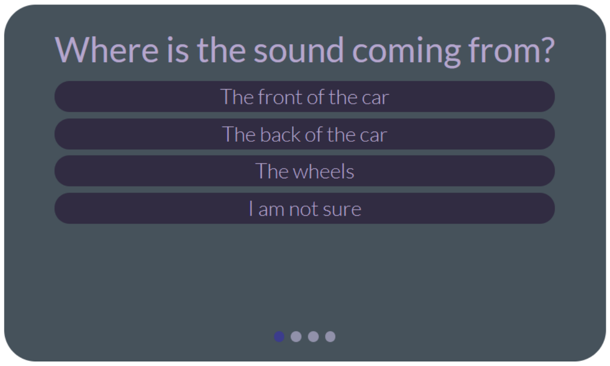
By far, the most reliable sources of diagnostic information are domain experts such as professional auto mechanics. However, the costs associated with visiting a mechanic cause consumers to hesitate and instead ask "Do I need to go to a mechanic?" Indeed, our user studies suggest that consumers often expend a significant amount of time in performing preliminary diagnostics before deciding to consult a mechanic. Since many drivers cannot identify specific causes or issues themselves, there have been efforts to build software to perform diagnosis of automotive issues

Recent work by Siegel et al. [2] demonstrates that convolutional neural networks achieve a 78.5% accuracy rate for visually diagnosing photos of damaged and unsafe tires—a significant improvement over the 55% accuracy achieved by humans not trained to detect flaws in tires. This shows the potential for systems to diagnose vehicle problems with an accuracy that exceeds non-expert performance. While Siegel et al. diagnose photos of tires (i.e., visual information), our system diagnoses the sounds produced by the vehicle.

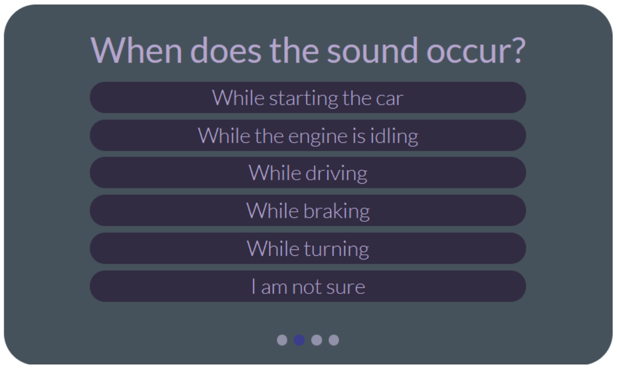
A strange sound is a useful indicator for a variety of specific vehicle issues (e.g., worn drive belts and low battery). Unfortunately, many people cannot name which component is failing from

This work was funded, in part, by USA National Science Foundation Award 1617497.

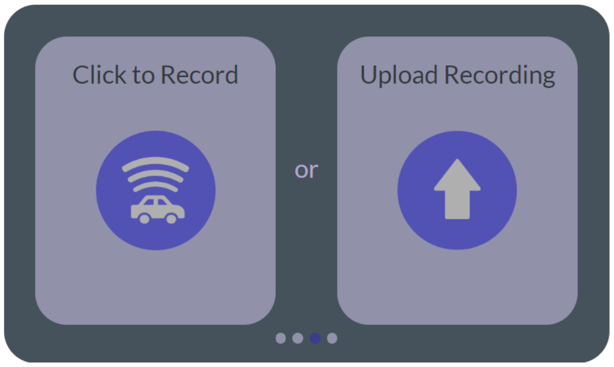
https://doi.org/10.33682/ne8s-1m78



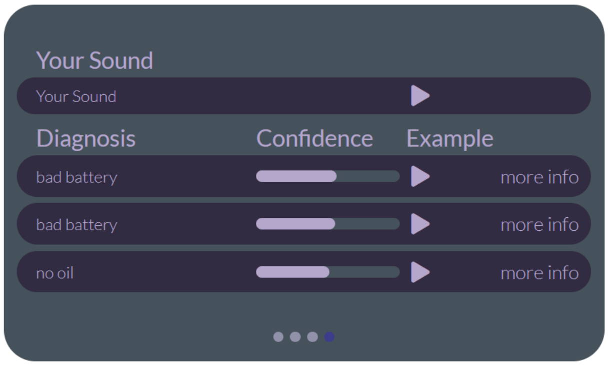
(a) Specifying location information



(b) Specifying timing information



(c) Audio upload interface



(d) Diagnostic results interface

Figure 1: The OtoMechanic user interface

the sound. Datasets of annotated car sounds offer one resource for consumers trying to identify a component making a strange sound. Existing datasets include the Car Noise Emporium [3] consisting of vocal imitations of 45 vehicle issues, the ClingClanger mobile application consisting of 27 actual recordings of vehicle issues taken from a single vehicle [4], and YouTube. YouTube videos demonstrating the sounds of common car issues have millions of views [5, 6]. However, finding a video to diagnose a specific issue requires consumers to identify by name the failing component in order to construct a useful search query.

In all of these approaches (including ClingClanger), the task of matching the sound emitted by their vehicle to example sounds with known causes falls to the user, who is required to listen to all of the available recordings to find the best match. The amount of audio that must be listened to increases linearly as the number of possible diagnostic sounds increases, imposing a significant time cost.

Auditory analysis is regularly used by professional test engineers and mechanics when diagnosing vehicle issues, and a large body of literature exists on this subject, specifically for engine diagnostics [7, 8, 9, 10, 11, 12, 13]. The work most similar to ours is by Nave and Sybingco [11], who perform a classification task on the sounds of three engine issues, and consider the high variance of sounds caused by the same diagnosis across different vehicles. Nave and Sybingco developed an application for the Android operating system to perform classification of these three engine sounds. However, this app is only useful when the issue has already been narrowed to an engine fault, which is itself a non-trivial diagnostic task. More generally, all of these works focus on the development of tools for professional use, and do not address use by non-expert consumers.

3. OTOMECHANIC

OtoMechanic is an end-user application for diagnosing automotive vehicle issues from an audio recording. It can diagnose many more issues than prior systems and is designed to be accessible by people with little or no expertise in automotive diagnosis or repair.

OtoMechanic asks users to provide two inputs: 1) a recording of a troubling sound coming from the user's vehicle and 2) answers to the questions: "Where and when does the sound occur?" Given this information, it queries a database of sounds associated with vehicle issues (see Section 3.3), narrowing the search based on when and where the query sound occurs. Users are then presented with the top 3 matching sounds, ordered by similarity with the user's recording, as well as the diagnoses and confidence level for each matching sound. For each retrieved sound, web links with more information about the diagnosis are provided. This helps users conduct further research on their vehicle's issue.

3.1. Interface Design

The OtoMechanic interface (Figure 1) presents, in sequence, four distinct displays to the user. The first display (Figure 1a) asks where on the vehicle the concerning sound is being produced. Users may respond with *front*, *rear*, or *wheels* of the vehicle, or indicate that they are not sure. The second display (Figure 1b) asks users when the concerning sound occurs. Users may respond with *while starting the car*, *while the engine is idling*, *while driving*, *while braking*, or *while turning*, or indicate that they are not sure. After specifying when and where the sound occurs, users are prompted to upload a recording of the sound (see Figure 1c).

Figure 1d shows example diagnostic results from OtoMechanic. In this example, a recording of a failing battery was uploaded, and no timing or location information was given. The diagnostic results allow users to listen to their uploaded recording and compare it to the three most relevant matches, as determined by our system. The diagnosis corresponding to each returned audio file is provided to the user, as well as a visual indicator of the confidence of the diagnosis and a weblink to Google search using curated search terms relevant to that issue. The search terms used ensure that both text and video descriptions of the diagnosis are included within the first page of search results. This allows users to verify the accuracy of the diagnosis, and determine if they need to take further action (e.g., taking their car to a mechanic for repair).

3.2. Diagnostic Procedure

Once the user provides their audio recording and any qualitative input (i.e., where and when the sound occurs), our system queries a database of annotated sounds of vehicle issues to determine the most likely diagnosis. To do this, our system first filters out all audio files in the database that are inconsistent with the time and location information provided by the user. Our system then computes the similarity between the user's audio recording and each audio file in this list of potential matches.

To compute the similarity between the user's audio recording and a recording in our database, we split each audio file into equal-sized slices of 960 milliseconds at a sampling rate of 16,000 Hz. Each 960 millisecond slice is transformed by passing it through a pretrained VGGish neural network [14]. This network is trained to classify audio taken from millions of YouTube videos. We make use of the modification proposed by Kim and Pardo [15], who show that a feature vector formed from the output of two neural network layers of the trained VGGish model is significantly better for a query-by-example task than using only the output of the final layer.

The final feature vector representation of a single audio recording is the element-wise average of the VGGish feature vectors extracted from all 960 ms slices from that recording. The similarity of a user's recording to a recording in our database is the cosine similarity between these fixed-length feature vectors. In informal experiments on both a commodity laptop and an Amazon Web Services EC2 t2-micro instance, inferring the most relevant audio recordings in the OtoMobile dataset of automotive sounds (see Section 3.3) takes between 200-500 milliseconds for user recordings up to 10 seconds in length, making it suitable for interactive use.

For each retrieved recording, we report a confidence score to the user. Confidence scores are derived by mean-shifting the similarity scores to 0.5 and scaling them to range between 0 and 1 using the mean and range computed over all pairwise similarity scores on the OtoMobile dataset (excluding self-similarities). To minimize the need for users to listen to many recordings, only the top three most similar recordings are displayed.

Currently, OtoMechanic selects one of 12 possible diagnoses. The number of possible diagnoses is determined by the number of different diagnoses that have representation in the dataset of vehicle sounds. As sounds relevant to new vehicle issues are placed in the dataset, OtoMechanic becomes able to suggest these new issues as possible diagnoses.

3.3. OtoMobile Dataset

As no large collection of audio recordings of vehicle issues existed, we curated our own dataset, called the OtoMobile dataset. OtoMobile consists of 65 recordings of vehicles with failing components, along with annotations. These annotations include the diagnosis of the failing component (one of 12 common automobile issues), the location of the component on the car (*front*, *rear*, or *wheels*), the time during which the sound occurred (*while starting*, *while idling*, *while driving*, *while braking*, or *while turning*) the video name and URL, and the start location of the video where the sound was extracted. Excerpts were selected based on the following criteria:

- The diagnosis of the sound coming from the vehicle was provided by either a professional auto mechanic, or someone who had consulted a professional auto mechanic to diagnose the sound.
- At least one second of audio of the problematic vehicle sound was available, during which other noises (e.g., speech) were absent.
- No more than one recording of the same diagnosis was extracted from each video.

The selected audio recordings were cropped to contain only the problematic sounds and normalized to all have the same maximum amplitude. The dataset is available for download for educational and research purposes[1]. Despite the availability of the OtoMobile dataset, we note that data scarcity is still a major bottleneck in auditory vehicle diagnosis.

4. USER STUDIES

We hypothesize that our system improves the ability of nonexperts to identify the vehicular problem causing a particular sound. To test our hypothesis, we conducted two user studies. The goals of the first study were two-fold: 1) understand the existing methods that consumers use to diagnose strange noises coming from their vehicles and 2) determine a baseline accuracy for non-expert diagnosis of such noises. The goal of the second study was to determine whether OtoMechanic can be used to improve non-expert auditory vehicle diagnosis relative to the baseline accuracy determined in our first study.

Both studies made use of Amazon Mechanical Turk (AMT) to recruit and pay participants. We required participants to be from the United States and have at least a 97% acceptance rate on AMT to qualify for our study. 86 participants completed the diagnostic baseline study study and 100 participants completed the system evaluation study.

4.1. Establishing a Diagnostic Baseline

The purpose of our first user study was to understand the existing methods people use to diagnose troublesome vehicle sounds and the efficacy of their methods. In this study, each of our participants was presented a randomly selected audio recording from the OtoMobile dataset (see Section 3.3), along with information about when (i.e. ”when turning”, ”when idling”) and where (i.e. ”from the wheels”) the troublesome sound occurs. Participants were asked to write a brief description of the steps they would take to diagnose the vehicle, given this information.

After participants described their approach, a new portion of the questionnaire was revealed to them, where we asked them to actually diagnose the sound they had been presented in the previous

[1]https://zenodo.org/record/3382945.XXCDG-hKhPY

question. Participants were presented a 12-way forced choice selection of diagnoses. Note that all sounds in the OtoMobile dataset are due to one of 12 issues and that the data set is balanced by issue, so selecting an answer at random will be correct 1/12 of the time.

4.2. Diagnostic Baseline User Study Results

Participant descriptions of their diagnosis method indicated a strong preference for manual inspection as a diagnostic method. Of the 86 diagnostic methods described by participants, 73 of the descriptions mentioned a physical interaction with the vehicle that they would use to gain more information on the issue. We found that participants were far more willing to actively participate in diagnosing the vehicle than to seek out a professional auto mechanic; only 23 of our 86 participants mentioned interacting with a mechanic in order to diagnose their vehicle.

Despite the wide availability of online resources, our results suggest that participants are unlikely to connect with these resources. Only four participants mentioned they would use online resources to assist their diagnosis. We hypothesize that this may be due to the difficulty that non-experts face in constructing relevant search terms. For example, one participant describes a process of first isolating symptoms of the vehicle and then using that information to construct a search query: "I'd see what factors might affect [the sound], e.g. stepping on the gas, changing gears. Then I'd look up potential answers on the Web." The ability for OtoMechanic to connect users to online resources by providing relevant web links associated with a diagnosis represents a potential solution to the lack of utilization of online resources that we observed.

Only two participants mentioned asking a friend or family member, and only two participants mentioned accessing their car's computer via an external OBD reader. Responses indicate a large variance in experience with vehicle repairs. One participant self-reported that they "don't know anything about cars", while others indicated significant knowledge of car components and prior experience replacing brake pads, fuel pumps, and drive belts.

When asked to select the diagnosis that best corresponded to a randomly selected audio recording from the OtoMobile dataset, participants were able to identify the correct diagnosis with 37.2% accuracy. Chance performance on this 12-way forced choice task is 8.3%.

4.3. System Evaluation User Study

We evaluate the efficacy of the OtoMechanic approach through a second user study. This study was performed by a new group of 100 participants. Each study participant was given a random audio recording from the OtoMobile dataset and the same qualitative information as in the first study (i.e., when and where the sound occurred). As with the diagnostic portion of the previous study, participants were presented a 12-way forced choice selection of diagnoses and chance performance on this task was 1/12.

In this study, participants were asked to use OtoMechanic to diagnose the audio recording, and provide their diagnosis by selecting from the same diagnostic options as in our first user study. To prevent trivial similarities, the audio recording provided to each participant was removed from the list of possible matches retrieved by OtoMechanic.

Diagnostic Method	Accuracy (%)
Random	8.3
Human Baseline	37.2
OtoMechanic (no time or location information)	34.8
Random (with time and location information)	39.3
Humans using OtoMechanic	57.0
Oracle System Performance	58.7

Table 1: Diagnostic accuracies of experiments on the OtoMobile dataset

4.4. Oracle System Classification Results

We present the diagnostic accuracy of selecting the diagnosis corresponding to the best matching audio recording determined by OtoMechanic on a 12-way classification of the OtoMobile dataset. When the location and time information is included as input alongside the audio recording, our system achieves an accuracy of 58.7% on the OtoMobile dataset. The likelihood of the correct diagnosis being one of the 3 results returned to the user by OtoMechanic (i.e., the top-3 accuracy) is 82.9%. Without using the information about when and where the sound occurred, our system achieves an accuracy of 34.8%, with a top-3 accuracy of 53.0%. If we use the location and time information but choose randomly from the recordings matching those criteria, our system achieves an accuracy of 39.3%. This indicates that the knowing when and were the sound occurs is a significant factor in diagnostic accuracy.

4.5. System Evaluation User Study Results

When asked to use OtoMechanic to diagnose a troublesome vehicle sound, our 100 participants achieved a diagnostic accuracy of 57.0%. This is nearly equal to the 58.7% accuracy achieved by simply selecting the top choice indicated by OtoMechanic and significantly greater than the 37.2% achieved by participants on the identical task when performed without access to OtoMechanic. This indicates that our application is significantly more effective at diagnosing vehicle issues than a baseline of prior knowledge of participants.

5. DISCUSSION

Vehicle failure poses serious risk to public health. Road traffic crashes are the 9th leading cause of death worldwide, and in the US alone there are an estimated 800 deaths and 47,000 injuries each year due to vehicle failures [16, 17]. OtoMechanic provides a convenient method for informing consumers of potential failures due to damaged vehicle components. Our user studies showed use of OtoMechanic significantly increases the accuracy of vehicle diagnosis from troublesome sounds without requiring expert knowledge of vehicle maintenance or repair. Our application connects non-expert consumers to resources that enable them to make educated decisions about their vehicle. In future work, we aim to increase the size of the OtoMobile dataset and the specificity of the annotations (e.g., the make and model of the vehicle). We also aim to compare our system against more challenging baselines (e.g., the performance of professional auto mechanics) and investigate the utility of OtoMechanic in on-board vehicle computers.

6. REFERENCES

[1] F. T. Commission, “Auto repair basics,” https://www.consumer.ftc.gov/articles/0211-auto-repair-basics, 2012.

[2] J. E. Siegel, Y. Sun, and S. Sarma, “Automotive diagnostics as a service: An artificially intelligent mobile application for tire condition assessment,” in *International Conference on Artificial Intelligence: Methodology, Systems, and Applications*. Springer, 2018, pp. 172–184.

[3] C. Talk, “Car noise emporium,” https://www.cartalk.com/content/car-noise-emporium-23, 2019.

[4] N. T. LLC, “Clingclanger: When your car makes noise,” http://mycarmakesnoise.com/, 2017.

[5] T. A. Rules, “What does a bad wheel bearing sound like?” https://www.youtube.com/watch?v=UpsLaSzcAu4, 2014.

[6] Jyarf, “What does a broken sway bar link sound like? watch and listen.” https://www.youtube.com/watch?v=36QW9UgGC9Q, 2013.

[7] S. Delvecchio, P. Bonfiglio, and F. Pompoli, “Vibro-acoustic condition monitoring of internal combustion engines: A critical review of existing techniques,” *Mechanical Systems and Signal Processing*, vol. 99, pp. 661–683, 2018.

[8] A. Deptuła, D. Kunderman, P. Osiński, U. Radziwanowska, and R. Włostowski, “Acoustic diagnostics applications in the study of technical condition of combustion engine,” *Archives of Acoustics*, vol. 41, no. 2, pp. 345–350, 2016.

[9] T. Figlus, J. Gnap, T. Skrúcanỳ, B. Šarkan, and J. Stoklosa, “The use of denoising and analysis of the acoustic signal entropy in diagnosing engine valve clearance,” *Entropy*, vol. 18, no. 7, p. 253, 2016.

[10] P. Henriquez, J. B. Alonso, M. A. Ferrer, and C. M. Travieso, “Review of automatic fault diagnosis systems using audio and vibration signals,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 5, pp. 642–652, 2014.

[11] R. F. Navea and S. Edwin, “Design and implementation of an acoustic-based car engine fault diagnostic system in the android platform,” 10 2013.

[12] D. Ning, J. Hou, Y. Gong, Z. Zhang, and C. Sun, “Auto-identification of engine fault acoustic signal through inverse trigonometric instantaneous frequency analysis,” *Advances in Mechanical Engineering*, vol. 8, no. 3, 2016.

[13] J. Siegel, S. Kumar, I. Ehrenberg, and S. Sarma, “Engine misfire detection with pervasive mobile audio,” in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2016, pp. 226–241.

[14] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, “Cnn architectures for large-scale audio classification,” in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[15] B. Kim and B. Pardo, “Improving content-based audio retrieval by vocal imitation feedback,” in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 4100–4104.

[16] N. H. T. S. Administration, “Critical reasons for crashes investigated in the national motor vehicle crash causation survey,” https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115, 2 2015.

[17] A. for Safe International Road Travel, “Road safety facts,” https://www.asirt.org/safe-travel/road-safety-facts/.

ONSETS, ACTIVITY, AND EVENTS: A MULTI-TASK APPROACH FOR POLYPHONIC SOUND EVENT MODELLING

Arjun Pankajakshan[1], *Helen L. Bear*[1], *Emmanouil Benetos*[1,2],

[1] School of EECS, Queen Mary University of London, UK [2] The Alan Turing Institute, UK
{a.pankajakshan, h.bear, emmanouil.benetos}@qmul.ac.uk

ABSTRACT

State of the art polyphonic sound event detection (SED) systems function as frame-level multi-label classification models. In the context of dynamic polyphony levels at each frame, sound events interfere with each other which degrade a classifier's ability to learn the exact frequency profile of individual sound events. Frame-level localized classifiers also fail to explicitly model the long-term temporal structure of sound events. Consequently, the event-wise detection performance is less than the segment-wise detection. We define 'temporally precise polyphonic sound event detection' as the subtask of detecting sound event instances with the correct onset. Here, we investigate the effectiveness of sound activity detection (SAD) and onset detection as auxiliary tasks to improve temporal precision in polyphonic SED using multi-task learning. SAD helps to differentiate event activity frames from noisy and silence frames and helps to avoid missed detections at each frame. Onset predictions ensure the start of each event which in turn are used to condition predictions of both SAD and SED. Our experiments on the URBAN-SED dataset show that by conditioning SED with onset detection and SAD, there is over a three-fold relative improvement in event-based F-score.

Index Terms— Polyphonic sound event detection, sound activity detection, onset detection, multi-task learning.

1. INTRODUCTION

Sound event detection (SED) [1] is the task of detecting the label, onset, and offset of sound events in audio streams. State of the art convolutional recurrent neural network (CRNN) based polyphonic SED systems use a frame-wise cost function for training [2, 3, 4]. The frame-level classifier performance depends on the dynamic polyphony level, masking effects between the sound events and the amount of co-occurrence of sound events in the training data. Frame-level classifiers also fail to explicitly model the long-term temporal structure of sound events. Due to these limitations, frame-level training methods are not sufficient to model the overall acoustic features of polyphonic sound events. Consequently, the event-based detection performance is very poor compared with the segment-based detection of sound events. For example, in the DCASE 2016 task on event detection in real life audio [5], the F-score is around 30% at segment level (frame length of 1 second), but only around 5% at event level (tolerance of 200 ms for onset and 200 ms or half length for offset). Here, we define 'temporally precise polyphonic sound event detection' as the subtask of detecting sound event instances with the correct onset. In applications like audio surveillance [6] and health care monitoring [7], temporally accurate event-based detection is very important.

AP is supported by a QMUL Principal's studentship. EB is supported by RAEng Research Fellowship RF/128 and a Turing Fellowship. This research was supported by an NVIDIA GPU Grant.

1.1. Related work

SED is related to the speech processing tasks of automatic speech recognition and speaker diarization, as well as the music signal-related task of automatic music transcription [8, 9]. Many sequence modelling methods in speech and music have been utilized in environmental sound event modeling. For example, in [10] hidden semi-Markov models separately model the duration of sound events; Wang and Metze used a connectionist temporal classification (CTC) cost function in a sequence-to-sequence model for SED [11]. Unlike speech and music language modelling there is not a well defined structure for environmental sound events. Explicit use of sequential information to improve sound event modelling is investigated in [12]; the co-occurrence probabilities of different events are modelled using N-grams which in turn smooth the spiky output of a neural net based SED system trained using CTC loss. However the SED performance is not much improved, considering the addition of both N-grams and the CTC loss. In [13] a hybrid approach that combined an acoustic-driven event boundary detection for sound event modelling with a supervised label prediction model is proposed for SED. This hybrid approach significantly improved event-based detection accuracy. It is assumed that the method requires additional post-processing to combine the event boundary information with the label predictions since both models are trained independently.

1.2. Contributions of this work

Within the limits of frame-wise training approaches using CRNN models for polyphonic SED, we propose a novel sequence modelling method using onset detection and SAD as auxiliary tasks to achieve temporally precise polyphonic SED using multi-task learning. SAD is the task of detecting the presence or absence of any sound events, which is analogous to voice activity detection in speech processing. The effectiveness of SAD to improve polyphonic SED is discussed in a preliminary work by the authors [14]. Unlike polyphonic SED, SAD is not affected by masking effects and acoustic variations in the sound events even when it is trained using frame-wise cost functions. The onset detector helps to predict the beginning of sound events accurately, thus reducing missed event detections which improves temporally precise SED. Both onset detection and SAD are not overwhelmed by polyphonic structure, instead these auxiliary tasks can exploit the polyphonic nature to improve temporal precision in SED. Inspired from the success of

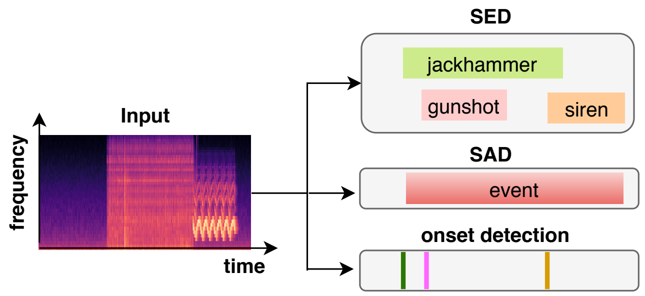


Figure 1: Block diagram of SED, SAD, and onset detection.

framewise note detection in piano transcription conditioned on onset predictions [15], we configure multi-task models for SED in a similar fashion. We investigate the individual effectiveness of conditioning onset detection and SAD on SED. Also we propose a joint model to improve the temporal precision of sound events in polyphonic SED.

2. PROPOSED METHOD

In this work, we investigate the effectiveness of onset detection and SAD to improve the temporal precision of polyphonic SED. Onset detection exclusively predicts the beginning of a sound event instance, which is useful because many sound event onsets are characterized by sudden increase in energy, e.g. percussive sound events. The onset predictions are used to condition framewise SED in a similar way as the music transcription in [15]. Conditioning SED based on onset predictions helps to precisely locate the beginning of sound events. SAD predicts whether any event activity is present or not in each frame of the audio and so avoids the pitfalls caused by masking effects between co-occurring sound events. Furthermore, SAD can exploit polyphony to ensure the presence of an event even if one event is masked by the occurrence of another event with similar or different acoustic properties. Hence conditioning SED with SAD helps to avoid missed detections in complex acoustic conditions such as real-world sound scenes. Fig. 1 shows the complete system.

We use a state-of-the-art CRNN model architecture ([2]) to build baseline SED, SAD, and onset models. To evaluate the individual effect of each auxiliary task on SED, we implement and analyse separate SED models conditioned on onset prediction and sound activity prediction using a multi-task learning set up.

- *sed_onset* is the SED model conditioned on onset prediction.
- *sed_sad* is the SED model conditioned on sound activity.
- The *sad_onset* model verifies the effect of SAD conditioned on onset detection.
- The joint SED model using both onset detection and SAD as auxiliary tasks, is denoted as *sed_sad_onset*.

2.1. Motivation for onset detection

We examine the importance of onset detector as an auxiliary task in polyphonic SED in different perspectives. Firstly, the onset detector is able to predict the beginning of sound events more effectively compared to a standalone baseline SED model which is affected by dynamic polyphony levels and acoustic variations of the sound events. Secondly, consider the case of two closely occurred sound events. Even if the onset detector could only detect either of the events, the same prediction can be used as two separate onsets while conditioning the event detection with minimum error. This way the conditioned SED model can avoid missed detection of sound events and ensure the detection of two closely occurred events even if the onset detector actually predicted only a single onset. We evaluate the onset models based on this assumption. More precisely, there is a one-to-many relation between the onset prediction and the reference. We consider this fact when counting the false negatives for onset model evaluation and verified this relaxation does not make much difference.

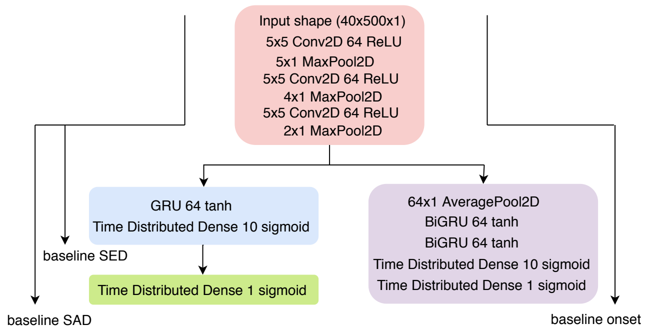


Figure 2: Baseline architectures for onset detection, SAD and SED.

2.2. Model configuration

The detailed network architecture of the three baseline models is in Fig. 2. We replicate the SED and SAD implementation from [14] and extend it with our onset detector. The onset detector has three blocks of convolutional layers. The output activations from the third convolutional layer are averaged using an average pooling layer, followed by two bidirectional Gated Recurrent Unit (GRU) layers and a fully connected sigmoid to output the onset predictions. Onsets are very localized sound events; hence the inter feature map representations learned at the final convolution layer are equally important to predict onsets so we opted for average-pooling across the third convolution layer feature maps instead of max-pooling. Bidirectional recurrent layers are proven to work well for musical onset detection [16]. The output of the SED model is a posteriogram matrix with dimensions $T \times C$, where T is the number of frames in the input data representation and C is the total number of sound event classes in the dataset. The output representation of the sound activity detector and the onset detector is a posteriogram vector with dimension T. The baseline model predictions are binarised with a threshold before evaluation. We investigate different threshold values on the baseline models using the validation set. Using the best results, we chose a threshold of 0.2 for the SED and onset predictions and 0.5 for the SAD predictions.

We implement conditional models for SED (*sed_sad*, *sed_onset*, *sed_sad_onset*) and SAD (*sad_onset*) using the baseline SED, SAD, and onset models in a multi-task joint training setup. Motivated from the effectiveness of piano note transcription conditioned on onsets, initially we implement our conditional models as explained in [15]. However, in our case the exact same architecture was not effective so we explore the possibilities of multitask learning [17, 18, 19, 20]. From our experiments we fix our

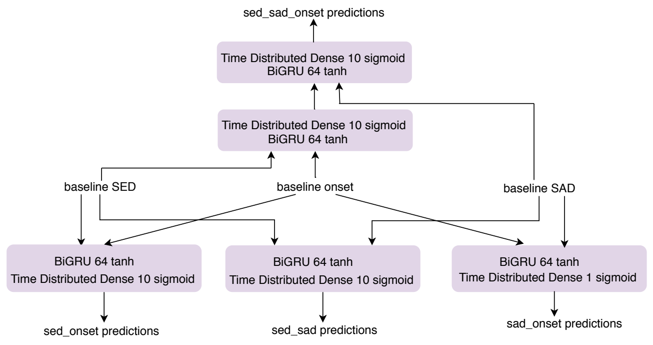

sed_sad_onset predictions
Time Distributed Dense 10 sigmoid
BiGRU 64 tanh
Time Distributed Dense 10 sigmoid
BiGRU 64 tanh
baseline SED
baseline onset
baseline SAD
BiGRU 64 tanh
Time Distributed Dense 10 sigmoid
BiGRU 64 tanh
Time Distributed Dense 10 sigmoid
BiGRU 64 tanh
Time Distributed Dense 1 sigmoid
sed_onset predictions
sed_sad predictions
sad_onset predictions

Figure 3: Block diagram of the proposed conditional models.

conditional model architectures by sharing the initial two convolutional layers of the respective tasks. The conditional model training is described in Section 2.4.

- We implement the SED model conditioned on activity detection (sed_sad) by concatenating the predictions of the SAD baseline model with the output of the baseline SED model, followed by a bidirectional GRU layer and a fully connected sigmoid layer to predict the sound events.
- The predictions of the onset baseline model are concatenated with the output of the baseline SED architecture, followed by a bidirectional GRU layer and a fully connected sigmoid layer to implement the SED model conditioned on onset detection (sed_onset). Similarly, we implement the SAD model conditioned on onset detection (sad_onset).

In the sad_onset model the activity detection is conditioned using the reference event onsets which are different from the onsets for sound activity. For example, if two sound events are overlapping with each other, there are two sound event onsets but only a single sound activity onset. The additional onsets in the event onset reference condition unwanted preference to the corresponding activity frames which cause slight disturbance of activity detection around the respective frames. As a result of this the segment-based F-score for activity detection of the sad_onset model is lower than the baseline SAD (shown in Section 4). In our analysis of the sed_onset model and the sed_sad model we realize that, conditioning SED using both onset detection and activity detection improves SED temporal precision. So we design the final joint conditioned model (sed_sad_onset) by first conditioning SED using onsets and then the onset-conditioned event prediction is reconditioned using activity predictions. Fig. 3 is a block diagram of the conditional models.

- For the sed_sad_onset model, the predictions of the onset baseline model are concatenated with the baseline SED outputs, followed by a bidirectional GRU layer and a fully connected sigmoid layer. The output from this sigmoid layer is concatenated with the predictions of the activity detector and passed through a bidirectional GRU layer and a fully connected sigmoid layer.

2.3. Feature extraction

We use librosa [21] to compute mel-scaled spectrograms from the input audio. The short-term Fourier transform (STFT) is employed to obtain the spectrogram from the input audio recordings with a hop length of 882, an FFT window of 2048, and a sample rate of 44.1 kHz. This process converts a ten second (duration of recordings in the URBAN-SED [22] dataset) audio recording into a 1024×500 dimensional spectrogram representation. Each frame of this spectrogram is converted into a 40-dimensional vector of log filter bank energies using a Mel filterbank. We apply min-max normalization on the mel band energies. Hence, each 10-second audio recording is represented by a 40×500 Mel-spectrogram.

2.4. Training

We train all the models in a supervised manner. The dimension of the labels for the SED is $T \times C$, and for the SAD and the onset detection, it is T. The training process for the SED and the SAD models is explained in our previous work [14]. For the onset detection, a single frame is used to mark each onset during the training process. The baseline models for SED, SAD and onset detection are trained using the respective cross-entropy losses denoted by L_{sed}, L_{sad}, and L_{onset}. The total loss of each of the conditional models is the weighted sum of the two corresponding cross-entropy losses as listed in (1). During training of the conditional models, the individual losses are equally weighted with a factor of 0.5.

$$\begin{aligned}
L_{sed\_sad} &= 0.5 \cdot L_{sed} + 0.5 \cdot L_{sad} \\
L_{sed\_onset} &= 0.5 \cdot L_{sed} + 0.5 \cdot L_{onset} \\
L_{sad\_onset} &= 0.5 \cdot L_{sad} + 0.5 \cdot L_{onset} \\
L_{sed\_sad\_onset} &= 0.5 \cdot L_{sed} + 0.5 \cdot L_{sad} + 0.5 \cdot L_{onset}
\end{aligned} \tag{1}$$

Every CNN layer activations are batch normalised [23] and regularised with dropout [24] (probability = 0.3). We train the network for 200 epochs using a binary cross entropy loss function for both tasks and with Adam [25] optimizer with a learning rate of 0.001. Early stopping is used to reduce overfitting. The proposed joint model is implemented using Keras with TensorFlow.

3. DATASET AND METRICS

We use the URBAN-SED [22] dataset in all experiments. URBAN-SED is a dataset of 10,000 soundscapes with sound event annotations generated using Scaper [22], an open-source library for soundscape synthesis. All recordings are ten seconds, 16-bit mono and sampled at 44.1kHz. The annotations are strong, meaning for every sound event the annotations include the onset, offset, and label of the sound event. Each soundscape contains between one to nine sound events from the list (air_conditioner, car_horn, children_playing, dog_bark, drilling, engine_idling, gun_shot, jackhammer, siren and street_music) and has a background of Brownian noise. The URBAN-SED [22] dataset comes with pre-sorted train, validation and test sets that we use. Among $10,000$ soundscapes, 6000 are used for training, 2000 for validation and 2000 for testing.

We use precision, recall and F-score as metrics for onset detection. For sound event and sound activity detection we use the F-score and Error Rate (ER), with F-score as the primary metric. The evaluation metrics are computed in both segment-wise and event-wise manners using the sed_eval tool [26]. Segment-based metrics show how well the system correctly detects the temporal regions where a sound event is active; with an event-based metric, the metric shows how well the system detects event instances with correct onset and offset. For temporally precise event detection, we give more importance to the event-based metric. The evaluation scores for activity detection and event detection are micro averaged values, computed by aggregating intermediate statistics over all test data; each instance has equal influence on the final metric value. We use a segment length of one second to compute segment metrics. The event-based metrics are calculated with respect to event instances by evaluating only onsets with a time collar of 250ms.

In the case of onset detection, an onset is considered to be correctly detected if there is a ground truth annotation within ±250ms around the predicted position. An important factor in the evaluation is how false positives and false negatives are counted [27]. Assume that two or more onsets are predicted inside the detection window around a single reference annotation. All predictions within the detection window around the single reference onset are treated as one true positive and zero false positives. The false negatives are counted by granting a one-to-many relationship between a single prediction and multiple reference onsets within an analysis window (±250 ms). Since our main goal is to use onset detection to condition SED we believe this evaluation approach is fair.

4. EVALUATION

Tables 1, 2, and 3 show the results of onset detection, sound activity detection and sound event detection respectively. Table 1, the baseline onset detector, has the best F-score (81.68%). When onset detection is used to condition SAD and SED, the onset F-scores are slightly lower than the baseline value. However, conditioning activity detection and event detection using onsets is really effective in temporally precise SED. This is demonstrated in Tables 2 and 3. By conditioning activity detection using onsets (*sad_onset* in Table 2), the event-based activity detection F-score increases from the baseline value of 43.14% to 70.31%. At the same time the segment-based F-score for the same model drops from 97.48% to 70.17%. This is due to the fact that the sound event onset labels are used to condition the activity detection which is different from the actual onsets of sound activity as explained in Sec 2. We see a similar improvement when SED is conditioned using onsets (*sed_onset* in Table 3). For this model the event-based F-score increases from the baseline value of 7.34% to 21.42%. The segment-based F-score for the same model is 47.76% compared with the baseline value of 35.48%. The improvement in the event-based F-scores for the *sad_onset* model and the *sed_onset* model verify the effectiveness of onset conditioned polyphonic event detection to improve temporal precision in polyphonic SED.

The *sed_sad* model performance (in Table 3) is compared with a joint model to enhance event detection by re-weighting the event prediction using the activity prediction from [14] (*sed_sad_joint* in Table 3). The *sed_sad* model segment-based and event-based F-score values are 43.52% and 17.40% respectively; both are improvements from the baseline and *sed_sad_joint* model. By analysing the *sed_sad* and *sed_onset* results, we know conditioning event detection using onsets is more effective than conditioning using sound activity. To utilize the advantage of both onsets and sound activity frames in conditioning event detection we implement the final conditional model (*sed_sad_onset*) as explained in Sec 2. Using this model the event-based F-score improves to 23.20%.

Further analysis of onset detection performance of the conditional models (*sed_onset* and *sed_sad_onset*) reveal that when false positive errors in onset detection are less, the sound event model is more effective. More precisely, when the precision of onset detection improved from 85.96% to 89.17% from the *sed_onset* model to the *sed_sad_onset* model the event-based F-score also improved from 21.42% and 23.20%; which implies that false positive errors in the onset detection are more influential than false negative errors in the performance of conditional event detection using onsets. This analysis again proves the effectiveness of conditional sound event detection using onsets. The class-wise evaluation of all the models are available[1].

Table 1: Onset detection results.

Case	Precision	Recall	F-score
baseline	81.16	**82.20**	**81.68**
sad_onset	**90.09**	73.28	80.82
sed_onset	85.96	74.31	79.71
sed_sad_onset	89.17	70.68	78.85

Table 2: Sound activity detection results.

	F1 (%)		Error rate	
Case	Segment	Event	Segment	Event
baseline	97.48	43.14	0.05	0.78
sed_sad_joint	**98.53**	46.23	**0.03**	0.72
sad_onset	70.17	**70.31**	0.46	**0.61**
sed_sad	98.39	45.53	**0.03**	0.73
sed_sad_onset	97.58	46.51	0.05	0.76

Table 3: Sound event detection results.

	F1 (%)		Error rate	
Case	Segment	Event	Segment	Event
baseline	35.48	7.34	1.54	3.81
sed_sad_joint	41.03	8.76	0.97	3.58
sed_sad	43.52	17.40	0.88	1.68
sed_onset	**47.76**	21.42	1.02	2.33
sed_sad_onset	44.12	**23.20**	**0.85**	**1.49**

5. CONCLUSIONS AND FUTURE WORK

Within the limits of frame-wise training in sequence modelling problems, we proposed a novel sequence modelling method for temporally precise polyphonic sound event detection conditioned on onsets and sound activity detection. From our experimental results we conclude that: 1) the performance of temporally precise event detection of the conditional models depends on the performance of onset and sound activity detection and also on the conditioning method. 2) conditioning the main task using auxiliary tasks and training in a multi-task set up is an effective method to improve SED performance. We believe an onset detector with precision and recall measures close to 100% can drastically improve temporal precision in SED performance. In the future, we plan to improve onset detection by modifying the onset loss with dedicated penalty terms for false positive and false negative onset predictions. Also our current conditional models are trained using an equally weighted sum of cross entropy losses of the individual tasks. Instead of this approach we plan to develop a task-dependent weighting scheme explicitly conditioning auxiliary tasks to derive a principled multi-task loss function as demonstrated in [28].

[1]`http://c4dm.eecs.qmul.ac.uk/DCASE2019/class-wise_evaluation_supplementary_doc.pdf`

6. REFERENCES

[1] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational analysis of sound scenes and events*. Springer, 2018.

[2] E. Cakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.

[3] E. Çakir and T. Virtanen, "End-to-end polyphonic sound event detection using convolutional recurrent neural networks with learned time-frequency representation input," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–7.

[4] S. Adavanne and T. Virtanen, "Sound event detection using weakly labeled dataset with stacked convolutional and recurrent neural network," *Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2017.

[5] "Sound event detection in real life audio, dcase 2016 challenge results," http://www.cs.tut.fi/sgn/arg/dcase2016/task-results-sound-event-detection-in-real-life-audio.

[6] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Reliable detection of audio events in highly noisy environments," *Pattern Recognition Letters*, vol. 65, pp. 22–28, 2015.

[7] S. Goetze, J. Schroder, S. Gerlach, D. Hollosi, J.-E. Appell, and F. Wallhoff, "Acoustic monitoring and localization for social care," *Journal of Computing Science and Engineering*, vol. 6, no. 1, pp. 40–50, 2012.

[8] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4774–4778.

[9] S. Sigtia, E. Benetos, and S. Dixon, "An end-to-end neural network for polyphonic piano music transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 5, pp. 927–939, 2016.

[10] T. Hayashi, S. Watanabe, T. Toda, T. Hori, J. Le Roux, K. Takeda, T. Hayashi, S. Watanabe, T. Toda, T. Hori, *et al.*, "Duration-controlled lstm for polyphonic sound event detection," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 25, no. 11, pp. 2059–2070, 2017.

[11] Y. Wang and F. Metze, "A first attempt at polyphonic sound event detection using connectionist temporal classification," in *ICASSP 2017-2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 2986–2990.

[12] G. Huang, T. Heittola, and T. Virtanen, "Using sequential information in polyphonic sound event detection," in *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*. IEEE, 2018, pp. 291–295.

[13] S. Kothinti, K. Imoto, D. Chakrabarty, G. Sell, S. Watanabe, and M. Elhilali, "Joint acoustic and class inference for weakly supervised sound event detection," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 36–40.

[14] A. Pankajakshan, H. L. Bear, and E. Benetos, "Polyphonic sound event and sound activity detection: A multi-task approach," *in Applications of Signal Processing to Audio and Acoustics (WASPAA), 2019.*

[15] C. Hawthorne, E. Elsen, J. Song, A. Roberts, I. Simon, C. Raffel, J. Engel, S. Oore, and D. Eck, "Onsets and frames: Dual-objective piano transcription," *International Conference of Music Information retrieval (ISMIR)*, 2018.

[16] F. Eyben, S. Böck, B. Schuller, and A. Graves, "Universal onset detection with bidirectional long-short term memory neural networks," in *Proc. 11th Intern. Soc. for Music Information Retrieval Conference (ISMIR)*, 2010, pp. 589–594.

[17] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th International Conference on Machine Learning (ICML)*. ACM, 2008, pp. 160–167.

[18] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning," *in International Joint Conference on Artificial Intelligence (IJCAI-16)*, 2016.

[19] Y. Zhang and Q. Yang, "A survey on multi-task learning," *arXiv preprint arXiv:1707.08114*, 2017.

[20] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *Advances in Neural Information Processing Systems (NIPS)*, 2007, pp. 41–48.

[21] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th Python in Science Conference*, 2015, pp. 18–25.

[22] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 344–348.

[23] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *International Conference on Machine Learning (ICML)*, 2015.

[24] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[25] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations (ICLR)*, 2015.

[26] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

[27] S. Böck, F. Krebs, and M. Schedl, "Evaluating the online capabilities of onset detection methods." in *Intern. Soc. for Music Information Retrieval Conference (ISMIR)*, 2012, pp. 49–54.

[28] A. Kendall, Y. Gal, and R. Cipolla, "Multi-task learning using uncertainty to weigh losses for scene geometry and semantics," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7482–7491.

TRELLISNET-BASED ARCHITECTURE FOR SOUND EVENT LOCALIZATION AND DETECTION WITH REASSEMBLY LEARNING

Sooyoung Park, Wootaek Lim, Sangwon Suh, Youngho Jeong,

Electronics and Telecommunications Research Institute
Media Coding Research Section
218 Gajeong-ro, Yuseong-gu, Daejeon, Korea
{sooyoung, wtlim, suhsw1210, yhcheong}@etri.re.kr

ABSTRACT

This paper proposes a deep learning technique and network model for DCASE 2019 task 3: Sound Event Localization and Detection. Currently, the convolutional recurrent neural network is known as the state-of-the-art technique for sound classification and detection. We focus on proposing TrellisNet-based architecture that can replace the convolutional recurrent neural network. Our TrellisNet-based architecture has better performance in the direction of arrival estimation compared to the convolutional recurrent neural network. We also propose reassembly learning to design a single network that handles dependent sub-tasks together. Reassembly learning is a method to divide multi-task into individual sub-tasks, to train each sub-task, then reassemble and fine-tune them into a single network. Experimental results show that the proposed method improves sound event localization and detection performance compared to the DCASE 2019 baseline system.

Index Terms— DCASE 2019, sound event localization and detection, TrellisNet, convolutional recurrent neural network, reassembly learning

1. INTRODUCTION

A sound event localization and detection (SELD) [1, 2] is a new estimation problem that combines sound event detection (SED) and direction of arrival estimation (DOAE) into a single task. Hirvonen [3] suggested a classification approach for DOAE using convolutional neural networks (CNN). The disadvantage of DOAE using classification is that only discrete direction of arrival (DOA) can be predicted. Besides, applying this method to polyphonic sound events increases the number of target classes and requires a large amount of dataset to train the network. Therefore, the DCASE 2019 baseline [1, 2] tried to overcome these disadvantages using multi-output regression.

Multi-output regression is a method to fit regressors for all target classes. So the DCASE 2019 baseline estimates DOA for both active and inactive sound events. As a result, multi-output regression interrupts training for DOAE because of unnecessary inactive events. Multi-output regression forces the DOA label to have azimuth and elevation values for all classes. Therefore, DOA labels have true DOA values for active events and default DOA values for inactive events. Multi-output regression loss includes both DOA loss for active events and DOA loss for inactive events. As a result, the network output is trained to be closer to the true DOA for active events and to the default DOA for inactive events. In short, SELD with multi-output regression operates to estimate the array containing the default DOA values, rather than estimating DOA only for active events. To overcome this problem, Cao et al. [4] proposed a two-stage learning method to avoid loss from inactive events. The two-stage learning excludes DOA prediction for inactive events by masking using ground-truth event labels. As a result, the two-stage learning makes significant performance improvement.

The two-stage learning solves the problem of multi-output regression by excluding the inactive events from the DOA loss. However, the two-stage learning still has a problem with inactive events at the inference stage. The two-stage learning derives the final SELD output prediction by concatenating SED network prediction and DOA network prediction. Here, the DOA network of two-stage learning excludes the inactive events in training. Therefore, the DOA prediction values of the inactive events are random. As a result, there is a problem in the Hungarian algorithm used to calculate the DOA error for polyphonic sound events. This problem occurs when the SED network incorrectly predicts the event. Since the Hungarian algorithm is a method to find a combination that reduces the pair-wise cost, the DOA error is derived from the random DOA output of the inactive sound event in the above case. In short, the two-stage learning causes the irony that DOAE for mispredicted events is made with random DOA predictions excluded from training.

In this paper, we propose a reassembly learning method to partially alleviate the problem from inactive sound events. This method is based on the two-stage learning [4]. Reassembly learning is a method of reassembling pre-trained SED network and DOA network into a single SELD network, and training the reassembled SELD network. The key idea of reassembly learning is to reduce the influence of inactive sound events through fine-tuning.

A recurrent neural network (RNN) is widely used for sequence modeling. Theoretically, RNN can train an infinite length of the sequence. However, in actual RNN, the vanishing gradient occurs while repeating the sequential operation. It means that the stability of the RNN structure is not guaranteed. Therefore gating mechanism has been proposed such as LSTM [5] and GRU [6]. But the instability problem was not completely solved. There have been attempts to process sequence data using TCN, which is based on 1D convolution and showed good performance [7, 8]. Bai et al. [9] proposed TrellisNet, a new architecture that takes advantage of two advantages of CNN and RNN. TrellisNet is a special form of TCN structure that stacks multiple layers of TCNs and acts like a truncated RNN at a specific weight. TrellisNet outperforms several benchmarks, such as language modeling and long-term memory retention [9].

CRNN is used in DCASE 2019 baseline. Furthermore, many SED and DOAE studies [1, 2, 4, 10] choose CRNN as basic network

https://doi.org/10.33682/nzsm-jr50

architecture. CRNN is currently state-of-the-art in sound classifications and detection. CRNN architecture uses CNN for local feature extraction and RNN for a temporal summary of the output of the CNN. In this paper, we propose a new network architecture for sound classification and detection using a TrellisNet [9] based on the temporal convolutional network (TCN). The key idea of the proposed architecture is to take advantage of both CNN and RNN by replacing RNN with TrellisNet.

2. DATASET

DCASE 2019 challenge task 3 provides audio dataset for 11 classes of sound events. The sound event of DCASE 2019 dataset is synthesized using spatial room impulse response recorded in five indoor locations. The development dataset consists of 400 files. Each audio file is a one-minute duration with a sampling rate of 48000 Hz. The development dataset is provided as two different types: four channel tetrahedral microphone arrays and a first-order ambisonic (FOA) format. Besides, DCASE challenge task 3 targets polyphonic sound events with a maximum of two sound events overlap.

3. FEATURE

Our models use log mel-band energy (4 channels), mel-band acoustic active intensity (3 channels) and mel-band acoustic reactive intensity (3 channels). Log mel-band energy is extracted from the tetrahedral microphone dataset. On the other hand, mel-band acoustic active and reactive intensity are extracted from FOA dataset. The input feature configuration used in this paper is shown in Table 1.

3.1. Log mel-band energy

In the DCASE 2018 challenge task 4, many participants used the log mel-band energy as an input feature for SED [11–16]. Mel-band energy is a feature that applies a mel filter to an energy spectrogram. The mel filter mimics the non-linear human auditory perceptions. The results of DCASE 2018 challenge proved that this non-linear feature has strength for SED. Also, we expect to obtain information of time difference, loud difference for sound localization from a multi-channel log mel-band energy feature.

3.2. Mel-band acoustic intensity

Ambisonic is a coefficient of the spatial basis of the audio signal. Each spatial basis is expressed as spherical harmonics. Zero-order ambisonic signal (W) represents the component that is omnidirectional. First-order ambisonic signals (X, Y, Z) represent three polarized bidirectional components. In the presence of multiple sources or reverberant environments, it is impossible to express complex sound fields using FOAs (W, X, Y, Z). Therefore, we need additional methods to extract spatial information from FOAs for the reverberant environment. Acoustic intensity is one of these methods that extract spatial information from FOAs [17].

Acoustic intensity is one of the physical quantities representing the sound field. The acoustic intensity vector $\mathbf{I}(t,f)$ can be expressed by using FOA as equation (1). Active acoustic intensity vector I_a is a real part of acoustic intensity that represents the flow of sound energy. It is a physical quantity directly related to DOA. The active acoustic intensity is expressed as a real part of the product of the pressure $p(t,f)$ and the particle velocity $\mathbf{v}(t,f)$. Reactive intensity I_r is an imaginary part of acoustic intensity that represents a dissipative local energy transfer. It is a physical quantity dominated by direct sound from a single source. We expect to obtain spatial decomposed information and phase information from acoustic intensities of 6 channels of mel-band acoustic intensity.

Table 1: Input features for single networks

Name	Feature configuration
MIC	8 channels, magnitude and phase spectrogram (Mic)
FOA	8 channels, magnitude and phase spectrogram (Foa)
Log-Mel	4 channels, log mel-band energy (Mic)
Log-Mel + I_a	7 channels, log mel-band energy (Mic) + mel-filtered active intensity
Log-Mel + I_a + I_r	10 channels, log mel-band energy (Mic) + mel-filtered active intensity + mel-filtered reactive intensity

$$\mathbf{I}(t,f) = p(t,f)\mathbf{v}^*(t,f) = -W(t,f)\begin{bmatrix} X^*(t,f) \\ Y^*(t,f) \\ Z^*(t,f) \end{bmatrix} \tag{1}$$

Finally, it is important that the size of the acoustic intensity feature and the size of mel-band energy feature are equal to deal with those features in the single network. Therefore, mel filter is applied to resize acoustic intensity features.

4. NETWORK ARCHITECTURE

4.1. CNN Layers

In Figure 1(a), the CNN layers consist of two gated linear unit (GLU) blocks and global average pooling that compresses the frequency (mel-bin) axis. Both the SED network and the DOA network use the same CNN layers. The details of the GLU block are shown in Figure 1(b).

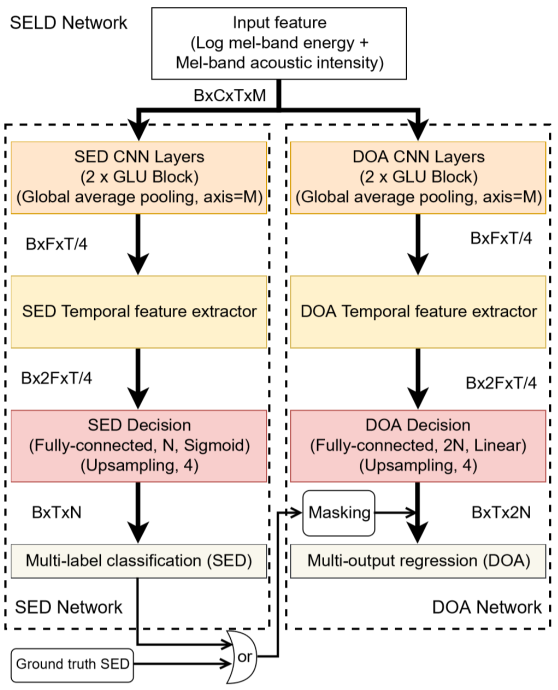
4.2. Temporal feature extractor

There are two different types of temporal feature extractors for the proposed model. One is Bidirectional-GRU, one of RNNs. The other is Bidirectional-TrellisNet which is a special form of TCNs. The details of the RNN block and the TrellisNet block are shown in Figure 1(b).

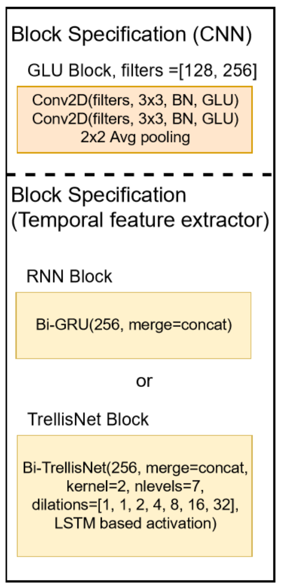
TrellisNet [9] tried to combine CNNs and RNNs through direct input injection and weight sharing among TCN layers. The key idea of TrellisNet is to implement a CNN that behaves like an RNN under certain conditions. When RNN is unfolded, a new input comes in every step and the same weight is applied. In TrellisNet, the temporal convolution layer with kernel size 2 corresponds to one RNN step. TrellisNet can replace the recurrent structure of the RNNs by stacking of multiple temporal convolution layers and adding the input injection and weight sharing techniques. Therefore TrellisNet can take advantage of both structural and algorithmic elements of CNN and RNN. In TrellisNet, LSTM is applied between each temporal convolution layer. We set the receptive field of TrellisNet to cover the input frame length for performance comparison with RNN.

4.3. Reassembly learning

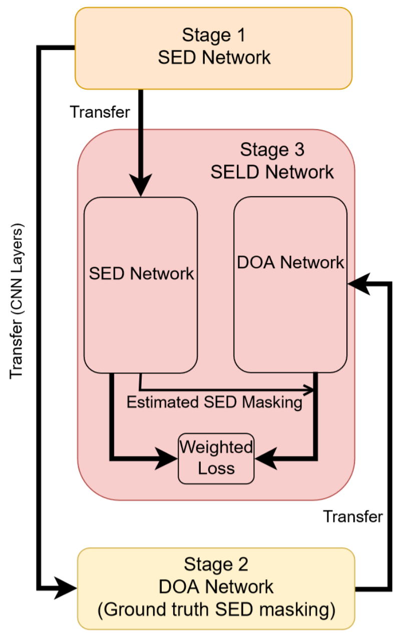
We propose a reassembly learning to design a single network for a multi-task problem which consists of dependent sub-tasks. Re-


SELD Network
Input feature (Log mel-band energy + Mel-band acoustic intensity)
BxCxTxM
SED CNN Layers (2 x GLU Block) (Global average pooling, axis=M)
DOA CNN Layers (2 x GLU Block) (Global average pooling, axis=M)
BxFxT/4
SED Temporal feature extractor
DOA Temporal feature extractor
Bx2FxT/4
SED Decision (Fully-connected, N, Sigmoid) (Upsampling, 4)
DOA Decision (Fully-connected, 2N, Linear) (Upsampling, 4)
BxTxN
Masking
BxTx2N
Multi-label classification (SED)
Multi-output regression (DOA)
SED Network
DOA Network
Ground truth SED
or

(a) Network architecture


Block Specification (CNN)
GLU Block, filters =[128, 256]
Conv2D(filters, 3x3, BN, GLU)
Conv2D(filters, 3x3, BN, GLU)
2x2 Avg pooling
Block Specification (Temporal feature extractor)
RNN Block
Bi-GRU(256, merge=concat)
or
TrellisNet Block
Bi-TrellisNet(256, merge=concat, kernel=2, nlevels=7, dilations=[1, 1, 2, 4, 8, 16, 32], LSTM based activation)

(b) Block specification



(c) Process of reassembly learning

Figure 1: Proposed system for DCASE2019 task 3; B: batch size, C: channel, T: time, M: number of mel-bin, F: filters, N: number of classes

Table 2: Hyper-parameters for proposed system

Name	Value
Epoch for SED network	25
Epoch for DOA network	100
Epoch for SELD network	10
Learning rate for SED/DOA network	0.001
Learning rate for SELD network	0.0001
Weight for SELD network	SED:DOA = 0.2:0.8
Batch size (B)	32
Frame length (T)	200
Step size between two segment	100
Number of Mel-bin (M)	96
Number of the FFT	1024

assembly learning consists of three stages. The first stage is training the SED network. After then CNN layers of the trained SED network are transferred to the DOA network. In the second stage, the loss of the DOA network is calculated with masking inactive event by the ground truth SED labels. The last stage is the SELD stage. The SELD network initializes the whole parameter from the pre-trained SED network and the DOA network as shown in Figure 1(c). At this stage, we use estimated SED for masking instead of ground truth SED labels.

As mentioned in the introduction, using the ground truth SED label masking causes an irony that DOAE for mispredicted events is made with random DOA predictions excluded from training. Reassembly learning is a technique to make the random DOA value of the mispredicted event closer to the default value through additional training. In short, the reassembly learning is a learning method that reduces randomness through additional fine-tuning. The trained SED network has more than 98% F-score for the training dataset. Therefore, reassembly learning plays a role in adjusting less than 2% outliers that are not detected correctly.

5. EVALUATION RESULT

In this section, we will describe network architectures using the proposed technique and network layers in the previous section and its performance. The hyper-parameters for training are summarized in Table 2.

5.1. Single network

Table 3 shows the experimental results for single networks by using pre-defined four-fold cross-validation split for DCASE 2019. We combined training and validation split for training proposed models. The system name and its description stand for following,

- **Baseline**: Baseline network model for DCASE 2019 challenge task 3 by task organizers
- **Reassembly 'SED-DOA'**: Proposed architecture as shown in Figure 1(a); 'SED-DOA' specifies the network that corresponds to the temporal feature extractor in SED and DOA networks. Candidates for temporal feature extractor are RNN and TrellisNet. For example, RNN-TrellisNet means RNN block used as SED temporal feature extractor and TrellisNet block used as DOA temporal feature extractor
- **SELD RNN-TrellisNet**: Proposed network structure without reassembly learning
- **Two-stage RNN-TrellisNet**: Proposed network structure with two-stage learning
- **Reassembly v1**: 4 GLU block for CNN layers of proposed model; DCASE challenge submission model

Table 3: Experimental results for DCASE 2019 task 3 development dataset; ER: error rate, F: F-score, DOA: DOA error, FR: frame recall

System	Feature	ER	F	DOA	FR
Baseline	FOA	34	79.9	28.5	85.4
Baseline	MIC	35	80.0	30.8	84.0
Reassembly RNN-TrellisNet	Log-Mel	16	90.9	10.2	88.1
Reassembly RNN-TrellisNet	Log-Mel + I_a	15	91.4	7.6	88.2
Reassembly RNN-TrellisNet	Log-Mel + I_a + I_r	**15**	**91.4**	**6.4**	**88.4**
Reassembly RNN-RNN	Log-Mel + I_a + I_r	15	91.4	8.8	88.4
Reassembly TrellisNet-RNN	Log-Mel + I_a + I_r	30	84.5	10.4	86.6
Reassembly TrellisNet-TrellisNet	Log-Mel + I_a + I_r	29	84.6	7.0	86.6
SELD RNN-TrellisNet with ground truth masking	Log-Mel + I_a + I_r	18	89.7	10.3	87.5
SELD RNN-TrellisNet with estimated SED masking	Log-Mel + I_a + I_r	19	89.3	9.4	86.9
Two-stage RNN-TrellisNet	Log-Mel + I_a + I_r	15	91.4	6.7	88.4
Reassembly v1 (Development dataset)	Log-Mel + I_a + I_r	16	90.6	6.4	85.7
Reassembly v1 (Evaluation dataset)	Log-Mel + I_a + I_r	15	91.9	5.1	87.4
Avg ensemble (SED + DOA), Reassembly RNN-TrellisNet	Log-Mel, Log-Mel + I_a, Log-Mel + I_a + I_r	13	92.7	7.7	88.9
Avg ensemble (SED), Reassembly RNN-TrellisNet	Log-Mel, Log-Mel + I_a, Log-Mel + I_a + I_r	**13**	**92.7**	**6.4**	**88.9**

Table 3 shows the results of applying three different input features to Reassembly RNN-TrellisNet. As a result, the higher the dimension of the input feature used, the higher the DOA result. These results show that spatial information and phase information from FOA were important for DOAE. On the other hand, SED results showed no significant change. This means that the log mel-band energy feature has been used primarily for SED. While the intensity vector feature does not help improve SED performance. This is because the direction does not need to be considered for SED.

SED is a problem that infers time-varying patterns. While DOAE for static events is a problem for estimating static phase difference. Therefore, we assumed that RNN would be advantageous in inferring the time-varying pattern for SED. On the other hand, we assumed that CNN would be more appropriate than RNN for estimating the static phase difference. These assumptions are proved in the results of Table 3. RNN is strong for SED and TrellisNet has strong point for DOAE. This result is that the TCN based network has an advantage in DOAE and is the possibility of being applied to a variety of sound classification and detection applications.

Reassembly RNN-TrellisNet system using Log-Mel + I_a + I_r is the best performance in a single model in Table 3. We submitted a single network, Reassembly v1, using 4 GLU blocks on CNN layers for Reassembly RNN-TrellisNet to DCASE 2019 challenge task 3. Reassembly v1 ranked 10th in DCASE 2019 challenge task 3 challenge. The model proposed in this paper has slightly changed the DCASE challenge submission model. By reducing the number of GLU blocks, the time pooling is reduced. So it brings 1%, 1%, and 3% performance gain for error rate, F-score and frame recall respectively.

The proposed system has achieved performance improvement over DCASE 2019 baseline. Compared to the performance of the proposed network without reassembly learning, pre-training subtask networks has proven to significantly improve the performance of all metrics. In Table 3 the reassembly learning showed a 0.4° improvement in DOA error compared to the two-stage learning. Reassembly learning has led to a small improvement since it plays a role in reducing randomness for mispredicted outliers.

5.2. Ensemble network

We use the simple and powerful ensemble method, average ensemble, to Reassembly RNN-TrellisNet model of the three different input features used in Table 3. Following is descriptions of the ensemble systems:

- **Avg ensemble (SED + DOA)**: Average ensemble for both SED and DOA prediction results of Reassembly RNN-TrellisNet.
- **Avg ensemble (SED)**: Average ensemble for SED prediction results of Reassembly RNN-TrellisNet + DOA prediction results from Log-Mel + I_a + I_r feature.

Table 3 shows that the Avg ensemble (SED) has better performance than the Avg ensemble (SED + DOA). In the case of SED, the performance improvement was achieved by using the average value of probability used for classification. However, the DOA value is directly obtained from the regression, so the performance of the average ensemble for DOAE is almost equal to the average error of the three systems. The overall performance was improved by using the average ensemble. For the development dataset, the Avg ensemble network makes 3%, 2%, and 3% performance gain for error rate, F-score, and frame recall compared to Reassembly v1.

6. CONCLUSION

We proposed reassembly learning to solve the sound event localization and detection problem which consists of two dependent subtasks. Reassembly learning is a way to retrain network that consists of pre-trained sub-task networks. Through reassembly learning, we tried to alleviate the problem of multi regression loss used for continuous polyphonic SELD. As a result, the proposed models significantly improved both SED and DOAE performance compared to the baseline. Also, we proved that the log mel-band energy and mel-band intensity are helpful input features for SED and DOAE. Moreover, the DOAE network using TrellisNet showed better performance than CRNN. Thus TCN based architecture demonstrated the possibility for other sound classification and detection applications.

7. ACKNOWLEDGMENT

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2017-0-00050, Development of Human Enhancement Technology for auditory and muscle support)

8. REFERENCES

[1] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, Mar 2019.

[2] S. Adavanne, A. Politis, and T. Virtanen, "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network," in *26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 1462–1466.

[3] T. Hirvonen, "Classification of spatial audio location and content using convolutional neural networks," in *Proc. Audio Eng. Soc. Conv. 138*, May 2015.

[4] Y. Cao, Q. Kong, T. Iqbal, F. An, W. Wang, and M. D. Plumbley, "Polyphonic sound event detection and localization using a two-stage strategy," 2019. [Online]. Available: http://arxiv.org/abs/1905.00268

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov 1997.

[6] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *NIPS 2014 Workshop on Deep Learning*, 2014.

[7] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," 2016. [Online]. Available: https://arxiv.org/abs/1609.03499

[8] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018. [Online]. Available: http://arxiv.org/abs/1803.01271

[9] ——, "Trellis networks for sequence modeling," in *International Conference on Learning Representation (ICLR)*, 2019.

[10] L. JiaKai, "Mean teacher convolution system for DCASE 2018 task 4," DCASE2018 Challenge, Tech. Rep., Sep 2018.

[11] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018, pp. 19–23.

[12] V. Morfi and D. Stowell, "Data-efficient weakly supervised learning for low-resource audio event detection using deep learning," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018, pp. 123–127.

[13] W. Lim, S. Suh, and Y. Jeong, "Weakly labeled semi-supervised sound event detection using crnn with inception module," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018, pp. 74–77.

[14] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Iterative knowledge distillation in R-CNNs for weakly-labeled semi-supervised sound event detection," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018, pp. 173–177.

[15] R. Harb and F. Pernkopf, "Sound event detection using weakly labelled semi-supervised data with gcrnns, vat and self-adaptive label refinement," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018, pp. 83–87.

[16] Y. Guo, M. Xu, i. Wu, Y. Wang, and K. Hoashi, "Multi-scale convolutional recurrent neural network with ensemble method for weakly labeled sound event detection," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018, pp. 98–102.

[17] L. Perotin, R. Serizel, E. Vincent, and A. Guerin, "CRNN-based multiple DOA estimation using acoustic intensity features for ambisonics recordings," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 22–33, Mar 2019.

WEAKLY LABELED SOUND EVENT DETECTION USING TRI-TRAINING AND ADVERSARIAL LEARNING

Hyoungwoo Park, Sungrack Yun, Jungyun Eum, Janghoon Cho, Kyuwoong Hwang

Qualcomm AI Research*, Qualcomm Korea YH
{c_hyoupa, sungrack, c_jeum, janghoon, kyuwoong}@qti.qualcomm.com

ABSTRACT

This paper considers a semi-supervised learning framework for weakly labeled polyphonic sound event detection problems for the DCASE 2019 challenge's task4 by combining both the tri-training and adversarial learning. The goal of the task4 is to detect onsets and offsets of multiple sound events in a single audio clip. The entire dataset consists of the synthetic data with a strong label (sound event labels with boundaries) and real data with weakly labeled (sound event labels) and unlabeled dataset. Given this dataset, we apply the tri-training where two different classifiers are used to obtain pseudo labels on the weakly labeled and unlabeled dataset, and the final classifier is trained using the strongly labeled dataset and weakly/unlabeled dataset with pseudo labels. Also, we apply the adversarial learning to reduce the domain gap between the real and synthetic dataset. We evaluated our learning framework using the validation set of the task4 dataset, and in the experiments, our learning framework shows a considerable performance improvement over the baseline model.

Index Terms— Sound event detection (SED), Tri-training, Pseudo labeling, Adversarial learning, Semi-supervised learning, Weakly supervised learning

1. INTRODUCTION

The polyphonic sound event detection (SED) has been attracting growing attention in the field of acoustic signal processing [1–8]. The SED aims to detect multiple sound events happened simultaneously as well as the time frame in a sequence of audio events. The applications of the SED include audio event classification [9–11], media retrieval [12, 13] and automatic surveillance [11] in living environments such as Google Nest Cam [14] which analyzes the audio stream to detect conspicuous sounds such as window breaking and dog barking among various sounds that could occur in daily environments.

Several researches [2, 7, 8, 10, 15–18] have been previously proposed . In [4], spectral domain features are used to characterize the audio events, and deep neural networks (DNN) [10] is used to learn a mapping between the features and sound events. In [18], multiple instance learning was exploited to predict the labels of new, unseen instances which rely on an ensemble of instances, rather than individual instances. In [5], convolutional recurrent neural networks (CRNN) was introduced which is a combined network of convolutional neural networks (CNN) [15, 16] and recurrent neural networks (RNN) [8] to get the benefits of both CNN and RNN. In [19], Mean Teacher method was adopted where the teacher model is an average of consecutive student models to overcome the limitations of temporal ensembling for semi-supervised learning.

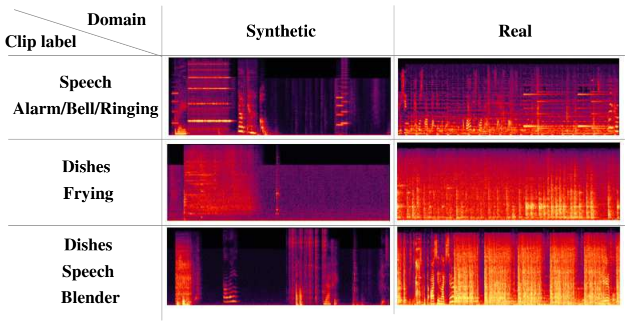


Figure 1: The spectrograms of synthetic and real dataset samples which have the same clip labels

In contrast to the task 4 of the last year's challenge [20], a synthetic dataset with strong annotation is additionally provided in DCASE 2019 challenge's task 4. Strong annotation includes onset, offset and class label of the sound events. Thus, how to utilize strongly-labeled synthetic data and mutual complement between real dataset and synthetic dataset is a challenging problem in weakly labeled SED problem in DCASE 2019 challenge's task 4. Previous methods have not focused on complement of strongly labeled synthetic dataeset. As shown in Figure 1, the log-mel spectrogram of samples which have the same clip label from synthetic data and real data seem too much different. For this reason, we assume that domain gap between synthetic data and real data exists and it causes degradation of performance on test samples.

This paper presents a sound event detection combining adversarial learning and tri-training. Adversarial learning helps to reduce the gap between synthetic and real data by learning domain-invariant feature while tri-training method [21] which is one of the semi-superived learning methods learns discriminative representations by pseudo labeling one the weakly labeled or unlabeled samples. Pseudo labels are obtained by agreement of output from confident two labelers on unlable data. Inspired by these properties, we present a weakly labeled polyphonic SED by considering both adversarial learning and tri-training.

The proposed learning framework was evaluated using a validation set of the DCASE 2019 challenge's task 4 [22]. In the evaluation results, combined adversarial training and tri-training shows a considerable performance improvement over the baseline model.

*Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

https://doi.org/10.33682/ntr9-6764

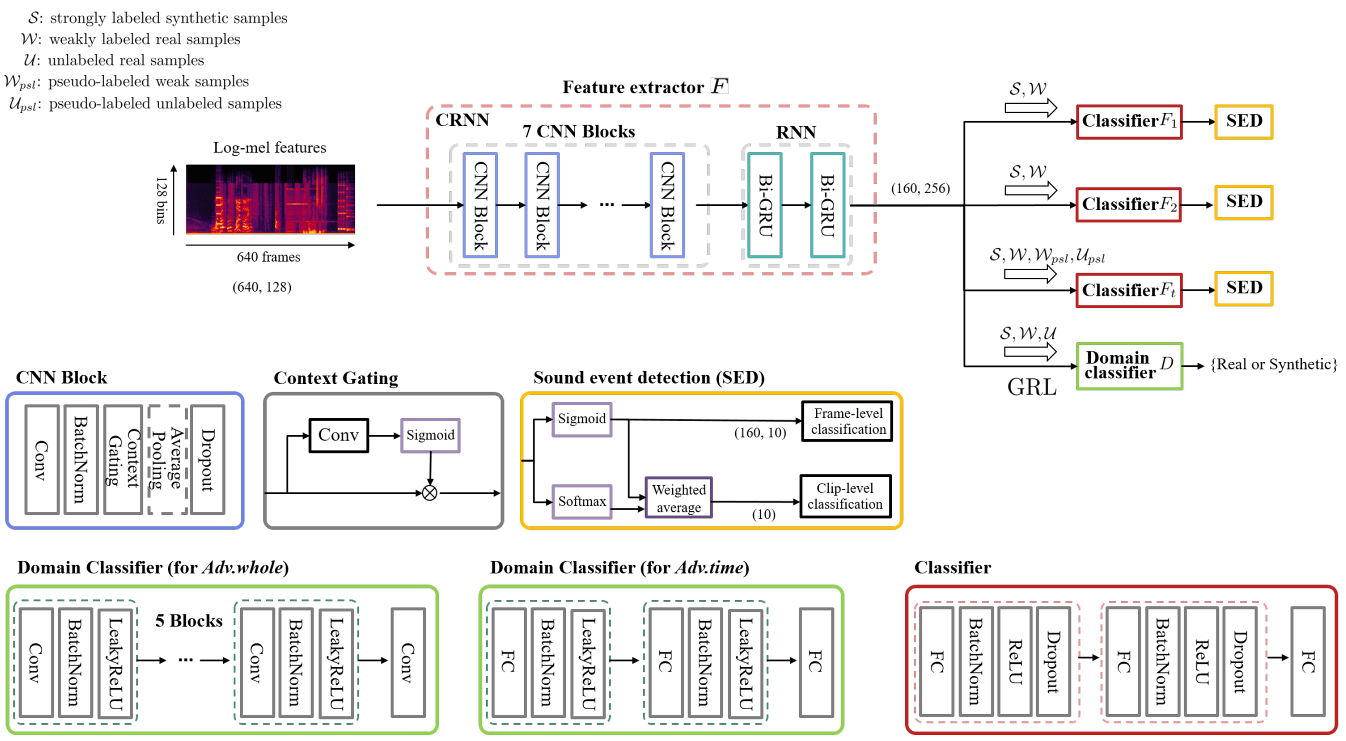

$\mathcal{S}$: strongly labeled synthetic samples
$\mathcal{W}$: weakly labeled real samples
$\mathcal{U}$: unlabeled real samples
$\mathcal{W}_{psl}$: pseudo-labeled weak samples
$\mathcal{U}_{psl}$: pseudo-labeled unlabeled samples
Log-mel features
128 bins
640 frames
(640, 128)
Feature extractor F
CRNN
7 CNN Blocks
CNN Block
RNN
Bi-GRU
(160, 256)
$\mathcal{S},\mathcal{W}$
Classifier F_1
SED
Classifier F_2
$\mathcal{S},\mathcal{W},\mathcal{W}_{psl},\mathcal{U}_{psl}$
Classifier F_t
$\mathcal{S},\mathcal{W},\mathcal{U}$
Domain classifier D
{Real or Synthetic}
GRL
CNN Block
Conv
BatchNorm
Context Gating
Average Pooling
Dropout
Context Gating
Sigmoid
Sound event detection (SED)
Softmax
Weighted average
Frame-level classification
(160, 10)
Clip-level classification
(10)
Domain Classifier (for *Adv.whole*)
5 Blocks
LeakyReLU
Domain Classifier (for *Adv.time*)
FC
Classifier
ReLU

Figure 2: The proposed learning framework includes feature extractor F, classifiers(pseudo-labelers) F_1, F_2, final classifier F_t and domain classifier D. The dataset to train each component is shown in the figure (e.g. classifier F_2 is trained using the strongly-labeled synthetic samples $\mathcal{S}$ and weakly-labeled real samples $\mathcal{W}$. The pseudo-labels are obtained by agreement from two different classifiers F_1, F_2 and used in training the final classifier F_t. The domain classifier D, connected to F via a GRL, classifies the input feature into real or synthetic. With the GRL from D to F, the feature distributions between synthetic and real domain become similar, and thus we can obtain the domain-invariant features.

2. PROBLEM STATEMENT AND NOTATIONS

For SED, we denote a sound clip by $x \in \mathcal{X}$ and corresponding $y \in \mathcal{Y}$. The SED systems are expected to produce strongly labeled output y^s (i.e. sound class label with start time and end time) from input x. However, for weakly labeled SED with semi-supervised setting, dataset consists of strongly labeled data $\mathcal{S} = \{(x_i^s, y_i^s)\}_{i=1}^m$, weakly labeled data $\mathcal{W} = \{(x_j^w, y_j^w)\}_{j=1}^n$ and unlabeled data $\mathcal{U} = \{x_k^u\}_{k=1}^l$. The weakly labeled data does not provide a temporal range of events but sound class labels detected in a clip. We focus on the usage of weakly labeled or unlabeled data and reducing domain gap between synthetic and real data. Thus, we combine the adversarial learning based on gradient reversal layer (GRL) [23] for reducing the domain gap and tri-training method for pseudo-labeling weakly labeled or unlabeled data such that the networks are learned to output discriminative representations on a real dataset.

3. PROPOSED METHOD

Our proposed method is based on *CRNN* [19] model, which showed the first place of the task 4 in the last year's challenge by combining with Mean Teacher algorithm [24]. The whole architecture is shown in Figure 2. A feature extractor F, which cosists of seven CNN blocks and two bi-directional gated recurrent units (Bi-GRU) [25], outputs shared features from log-mel features used as input for four networks. Two labelers F_1, F_2 and classifier F_t predict multiple classes for each time frame and class events for a clip from features extracted by F. Let L_y be the classification loss with frame-level classification loss L_{frame} and clip-level classification loss L_{clip} for multi-label prediction.

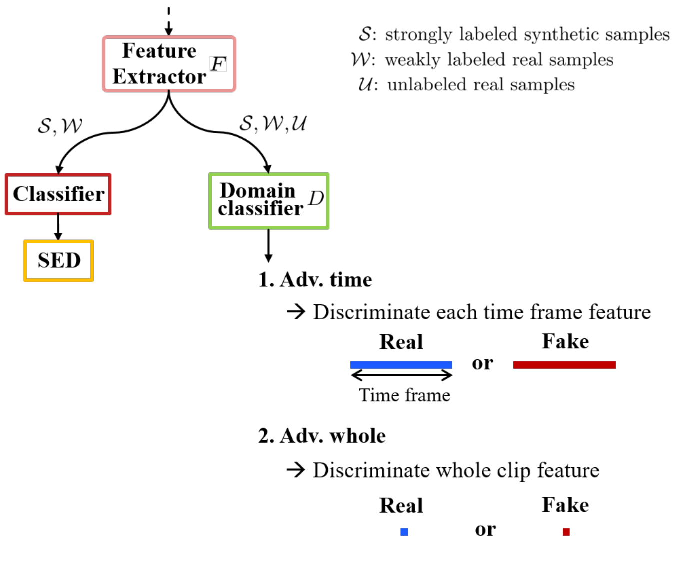
$$L_y = L_{frame} + L_{clip} \tag{1}$$

For training with frame-level classification loss and clip-level classfication loss, binary cross entropy (BCE) loss is used with the sigmoid output:

$$L_{clip} = \sum_{i=1}^{N}\sum_{k=1}^{K}[y_{i,k}\log\hat{y}_{i,k} + (1-y_{i,k})\log(1-\hat{y}_{i,k})] \tag{2}$$

$$L_{frame} = \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K}[y_{i,t,k}\log\hat{y}_{i,t,k}+(1-y_{i,t,k})\log(1-\hat{y}_{i,t,k})] \tag{3}$$

where $y_{i,k}, y_{i,t,k}^i \in [0,1]$ are the label of sound class k of clip i and the label sound class k at time frame t of clip i, respectively. Also, $\hat{y}_{i,k}$ is the predicted probability of sound class k of clip i, and $\hat{y}_{i,t,k}$ is $\hat{y}_{i,k}$ at time frame t. A domain classifier D classifies features from F into real or synthetic. Based on this architecture, the proposed adversarial learning with tri-training framework for SED will be explained in the next section.



$\mathcal{S}$: strongly labeled synthetic samples
$\mathcal{W}$: weakly labeled real samples
$\mathcal{U}$: unlabeled real samples
Feature Extractor F
$\mathcal{S},\mathcal{W}$
$\mathcal{S},\mathcal{W},\mathcal{U}$
Classifier
Domain classifier D
SED
1. Adv. time
→ Discriminate each time frame feature
Real or Fake
Time frame
2. Adv. whole
→ Discriminate whole clip feature
Real or Fake

Figure 3: Two approaches of adversarial learning for sound event detection problems

```
Algorithm 1: The function Pseudo-labeling is the process of assigning pseudo-labeling based on agreement threshold from two labelers. We assign pseudo-labels to weakly labeled or unlabeled samples when both predictions of $F_1$ and $F_2$ are confident and agreed to the same prediction.

Input: strongly labeled synthetic data $\mathcal{S} = \{(x_i^s, y_i^s)\}_{i=1}^m$
weakly labeled real data $\mathcal{W} = \{(x_j^w, y_j^w)\}_{j=1}^n$
unlabeled real data $\mathcal{U} = \{(x_k^u)\}_{k=1}^l$
pseudo-labeled weakly labeled data $\mathcal{W}_{psl} = \emptyset$
pseudo-labeled unlabeled data $\mathcal{U}_{psl} = \emptyset$
for i = 1 to iter do
    Train $F, F_1, F_2, F_t, D$ with mini-batch from labeled training set $\mathcal{S}, \mathcal{W}, \mathcal{U}$
end
$\mathcal{W}_{psl}$ = Pseudo-labeling($F, F_1, F_2, \mathcal{W}$)
$\mathcal{U}_{psl}$ = Pseudo-labeling($F, F_1, F_2, \mathcal{U}$)
for j =1 to iter do
    Train $F, F_t, D$ with mini-batch from labeled training set $\mathcal{S}, \mathcal{W}$ and pseudo-labeled training set $\mathcal{W}_{psl}, \mathcal{U}_{psl}$
end
```

3.1. Adversarial learning

We denote strongly labeled synthetic dataset by $\mathcal{S}$ and weakly labeled or unlabeled real dataset $\mathcal{W}, \mathcal{U}$ are the different domain (synthetic or real). As shown in Figure 1, since the domain gap between the synthetic and real dataset is quite big, we construct a domain classifier to reduce the gap between two domains by adversarial learning. The domain classifier D classifies input features into real or synthetic. By applying the GRL [23] from D to the feature extractor F, we can obtain the feature representation whose distributions are almost similar in both real and synthetic domain. We consider two approaches to apply the adversarial learning for SED as shown in Figure 3. First, D classifies the whole feature from F into one result: real or synthetic (*Adv.whole*). In this case, GRL makes the features from F domain-invariant. Second, D classifies each time frame of the feature into real or synthetic (*Adv.time*). The second approach is more appropriate than the first one since our architecture predicts multiple sound event classes in each time frame from features extracted from F. We denote $\theta_F, \theta_{F_1}, \theta_{F_2}, \theta_{F_t}$ and θ_D by the parameters of each network, respectively. Also, L_d is the loss for the domain classification. For training with domain classfication loss, BCE loss is used with the sigmoid output:

$$L_d = \sum_{i=1}^{N} [d_{i,t} \log \hat{d}_{i,t} + (1 - d_{i,t}) \log (1 - \hat{d}_{i,t})] \tag{4}$$

where $d_{i,t}$ is the label of real or synthetic at time frame t of clip i, and $\hat{d}_{i,t}$ is predicted probability at time frame t of clip i. Based on GRL, the parameters are updated as follows:

$$\theta_F \leftarrow \theta_F - \mu(\frac{\partial L_y}{\partial \theta_F} - \alpha \frac{\partial L_d}{\partial \theta_F}) \tag{5}$$

$$\theta_{F_1,F_2,F_t} \leftarrow \theta_{F_1,F_2,F_t} - \mu \frac{\partial L_y}{\partial \theta_{F_1,F_2,F_t}} \tag{6}$$

$$\theta_D \leftarrow \theta_D - \mu \frac{\partial L_d}{\partial \theta_D} \tag{7}$$

where μ, α are the learning rate and hyperparameter of GRL, respectively.

3.2. Tri-training

We apply the tri-training method to train a network using the pseudo-labeled weakly labeled samples $\mathcal{W}_{psl}$ and pseudo-labeled unlabeled samples $\mathcal{U}_{psl}$. The entire procedure of tri-training is shown in Algorithm 1. First, we train common feature extractor F, two labeling networks F_1 and F_2 , a final classifier F_t and a domain classifier D with labeled samples $\mathcal{S}, \mathcal{W}$ and unlabeled samples $\mathcal{U}$. Second, pseudo-labeled samples are obtained by F_1 and F_2 trained with labeled samples. When the confidences of both networks' outputs exceed the agreement threshold, the prediction can be considered reliable. We set this threshold to 0.5 in the experiments. Also, we expect each labeler to obtain different classifiers F_1 and F_2 given the same training data, we use the following regularization loss:

$$L = L_y + \lambda \left| \left(\frac{W_{F_1}}{|W_{F_1}|} \right)^\top \left(\frac{W_{F_2}}{|W_{F_2}|} \right) \right| \tag{8}$$

where W_{F_1} and W_{F_2} are weights of first layer of two labelers F_1 and F_2, respectively. We set λ to 1.0 based on the validation set. Then, we use both labeled samples $\mathcal{S}, \mathcal{W}$ and pseudo-labeled samples $\mathcal{W}_{psl}, \mathcal{U}_{psl}$ for training F, F_t, and D. Then, F and F_t will learn from the labeled real dataset.

4. EXPERIMENTS

4.1. Dataset

The DCASE 2019 challenge's task 4 [22] provides the following 3 subsets of the dataset in the training: 1,578 clips of the weakly labeled set, 14,412 clips of the unlabeled in-domain set and 2,045 clips of the synthetic set with strong annotations of events and timestamps. Weakly labeled and unlabeled in-domain sets are from

Audioset [26] which drawn from 2 million YouTube videos. The synthetic set is generated with Scaper [27] to increase the variability of the output for soundscape synthesis and augmentation. These audio clips are 10 second-long and contain one or multiple sound events among 10 different classes (speech, dog, cat, alarm/bell/ringing, dishes, frying, blender, running water, vacuum cleaner and electric shaver/toothbrush) which may partly overlap.

4.2. Experimental setup

The model was developed using PyTorch [28] and all experiments were conducted on an a GeForce GTX TITAN X GPU 12GB RAM. Also, our architecture was trained with a mini-batch size of 64 using Adam optimizer [29] with an initial learning rate of 0.001 and exponential decay rate for the $1st$ and $2nd$ moments of 0.9 and 0.999, respectively. The input audio clips are down-sampled from 44.10 kHz to 22.05 kHz. And, the log-mel spectrogram is extracted from the audio clip with the size of 640×128: 128-bin is used, and 2048-window with 345-hop is used to convert into 640 frames.

4.3. Experimental results

We evaluated our proposed framework using the DCASE 2019 challenge's task4 validation dataset. We could not measure performance on evaluation dataset since the labels of evaluation dataset are not available yet. The macro event-based F1 scores and segment-based F1 scores on validation dataset are shown in Table 1. Segment-based metrics evaluate an active/inactive state for each sound event in a fixed-length interval, while event-based metrics evaluate sound event class detected in the fixed-length interval. The baseline of DCASE 2019 challenge's task 4 used the Top-1 ranked model [19] of DCASE 2018 challenge's task 4, which proposed Mean Teacher method for SED; however, this baseline was designed with smaller architecture. Thus, we designed our baseline model based on originally Top-1 ranked model in DCASE 2018. Our baseline consists of feature extractor F and classifier F_t which are trained using the strongly labeled synthetic data and weakly labeled real data without Mean Teacher algorithm. The baseline showed 24.15% event-based F1 score. For the comparisons, the official results of various SED frameworks submitted for the DCASE 2019 challenge's task 4 are shown in Table 1.

With our baseline model, we applied the adversarial learning method in two ways. The domain classifier D predicted real or synthetic on whole feature map (*Adv.whole*) and on features in each time frame (*Adv.time*). Both adversarial learning approaches improved the performance as shown in Table 1. These approaches reduced domain gap between synthetic and real feature distributions, thsu we could improve performance since the validation set was also from the real audio clips. The *Adv.time* and *Adv.whole* achieved 31.33% and 30.65%, respectively. The *Adv.time* method showed better performance than the *Adv.whole* since the architecture tries to predict multi-label in each time frame. We also performed pseudo-labeling method using tri-training procedure. The tri-training method achieved 30.23%. Tri-training method showed better performance than the adversarial learning in evaluating segment-based F1 scores. Finally, we evaluated the combined architecture: adversarial learning with the tri-training. When we trained two labelers in tri-training, we also trained domain classifier simultaneously for adversarial learning. After training two labelers with adversarial learning, more confident labelers for predicting sound event classes on real dataset were obtained. Then, we assigned pseudo-label to weakly labeled and unlabeled samples based on two labelers. We trained the final classifier with labeled samples and pseudo-labeled samples by tri-training scheme. In combining the adversarial learning with tri-training method, we considered the previous two approaches: *Adv.whole* and *Adv.time*. The tri-training method combined with *Adv.whole* approach showed 32.64% event-based F1 score and the tri-training method combined with *Adv.time* achieved 35.10%. *Adv.time*+Tri-training achieved the highest event-based F1 score of our models. The tri-training method achieved 62.86% segment-based F1 score and it is better than *Adv.time*+Tri-training. We think that adversarial learning contributes more to inference exact sound label in time frame while the tri-training contributes more to inference the exact boundary of the sound event.

Table 1: The event based macro F1 scores and segment based macro F1 scores of proposed methods on validation dataset in DCASE 2019 challenge's task 4

Model	**Macro F1 (%)**	
	Event-based	Segment-based
Wang_YSU_task4_1	19.4%	-
Kong_SURREY_task4_1	21.3%	-
Wang_NUDT_task4_3	22.4%	-
DCASE 2019 baseline [22]	23.7%	55.2%
Rakowski_SRPOL_task4_1	24.3%	-
mishima_NEC_task4_4	24.7%	-
Lee_KNU_task4_3	26.7%	-
bolun_NWPU_taks4_2	31.9%	-
Kothinti_JHU_task4_1	34.6%	-
ZYL_UESTC_task4_2	35.6%	-
Kiyokawa_NEC_task4_4	36.1 %	-
PELLEGRINI_IRIT_task4_1	39.9%	-
Lim_ETRI_task4_4	40.9 %	-
Shi_FRDC_task4_2	42.5%	-
Yan_USTC_task4_4	42.6 %	-
Delphin_OL_task4_2	43.6%	-
Lin_ICT_task4_3	**45.3**%	-
Our baseline	24.15%	57.70%
Adv.whole	30.65%	59.06%
Adv.time	31.33%	59.26%
Tri-training	30.23%	**62.86**%
Adv.whole + Tri-training	32.64%	60.48%
Adv.time + Tri-training	**35.10**%	60.67%

5. CONCLUSION

In this paper, we consider the semi-supervised learning framework for weakly labeled SED problem for the DCASE 2019 challenge's task4 by combining both tri-training and adversarial learning. The entire dataset consists of the synthetic data with the strong label (sound event labels with boundaries) and real data with weakly labeled (sound event label) and unlabeled dataset. We reduce domain gap between strongly labeled synthetic dataset and weakly labeled or unlabeled real dataset to train networks to learn domain-invariant feature for preventing degradation of performance. Also, we utilize pseudo labeled samples based on confident multiple labelers trained by labeled samples. Then, networks learn the discriminative representation of the unlabeled dataset. The tri-training method combined with adversarial learning on each time frame shows a considerable performance improvement over the baseline model.

6. REFERENCES

[1] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the dcase 2016 challenge," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 26, no. 2, pp. 379–393, 2018.

[2] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," in *Proc. DCASE, 2018*, 2018.

[3] A. Mesaros, T. Heittola, O. Dikmen, and T. Virtanen, "Sound event detection in real life recordings using coupled matrix factorization of spectral representations and class activity annotations," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 151–155.

[4] S. Adavanne, P. Pertilä, and T. Virtanen, "Sound event detection using spatial features and convolutional recurrent neural network," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 771–775.

[5] E. Çakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.

[6] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," in *Proceedings of the IEEE European Signal Processing Conference (EUSIPCO)*, 2016, pp. 1128–1132.

[7] E. Cakir, T. Heittola, H. Huttunen, and T. Virtanen, "Polyphonic sound event detection using multi label deep neural networks," in *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, 2015, pp. 1–7.

[8] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 6440–6444.

[9] A. Rakotomamonjy and G. Gasso, "Histogram of gradients of time–frequency representations for audio scene classification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 23, no. 1, pp. 142–153, 2015.

[10] I. McLoughlin, H. Zhang, Z. Xie, Y. Song, and W. Xiao, "Robust sound event classification using deep neural networks," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 23, no. 3, pp. 540–552, 2015.

[11] N. Almaadeed, M. Asim, S. Al-Maadeed, A. Bouridane, and A. Beghdadi, "Automatic detection and classification of audio events for road surveillance applications," *Sensor Signal and Information Processing (SSIP)*, vol. 18, no. 6, 2018.

[12] M. Fan, W. Wang, P. Dong, L. Han, R. Wang, and G. Li, "Cross-media retrieval by learning rich semantic embeddings of multimedia," in *Proceedings of the ACM international conference on Multimedia (ACMMM)*, 2017, pp. 1698–1706.

[13] N. Rasiwasia, J. Costa Pereira, E. Coviello, G. Doyle, G. R. Lanckriet, R. Levy, and N. Vasconcelos, "A new approach to cross-modal multimedia retrieval," in *Proceedings of the ACM international conference on Multimedia (ACMMM)*, ser. MM '10, 2010, pp. 251–260.

[14] https://store.google.com/gb/product/nest_cam_outdoor.

[15] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 559–563.

[16] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6.

[17] J. Salamon and J. P. Bello, "Unsupervised feature learning for urban sound classification," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 171–175.

[18] A. Kumar and B. Raj, "Audio event detection using weakly labeled data," in *Proceedings of the ACM international conference on Multimedia (ACMMM)*, 2016, pp. 1038–1047.

[19] L. JiaKai, "Mean teacher convolution system for DCASE 2018 task 4," *Detection and Classification of Acoustic Scenes and Events*, 2018.

[20] http://dcase.community/challenge2018/.

[21] K. Saito, Y. Ushiku, and T. Harada, "Asymmetric tri-training for unsupervised domain adaptation," in *International Conference on Machine Learning*, 2017.

[22] http://dcase.community/challenge2019/.

[23] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International Conference on Machine Learning*, 2015, pp. 1180–1189.

[24] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proceedings of the IEEE International Conference on Neural Information Processing Systems (NeurIPS)*, 2017, pp. 1195–1204.

[25] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Gated feedback recurrent neural networks," in *International Conference on Machine Learning*, 2015, pp. 2067–2075.

[26] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 776–780.

[27] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 344–348.

[28] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS Autodiff Workshop*, 2017.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

A HYBRID PARAMETRIC-DEEP LEARNING APPROACH FOR SOUND EVENT LOCALIZATION AND DETECTION

Andrés Pérez-López[1,2], *Eduardo Fonseca*[1*], *Xavier Serra*[1]

[1] Music Technology Group, Universitat Pompeu Fabra, Barcelona
{andres.perez, eduardo.fonseca, xavier.serra}@upf.edu
[2] Eurecat, Centre Tecnològic de Catalunya, Barcelona

ABSTRACT

This work describes and discusses an algorithm submitted to the *Sound Event Localization and Detection* Task of DCASE2019 Challenge. The proposed methodology relies on parametric spatial audio analysis for source localization and detection, combined with a deep learning-based monophonic event classifier. The evaluation of the proposed algorithm yields overall results comparable to the baseline system. The main highlight is a reduction of the localization error on the evaluation dataset by a factor of 2.6, compared with the baseline performance.

Index Terms— SELD, parametric spatial audio, deep learning

1. INTRODUCTION

Sound Event Localization and Detection (SELD) refers to the problem of identifying, for each individual event present in a sound field, the temporal activity, spatial location, and sound class to which it belongs. SELD is a current research topic which deals with microphone array processing and sound classification, with potential applications in the fields of signal enhancement, autonomous navigation, acoustic scene description or surveillance, among others.

SELD arises from the combination of two different problems: Sound Event Detection (SED) and Direction of Arrival (DOA) estimation. The number of works in the literature which jointly address SED and DOA problems is relatively small. It is possible to classify them by the type of microphone arrays used: distributed [1, 2, 3] or near-coincident [4, 5, 6]. As mentioned in [6], the usage of near-coincident circular/spherical arrays enables the representation of the sound field in the spatial domain, using the spherical harmonic decomposition, also known as Ambisonics [7, 8]. Such spatial representation allows a flexible, device-independent comparison between methods. Furthermore, the number of commercially available ambisonic microphones has increased in recent years due to their suitability for immersive multimedia applications. Taking advantage of the compact spatial representation provided by the spherical harmonic decomposition, several methods for parametric analysis of the sound field in the ambisonic domain have been proposed [9, 10, 11, 12]. These methods ease sound field segmentation into direct and diffuse components, and further localization of the direct sounds. The advent of deep learning techniques for DOA estimation has also improved the results of traditional methods [6]. However, none of the deep learning-based DOA estimation methods explicitly exploits the spatial parametric analysis. This situation is further extended to the SELD problem, with the exception of [5], where DOAs are estimated from the *active intensity vector* [9].

*E.F. is partially supported by a Google Faculty Research Award 2018.

The motivation for the proposed methodology is two-fold. First, we would like to check whether the usage of spatial parametric analysis in the ambisonic domain can improve the performance of SELD algorithms. Second, temporal information derived by the parametric analysis could be further exploited to estimate event onsets and offsets, thus lightening the event classifier complexity; such reduction might positively impact algorithm's performance.

In what follows, we present the methodology and the architecture of the proposed system (Section 2). Then, we describe the design choices and the experimental setup (Section 3), and discuss the results in the context of DCASE2019 Challenge - Task 3 (Section 4). A summary is presented in Section 5. In order to support open access and reproducibility, all code is freely available at [13].

2. METHOD

The proposed method presents a solution for the SELD problem splitting the task into four different problems: *DOA estimation*, *association*, *beamforming* and *classification*, which will be described in the following subsections. The former three systems follow a heuristic approach—in what follows, they will be jointly referred to as the *parametric front-end*. Conversely, the *classification* system is data-driven, and will be referred to as the *deep learning back-end*. The method architecture is depicted in Figure 1.

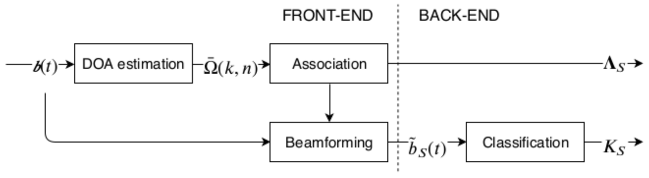


Figure 1: System architecture.

2.1. DOA estimation

The *DOA estimation* system (Figure 2) is based on parametric time-frequency (TF) spatial audio analysis. Let us consider a first-order ($L = 1$) ambisonic signal vector $\boldsymbol{b}(t)$ with N3D normalization [14]:

$$\boldsymbol{b}(t) = [b_w(t), \sqrt{3}b_x(t), \sqrt{3}b_y(t), \sqrt{3}b_z(t)]. \tag{1}$$

From its short-time frequency domain representation $\boldsymbol{B}(k,n)$, the instantaneous DOA at each TF bin $\boldsymbol{\Omega}(k,n)$ can be estimated as:

$$\begin{aligned}\boldsymbol{I}(k,n) &= -\frac{1}{Z_0}\mathbb{R}\{[B_x(k,n), B_y(k,n), B_z(k,n)]B_w(k,n)^*\},\\ \boldsymbol{\Omega}(k,n) &= [\varphi(k,n), \theta(k,n)] = \angle(-\boldsymbol{I}(k,n)),\end{aligned} \tag{2}$$

https://doi.org/10.33682/v1za-0k45

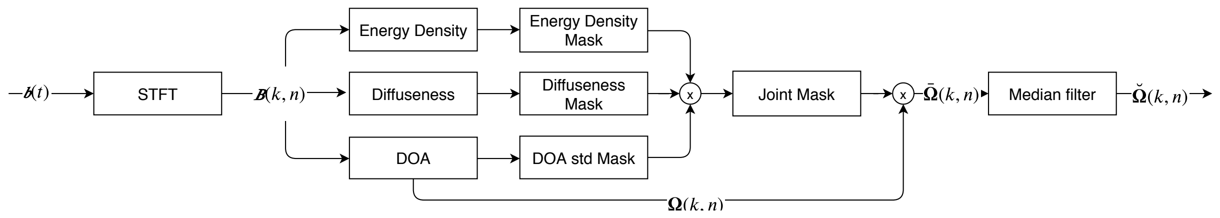


Figure 2: DOA estimation architecture.

where $\boldsymbol{I}(k, n)$ stands for the *active intensity vector* [9], Z_0 is the characteristic impedance of the medium, * represents the complex conjugate operator, and $\angle$ is the spherical coordinates angle operator, expressed in terms of azimuth φ and elevation θ.

It is desirable to identify the TF regions of $\boldsymbol{\Omega}(k, n)$ which carry information from the sound events, and discard the rest. Three binary masks are computed with that aim. The first mask is the *energy density mask*, which is used as an activity detector. The energy density $E(k, n)$ is defined as in [15] :

$$E(k,n) = \frac{|B_w(k,n)|^2 + ||[B_x(k,n), B_y(k,n), B_z(k,n)]||^2}{2Z_0 c}, \tag{3}$$

with c being the sound speed. A gaussian adaptive thresholding algorithm is then applied to $E(k, n)$, which selects TF bins with local maximum energy density, as expected from direct sounds.

The *diffuseness mask* selects the TF bins with high energy propagation. Diffuseness $\Psi(k, n)$ is defined in [16] as:

$$\Psi(k,n) = 1 - ||\,\langle\, \boldsymbol{I}(k,n)\,\rangle\,|| / (c\,\langle\, E(k,n)\,\rangle), \tag{4}$$

where $\langle\,\cdot\,\rangle$ represents the temporal expected value.

The third mask is the *DOA variance mask*. It tries to select TF regions with small standard deviation[1] with respect to their neighbor bins—a characteristic of sound fields with low diffuseness [12].

The three masks are then applied to the DOA estimation, obtaining the TF-filtered DOAs $\bar{\boldsymbol{\Omega}}(k, n)$. Finally, a median filter is applied, with the aim of improving DOA estimation consistency and removing spurious TF bins. The median filter is applied in a TF bin belonging to $\bar{\boldsymbol{\Omega}}(k, n)$ only if the number of TF bins belonging to $\bar{\boldsymbol{\Omega}}(k, n)$ in its vicinity is greater than a given threshold B_{min}. The resulting filtered DOA estimation is referred to as $\breve{\boldsymbol{\Omega}}(k, n)$.

2.2. Association

The association step (Figure 3) tackles the problem of assigning the time-frequency-space observation $\breve{\boldsymbol{\Omega}}(k, n)$ to a set of events, each one having a specific onset, offset and location. First, DOA estimates are resampled into *frames* of the task's required length (0.02 s). In what follows, frames will be represented by index m. An additional constraint is applied: for a given window n_0, the DOA estimates $\breve{\boldsymbol{\Omega}}(k, n_0)$ are assigned to the corresponding frame m_0 only if the number of estimates is greater than a threshold K_{min}.

Next, the standard deviation in azimuth (σ_φ) and elevation (σ_θ) of the frame-based DOA estimates $\breve{\boldsymbol{\Omega}}(k, m)$ are compared to a threshold value (σ_{max}), and the result is used to estimate the frame-based event overlapping amount $o(m)$:

$$o(m) = \begin{cases} 1, & \text{if } \sigma_\varphi/2 + \sigma_\theta < \sigma_{max}, \\ 2, & \text{otherwise.} \end{cases} \tag{5}$$

[1]In this work, all statistical operators for angular position refer to the 2π-*periodic* operator for azimuth, and the standard operator for elevation.

The clustered values $\boldsymbol{\Omega}_{\text{cluster}}(m)$ are then computed as the $K = o(m)$ centroids of $\breve{\boldsymbol{\Omega}}(k, m)$, using a modified version of K-Means which minimizes the central angle distance. Notice that, for $o(m) = 1$, the operation is equivalent to the median.

The following step is the grouping of clustered DOA values into events. Let us define $\boldsymbol{\Omega}_S(m)$ as the frame-wise DOA estimations belonging to the event S. A given clustered DOA estimation $\boldsymbol{\Omega}_{\text{cluster}}(m)$ belongs to the event S if the following criteria are met:

- The central angle between $\boldsymbol{\Omega}_{\text{cluster}}(m)$ and the median of $\boldsymbol{\Omega}_S(m)$ is smaller than a given threshold d_{max}^{ANGLE}, and
- The frame distance between M and the closest frame of $\boldsymbol{\Omega}_S(m)$ is smaller than a given threshold d_{max}^{FRAME}.

The resulting DOAs $\boldsymbol{\Omega}_S(m)$ are subject to a postprocessing step with the purpose of delaying event onsets in frames where $o(m) > 2$, and discarding events shorter than a given minimum length. Finally, the frame-based event estimations are converted into *metadata annotations* in the form $\boldsymbol{\Lambda}_S = (\boldsymbol{\Omega}_S, \text{onset}_S, \text{offset}_S)$.

2.3. Beamforming

The last step performed in the front-end is the input signal segmentation. The spatial and temporal information provided by the annotations $\boldsymbol{\Lambda}_S$ are used to produce monophonic signal estimations of the events, $\tilde{b}_S(t)$, as the signals captured by a virtual hypercardioid:

$$\tilde{b}_S(t) = \boldsymbol{Y}(\boldsymbol{\Omega}_S)\boldsymbol{b}^{\mathsf{T}}(t), \tag{6}$$

where $\boldsymbol{Y}(\boldsymbol{\Omega}_S) = [Y_w(\boldsymbol{\Omega}_S), Y_x(\boldsymbol{\Omega}_S), Y_y(\boldsymbol{\Omega}_S), Y_z(\boldsymbol{\Omega}_S)]$ is the set of real-valued spherical harmonics up to order $L = 1$ evaluated at $\boldsymbol{\Omega}_S$.

2.4. Deep learning classification back-end

The parametric front-end performs DOA estimation, temporal activity detection and time/space segmentation, and produces monophonic estimations of the events, $\tilde{b}_S(t)$. Then, the back-end classifies the resulting signals as belonging to one of a target set of 11 classes. Therefore, the multi-task nature of the front-end allows us to define the back-end classification task as a simple multi-class problem, even though the original SELD task is multi-label. It must be noted, however, that due to the limited directivity of the first-order beamformer, the resulting monophonic signals can present a certain leakage from additional sound sources when two events overlap, even when the annotations $\boldsymbol{\Lambda}_S$ are perfectly estimated.

The classification method is divided into two stages. First, the incoming signal is transformed into the log-mel spectrogram and split into TF patches. Then, the TF patches are fed into a single-mode based on a Convolutional Recurrent Neural Network (CRNN), which outputs probabilities for event classes $k \in \{1...K\}$, with $K = 11$. Predictions are done at the event-level (not at the frame level), since the temporal activities have been already determined by the front-end.

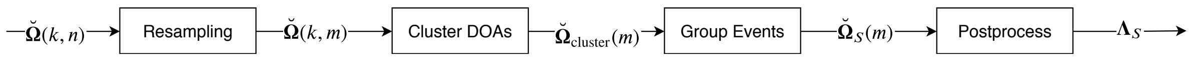


Figure 3: Association architecture.

The proposed CRNN is depicted in Figure 4. It presents three convolutional *blocks* to extract local features from the input representation. Each convolutional block consists of one convolutional layer, after which the resulting feature maps are passed through a ReLU non-linearity [17]. This is followed by a max-pooling operation to downsample the feature maps and add invariance along the frequency dimension. The target classes vary to a large extent in terms of their temporal dynamics, with some of them being rather impulsive (e.g., *Door slam*), while others being more stationary (e.g., *Phone ringing*). Therefore, after stacking the feature maps resulting from the convolutional blocks, this representation is fed into one bidirectional recurrent layer in order to model discriminative temporal structures. The recurrent layer is followed by a Fully Connected (FC) layer, and finally a 11-way softmax classifier layer produces the event-level probabilities. Dropout is applied extensively. The loss function used is categorical cross-entropy. The model has ∼175k weights.

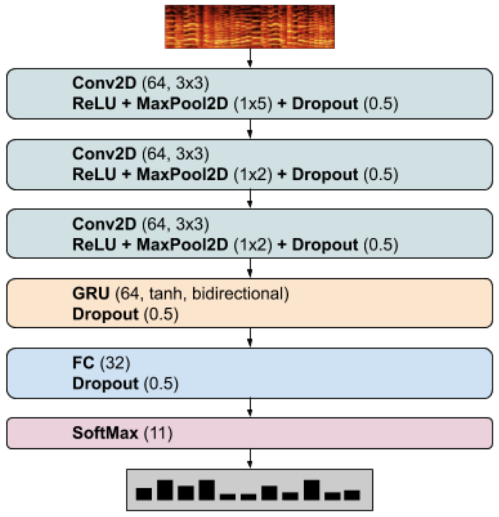

Conv2D (64, 3x3)
ReLU + MaxPool2D (1x5) **+ Dropout** (0.5)
Conv2D (64, 3x3)
ReLU + MaxPool2D (1x2) **+ Dropout** (0.5)
Conv2D (64, 3x3)
ReLU + MaxPool2D (1x2) **+ Dropout** (0.5)
GRU (64, tanh, bidirectional)
Dropout (0.5)
FC (32)
Dropout (0.5)
SoftMax (11)

Figure 4: Back-end architecture.

3. EXPERIMENTS

3.1. Dataset, evaluation metrics and baseline system

We use the TAU Spatial Sound Events 2019 - Ambisonic, which provides first-order ambisonic recordings. Details about the recording format and dataset specifications can be found in [18]. The dataset features a vocabulary of 11 classes encompassing human sounds and sound events typically found in indoor office environments. The dataset is split into a development and evaluation sets. The development set consists of a four fold cross-validation setup.

The SELD task is evaluated with individual metrics for SED (F-score (*F*) and error rate (*ER*) calculated in one-second segments) and DOA estimation (DOA error (*DOA*) and frame recall (*FR*) calculated frame-wise) [6]. The *SELD score* is an averaged summary of the system performance.

The baseline system features a CRNN that jointly performs DOA and SED through multi-task learning [6]. Baseline results are shown in Table 2.

3.2. Parametric front-end

Based on the method's exploratory analysis, we propose the following set of parameter values, which are shown in Table 1.

Table 1: Parameter values for the selected configuration. Top: *DOA analysis* parameters. Bottom: *Association* parameters.

Parameter	Unit	Value
STFT window size	sample	256
analysis frequency range	Hz	[0,8000]
time average vicinity radius r	bin	10
diffuseness mask threshold Ψ_{max}	-	0.5
energy density filter length	bin	11
std mask vicinity radius	bin	2
std mask normalized threshold	-	0.15
median filter minimum ratio B_{min}	-	0.5
median filter vicinity radius (k,n)	bin	(20, 20)
resampling minimum valid bins K_{min}	bin	1
overlapping std threshold σ_{max}	degree	10
grouping maximum angle d_{max}^{ANGLE}	degree	20
grouping maximum distance d_{max}^{FRAME}	frame	20
event minimum length	frame	8

3.3. Deep learning classification back-end

We use the provided four fold cross-validation setup. Training and validation stages use the outcome of an *ideal* front-end, where the groundtruth DOA estimation and activation times are used to feed the beamformer for time-space segmentation. Conversely, we test the trained models with the signals coming from the *complete* front-end described in Section 2. We conducted a set of preliminary experiments with different types of networks including a VGG-like net, a less deep CNN [19], a Mobilenetv1 [20] and a CRNN [21]. The latter was found to stand out, and we explore certain facets of the CRNN architecture and the learning pipeline.

Sound events in the dataset last from ∼ 0.2 to 3.3 s. First, clips shorter than 2s are replicated to meet this length. Then, we compute TF patches of log-mel spectrograms of $T = 50$ frames (1 s) and $F = 64$ bands. The values come from the exploration of $T \in \{25, 50, 75, 100\}$ and $F \in \{40, 64, 96, 128\}$. $T = 50$ is the top performing value, roughly coinciding with the median event duration. In turn, more than 64 bands provide inconsistent improvements, at the cost of increasing the number of network weights.

Several variants of the CRNN architecture were explored until reaching the network of Figure 4. This included a small grid search over number of CNN filters, CNN filter size and shape, number of GRU units, number of FC units, dropout [22], learning rate, and the usage of Batch Normalization (BN) [23]. Network extensions (involving more weights) were considered only if providing major improvements, as a measure against overfitting. The main takeaways are: *i)* squared 3x3 filters provide better results than larger filters, *ii)* dropout of 0.5 is critical for overfitting mitigation, *iii)* more than one recurrent layer does not yield improvements, while slowing down training, and *iv)* surprisingly, slightly better performance is attained without BN nor pre-activation [24]. For all experiments, the batch size was 100 and Adam optimizer was used [25] with initial learning rate of 0.001, halved each time the validation accuracy plateaus for 5 epochs. Earlystopping was adopted with a patience of 15 epochs, monitoring validation accuracy. Prediction for every event was obtained by computing predictions at the patch level, and aggregating them with the geometric mean to produce a clip-level prediction.

Finally, we apply *mixup* [26] as data augmentation technique. Mixup consists in creating virtual training examples through linear interpolations in the feature space, assuming that they correspond to linear interpolations in the label space. Essentially, virtual TF patches are created on the fly as convex combinations of the input training patches, with a hyper-parameter α controlling the interpolation strength. Mixup has been proven successful for sound event classification, even in adverse conditions of corrupted labels [27]. It seems appropriate for this task since the front-end outcome can present leakage due to overlapping sources, effectively mixing two sources while only one training label is available, which can be understood as a form of label noise [19]. Experiments revealed that mixup with $\alpha = 0.1$ boosted testing accuracy in $\sim 1.5\%$.

4. RESULTS AND DISCUSSION

Table 2: Results for development (top) and evaluation (bottom) sets.

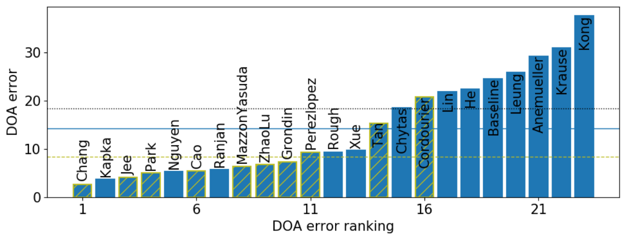
Method	*ER*	*F*	*DOA*	*FR*	*SELD*
Baseline	0.34	79.9%	28.5°	85.4%	0.2113
Proposed	0.32	79.7%	9.1°	76.4%	**0.2026**
Ideal front-end	0.08	93.2%	$\sim 0°$	$\sim 100\%$	0.0379
Baseline	0.28	85.4%	24.6°	85.7%	0.1764
Proposed	0.29	82.1%	9.3°	75.8%	**0.1907**

Table 2 shows the results of the proposed method for both development and evaluation sets, compared to the baseline. Focusing on evaluation results, our method and the baseline obtain similar performance in SED (*ER* and *F*). However, there is a clear difference in the DOA metrics: in our method, *DOA* error is reduced by a factor of 2.6, but *FR* is $\sim$ 10 points worst. In terms of *SELD score*, our method performs slightly worse than the baseline in evaluation mode, while marginally outperforming it in development mode.

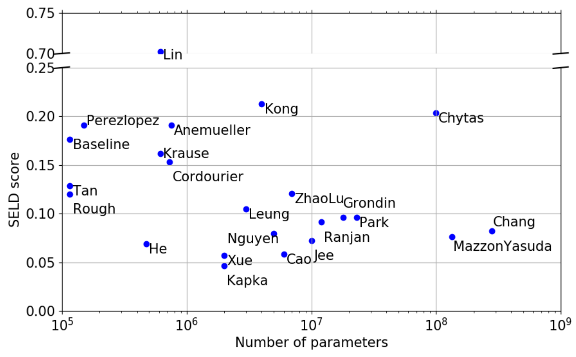
The most relevant observation is the great improvement in *DOA* error. Results suggest that using spatial audio parametric analysis as a preprocessing step can help to substantially improve localization. Figure 5a provides further evidence for this argument: Challenge methods using some kind of parametric preprocessing (*GCC-PHAT* with the microphone dataset, and *Intensity Vector-Based* in ambisonics) obtained in average better *DOA* error results.

Conversely, the front-end fails regarding *FR*. This is probably due to the complexity added by the association step [6], and its lack of robustness under highly reverberant scenarios. Including spectral information at the grouping stage might help to improve *FR* — such information could be provided by the classification back-end, in a similar approach to the baseline system. Another option would be the usage of more sophisticated source counting methods [28, 29].

In order to gain a better insight of the classification back-end performance, Table 2 shows the method results when the testing clips are obtained by feeding the beamformer with groundtruth annotations (*ideal* front-end). In this ideal scenario of DOA performance, the SED metrics show a significant boost. This result suggests that the low *FR* given by the front-end has a severe impact on the back-end performance. Yet, the proposed system reaches similar performance to the baseline system in terms of SED metrics.



(a) *DOA error* across submissions. Hatched bars denote methods using parametric preprocessing. Horizontal lines depict average DOA error accross different subsets: all methods (solid), parametric methods (dashed), non-parametric methods (dotted).



(b) *SELD score* versus complexity.

Figure 5: DCASE2019 Challenge Task 3 results, evaluation set.

Finally, we would like to discuss algorithm complexity among Challenge methods. As depicted in Figure 5b, there is a general trend towards architectures with very high number of weights, as a consequence of the usage of ensembles and large capacity networks. Specifically, 66% of submitted methods employ 1M weights or more, 30% employ 10M or more, and 15% employ 100M or more. Such complexities are several orders of magnitude greater than the baseline (150k weights) or the proposed method ($\sim$175k weights). In this context, our method represents a low-complexity solution to the SELD problem, featuring a number of parameters and a performance comparable to the baseline method.

5. CONCLUSION

We present a novel approach for the SELD task. Our method relies on spatial parametric analysis for the computation of event DOAs, onsets and offsets. This information is used to filter the input signals in time and space, and the resulting event estimations are fed into a CRNN which predicts the class to which the events belong; the classification problem is thereby handled from a simple multi-class perspective. The proposed method is able to obtain an overall performance comparable to the baseline system. The localization accuracy achieved by our method greatly improves the baseline performance, suggesting that spatial parametric analysis might enhance performance of SELD algorithms. Moreover, detection and classification performance in our method suffers from a low Frame Recall; improving this metric could lead to promising SELD scores.

6. REFERENCES

[1] C. Grobler, C. P. Kruger, B. J. Silva, and G. P. Hancke, “Sound based localization and identification in industrial environments,” in *IECON 2017-43rd Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2017, pp. 6119–6124.

[2] T. Butko, F. G. Pla, C. Segura, C. Nadeu, and J. Hernando, “Two-source acoustic event detection and localization: Online implementation in a smart-room,” in *2011 19th European Signal Processing Conference*. IEEE, 2011, pp. 1317–1321.

[3] R. Chakraborty and C. Nadeu, “Sound-model-based acoustic source localization using distributed microphone arrays,” in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 619–623.

[4] T. Hirvonen, “Classification of spatial audio location and content using convolutional neural networks,” in *Audio Engineering Society Convention 138*. Audio Engineering Society, 2015.

[5] K. Lopatka, J. Kotus, and A. Czyzewski, “Detection, classification and localization of acoustic events in the presence of background noise for acoustic surveillance of hazardous situations,” *Multimedia Tools and Applications*, vol. 75, no. 17, pp. 10 407–10 439, 2016.

[6] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, “Sound event localization and detection of overlapping sources using convolutional recurrent neural networks,” *IEEE Journal of Selected Topics in Signal Processing*, pp. 1–1, 2018.

[7] M. A. Gerzon, “Periphony: With-height sound reproduction,” *Journal of the Audio Engineering Society*, vol. 21, no. 1, pp. 2–10, 1973.

[8] J. Daniel, “Représentation de champs acoustiques, application à la transmission et à la reproduction de scènes sonores complexes dans un contexte multimédia,” 2000.

[9] V. Pulkki, “Directional audio coding in spatial sound reproduction and stereo upmixing,” in *Audio Engineering Society Conference: 28th International Conference: The Future of Audio Technology–Surround and Beyond*. Audio Engineering Society, 2006.

[10] S. Berge and N. Barrett, “High angular resolution planewave expansion,” in *Proc. of the 2nd International Symposium on Ambisonics and Spherical Acoustics May*, 2010, pp. 6–7.

[11] A. Politis, S. Tervo, and V. Pulkki, “COMPASS: Coding and Multidirectional Parameterization of Ambisonic Sound Scenes,” *IEEE International Conference on Acoustics, Speech, and Signal Processing*, no. May, pp. 6802–6806, 2018.

[12] V. Pulkki, S. Delikaris-Manias, and A. Politis, *Parametric time-frequency domain spatial audio*. Wiley Online Library, 2018.

[13] https://github.com/andresperezlopez/DCASE2019_task3.

[14] T. Carpentier, “Normalization schemes in ambisonic: does it matter?” in *Audio Engineering Society Convention 142*. Audio Engineering Society, 2017.

[15] D. Stanzial, N. Prodi, and G. Schiffrer, “Reactive acoustic intensity for general fields and energy polarization,” *The Journal of the Acoustical Society of America*, vol. 99, no. 4, pp. 1868–1876, 1996.

[16] J. Merimaa and V. Pulkki, “Spatial impulse response rendering i: Analysis and synthesis,” *Journal of the Audio Engineering Society*, vol. 53, no. 12, pp. 1115–1127, 2005.

[17] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[18] S. Adavanne, A. Politis, and T. Virtanen, “A multiroom reverberant dataset for sound event localization and uetection,” in *Submitted to Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019. [Online]. Available: https://arxiv.org/abs/1905.08546

[19] E. Fonseca, M. Plakal, D. P. W. Ellis, F. Font, X. Favory, and X. Serra, “Learning sound event classifiers from web audio with noisy labels,” in *Proc. IEEE ICASSP 2019*, Brighton, UK, 2019.

[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” *arXiv preprint arXiv:1704.04861*, 2017.

[21] E. Cakır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, “Convolutional recurrent neural networks for polyphonic sound event detection,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1291–1303, 2017.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting,” *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[23] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International Conference on Machine Learning*, 2015, pp. 448–456.

[24] E. Fonseca, R. Gong, and X. Serra, “A simple fusion of deep and shallow learning for acoustic scene classification,” in *Proceedings of the 15th Sound & Music Computing Conference (SMC 2018)*, Limassol, Cyprus, 2018.

[25] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *ICLR 2015*. [Online]. Available: https://arxiv.org/abs/1412.6980

[26] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” *arXiv preprint arXiv:1710.09412*, 2017.

[27] E. Fonseca, F. Font, and X. Serra, “Model-agnostic approaches to handling noisy labels when training sound event classifiers,” in *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, New York, US, 2019.

[28] Z. He, A. Cichocki, S. Xie, and K. Choi, “Detecting the number of clusters in n-way probabilistic clustering,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 11, pp. 2006–2021, 2010.

[29] N. Stefanakis, D. Pavlidi, and A. Mouchtaris, “Perpendicular Cross-Spectra Fusion for Sound Source Localization with a Planar Microphone Array,” *IEEE/ACM Transactions on Audio Speech and Language Processing*, vol. 25, no. 9, pp. 1517–1531, 2017.

CLASSIFYING NON-SPEECH VOCALS: DEEP VS SIGNAL PROCESSING REPRESENTATIONS

Fatemeh Pishdadian, Bongjun Kim, Prem Seetharaman, Bryan Pardo

Northwestern University
Electrical Engineering and Computer Science
Evanston, IL, USA
{fpishdadian, bongjun, prem}@u.northwestern.edu, pardo@northwestern.edu

ABSTRACT

Deep-learning-based audio processing algorithms have become very popular over the past decade. Due to promising results reported for deep-learning-based methods on many tasks, some now argue that signal processing audio representations (e.g. magnitude spectrograms) should be entirely discarded, in favor of learning representations from data using deep networks. In this paper, we compare the effectiveness of representations output by state-of-the-art deep nets trained for task-specific problems, to off-the-shelf signal processing representations applied to those same tasks. We address two tasks: query by vocal imitation and singing technique classification. For query by vocal imitation, experimental results showed deep representations were dominated by signal-processing representations. For singing technique classification, neither approach was clearly dominant. These results indicate it would be premature to abandon traditional signal processing in favor of exclusively using deep networks.

Index Terms— audio signal representation, audio processing, audio classification, query by example, deep learning

1. INTRODUCTION

Recently, deep-learning-based audio processing has gained great popularity, due to the promising results these methods have produced for tasks such as audio classification [1] and audio source separation [2]. As a result, some argue that representations built using signal-processing knowledge and theory (e.g. Fourier transforms, cepstrograms, etc.) should be entirely discarded, in favor of learning representations from data using deep networks [3].

In this work, we study the efficacy of both deep and signal-processing representations in the context of content-based audio retrieval and audio classification, focusing on two example tasks within these broad categories: query by vocal imitation and singing technique classification. Given a collection of audio files, Query by vocal imitation (QBV) [4, 5, 6] aims to retrieve those files most similar to a user's vocal imitation of a sound (e.g. an imitation of dog barking). QBV is particularly useful when detailed text labels for audio samples are not available. Singing technique classification (e.g. Broadway belting, vocal fry) is useful for automated music instruction, genre recognition [7], and singer identification [8]. On both of these tasks, the current state-of-the-art reported in the literature uses a deep model to encode the audio.

This work was supported, in part, by USA NSF award 1617497.

We compare the effectiveness of representing the audio using a state-of-the-art deep model, trained specifically for a task, to the effectiveness of using one of three off-the-shelf signal processing representations. We use a nearest-neighbor classification framework to perform query by vocal imitation and singing technique classification. Audio queries and sound files in the database are both encoded in the same way (with either a deep net or a signal processing method), and then a nearest-neighbor classification is performed to find the database example most like the query. If deep representations are truly better for these tasks, then encoding audio with a task-specific deep model should make the task-salient information more prominent than encoding with a signal processing method. This should translate to better performance.

The results of this study are not what recent literature would lead one to expect. The representation that stands out as the most useful is the 2D Fourier transform of a constant-Q spectrogram, rather than the representation produced by any deep network. These results indicate it would be premature to abandon traditional signal processing approaches in favor of exclusively using deep networks.

2. DEEP REPRESENTATIONS

In query by vocalization (QBV), people tend to remain more faithful to the general shape of spectral modulations rather than the exact pitch or timing of a reference audio. Therefore, a QBV representation should be able to capture modulation patterns and also be robust against small deviations in the pitch or timing of a query with respect to the target sound. Similarly, singing techniques create modulation patterns that serve as powerful discriminants for singing styles. It is thus desirable for an audio representation used for these tasks to preserve the modulations and present them explicitly.

Convolutional layers in deep networks are known to be effective in capturing shift-invariant patterns. For instance, if the input to a convolutional layer is an audio spectorgram, the layer can be trained to extract up-/downward moving spectral patterns, i.e. spectro-temporal modulations, regardless of their start time and offset frequency [9]. It is, therefore, not surprising that convolutional nets (CNNs) are the current state-of-the-art on both tasks. We now describe the specific networks used in our experiments.

TL-IMINET [10] is a deep net built specifically for query by vocal imitation (QBV). The trained network takes a pair of audio recordings as input: a vocal imitation (e.g. a human imitation of a dog bark) and an original recording (e.g. a real dog bark) and outputs a similarity rating ranging from 0 to 1. TL-IMINET has two convolutional towers that feed into several fully connected layers that combine input from the two towers. Each tower has three

https://doi.org/10.33682/bczf-wv12

convolutional layers with max-pooling. The specific filters (their number, arrangement, etc.) differ between the two towers, as one was designed to capture features from an original recording and the other from a vocal imitation. The authors argue that the convolutional towers capture spectro-temporal modulation patterns resembling feature maps used by mammals in the auditory system. Each CNN tower takes a 4 second-long log-mel spectrogram as input. We use a replication of TL-IMINET, trained on the same data sets to a performance level equal to that reported in the original paper. One can consider this network a specialist for the QBV task.

The *VGGish model* [11] is a CNN-based model trained on the audio from 8 million YouTube videos to distinguish 3,000 sound classes. It has 6 convolutional layers, followed by 3 fully-connected layers. It takes a log-mel spectrogram (64 Mel bins, window size of 25 ms, and hop size of 10 ms) as input and outputs a 128-dimensional feature embedding for every 1-second segment of the input audio. We selected it as an example of deep network architectures and trained models used for a variety of audio labeling tasks. One can consider the audio embedding produced by VGGish as a "general" audio representation. As such, it is used as a sanity-check baseline for both QBV and singing technique classification. No task-specific model should do worse than this general audio model.

Modified VGGish [12] (M-VGGish) is a network for query by vocalization. Instead of extracting the feature embedding from the final layer of a VGGish model, the authors used intermediate representations from the convolutional layers, which resulted in better QBV performance than the original VGGish feature embedding. The model takes an arbitrary length recording and outputs a feature vector for every 2-second segment of the recording. To form the segment-level feature vector, the outputs from the last two convolutional layers are concatenated, then the set of segment-level feature vectors are averaged to form a clip-level feature vector. One can consider this network a specialist for the QBV task.

Wilkins et al. [13] made a convolutional neural network that is the current state-of-the-art for singing technique classification. It is an end-to-end model that takes a raw PCM audio waveform as input and outputs a probability distribution over singing techniques. It is a specialist for singing technique classification.

2.1. Signal-processing-based representations

In this section, we discuss the signal processing representations used in our experiments. Time-frequency representations, such as the magnitude spectrogram are, perhaps, the most commonly used audio features. For this study, we used a log-frequency magnitude spectrogram built using a *Constant-Q Transform* (CQT) [14]. The log-scale frequency spacing of the CQT preserves the spacing between overtones of harmonic sounds (e.g. human speech) when the fundamental frequency changes. A log-frequency magnitude spectrogram is used as input to three of the four deep models included in this study (TL-IMINET, VGGish, modified VGGish). Therefore, one can consider the CQT spectrogram a baseline. If a nearest-neighbor classifier performs better using a CQT spectrogram as input than it does using the output of one of these deep models, then that model is not performing task-relevant work.

The *2D Fourier Transform (2DFT)* is an image processing tool that was not originally developed for audio. It decomposes an image into a set of scaled and phase-shifted 2D sinusoids. The 2DFT can be used to analyze the time-frequency representations (e.g. CQT) of audio signals [15, 16]. Repeating patterns in a time-frequency representation, such as overtones of a harmonic sound, are grouped together and manifest as peaks in the 2DFT domain. Spectro-temporal modulation patterns can thus be effectively encoded by the 2DFT as a set of peaks. The magnitude 2DFT of an audio spectrogram is invariant with respect to frequency or time shifts of modulation patterns. The 2DFT has been recently used in applications such as music/voice separation [15] and cover song identification [17, 16]. Since it has proven successful in these very different tasks, we were interested in exploring its potential for the tasks in this study. In our experiments, we apply the 2DFT to the log-frequency magnitude spectrogram built from the CQT.

Scale-rate (*SR* in this work) is a modulation-related feature representation computed based on the *Multi-resolution Common Fate Transform (MCFT)* [18][1]. The MCFT is a bio-inspired representation initially proposed for the task of audio source separation, which encodes spectro-temporal modulation patterns as explicit dimensions. The modulation-related dimensions are termed *scale* and *rate* [19], respectively encoding the spectral spread and modulation velocity over time. SR is built by applying the 2D filterbank of the MCFT to the magnitude CQT of audio signals and averaging the results over time and frequency. This representation was chosen due to its explicit representation of spectro-temporal modulations, which we believe to be useful in both QBV and singing technique classification tasks.

While we had reason to believe that the 2DFT of the log spectrogram and the scale-rate representation would capture vocal spectro-temporal modulations well, none of the signal processing representations in our study were designed for either the QBV or singing technique classification task. This contrasts with the deep networks we tested, which were made specifically for each task.

3. EXPERIMENTS

We consider an audio representation (e.g. the deep embedding output by M-VGGish, or the CQT spectrogram) as effective if the task-relevant distinctions between audio examples can be easily captured by a similarity measure applied to those examples encoded in the representation. The performance of a K-nearest-neighbor classifier, for instance, is directly affected by the audio representation. The better the task-related information is represented, the better the classifier works, the better the retrieval (or labeling) performance would be. We evaluate the effectiveness of a representation in this light.

3.1. Query by vocal imitation

To evaluate the performance of the representations in the query by vocal imitation (QBV) scenario we used the VimSketch dataset[2], which combines the datasets from two previous publications [20, 21] to create the largest single dataset for QBV. VimSketch contains 542 reference sounds (including a variety of animal sounds, musical snippets, and environmental noise samples) and 12,543 vocal imitations of those reference sounds with a minimum of 13 and a maximum of 37 vocal imitations per reference.

All audio examples encoded by the representations were zero-padded to the length of the longest example in the dataset (15.4 seconds). All signal processing representations, VGGish and M-VGGish used full length audio examples. Audio examples for TL-IMINET were limited to the initial 4 seconds long. This duplicates the published approach for TL-IMINET [10].

[1]https://interactiveaudiolab.github.io/MCFT/
[2]http://doi.org/10.5281/zenodo.2596911

The VimSketch audio files were originally sampled at different rates (ranging from 8 kHz to 192 kHz). Thus, we first resampled them to have a common rate, 8 kHz. For the signal processing representations (CQT, 2DFT, and SR), the range of frequencies was limited to 55 Hz to 2.09 kHz (pitches A1 to C7). This was done to keep the computations tractable. Audio was upsampled to 16kHz to accommodate the input requirements of VGGish and M-VGGish. Input to TL-IMINET was upsampled to 16kHz for the imitation tower and 44.1kHz for the reference tower, to meet its requirements. We note that even though the initial resampling removes the frequencies above 4 kHz, the comparison is still reasonable since: i) these high frequencies are unavailable to all representations alike and ii) the sounds remain recognizable by humans.

Given a vocal query, the output of a QBV system is a list of reference sound examples ordered based on their similarity to the query. In our experiments with the deep net encoders (VGGish, M-VGGish) and all signal processing features (CQT, 2DFT, SR), we use the cosine similarity measure to select the most similar sound examples to a query in the representation domain. We selected the cosine similarity measure because this is the similarity measure applied in the published results for the state-of-the-art M-VGGish network for query by vocal imitation [21].

The cosine similarity between a query $\mathbf{V_q}$ and a reference $\mathbf{V_r}$ is defined as:

$$S_{cos}(\mathbf{V_q}, \mathbf{V_r}) = \frac{\langle \mathbf{V_q}, \mathbf{V_r} \rangle}{\|\mathbf{V_q}\| \|\mathbf{V_r}\|}, \quad (1)$$

where $\langle \cdot, \cdot \rangle$ and $\|\cdot\|$ are the inner product and Euclidean norm.

Unlike VGGish and M-VGGish, TL-IMINET was not designed to produce an audio embedding (a.k.a. representation) to be used by an external similarity measure or classifier. It instead directly outputs a similarity measure between pairs of examples, which we use in place of the cosine similarity applied to all other representations.

The performance of the QBV system can be evaluated in terms of the rank of the target sound in the output list of sound examples. We use Mean Reciprocal Rank (MRR) as the performance measure:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}, \quad (2)$$

where Q denotes a set of queries and $rank_i$ refers to the rank position of the target sound for the ith query.

Recall that a 'reference' in this context is an audio file in the collection (e.g. a car horn recording) and a 'query' is a vocal imitation of some reference file. The MRR value for each representation is computed over datasets of size n = 20, 50, 100, 200, 400, and 542 references. Since there are 12,543 vocal imitations in the data, the MRR value for each reference set is computed by averaging over 12,543 reciprocal ranks. To ensure no result is due to a selection of a reference set that is skewed to favor a particular representation, reference set selection is repeated 100 times for each value of n below 542 (the size of the full reference set). The average MRR over all 100 iterations is reported.

3.1.1. Signal Processing Hyperparameters

In computing the CQT and 2DFT, we treat the frequency resolution of the CQT as a tunable parameter, taking on values of 12, 24, 48, or 96 bins/octave. For SR, we keep the frequency resolution fixed to the best performing 2DFT resolution for the QBV task and then treat the scale and rate resolutions as tunable parameters with values 1, 2, 4, or 8 bins/octave.

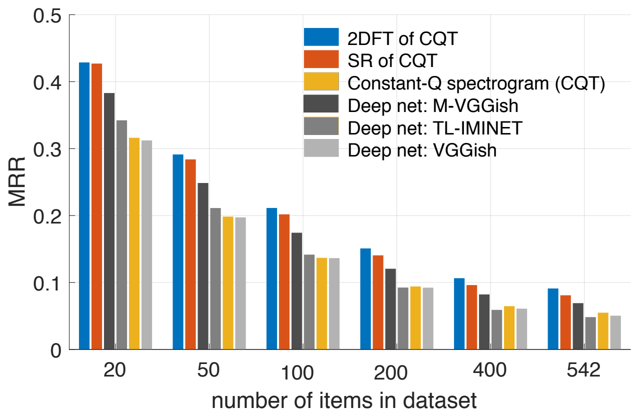


Figure 1: Query by vocalization search results. Deep representations (grayscale bars) are compared to signal processing representations (colored bars). Higher values are better. Each signal processing representation uses the **worst** parameters found for the task. Two of the deep nets (M-VGGish and TL-IMINET) were constructed specifically for query by vocalization. Nevertheless, all deep representations are dominated by two signal processing representations (2DFT of CQT and SR).

For the QBV task, the worst and best results using the CQT and 2DFT are obtained with a frequency resolution of 96 (worst) and 12 bins/octave (best). Increasing the frequency resolution has a negative effect on the performance of both features, showing the importance of high temporal resolutions in capturing the fine structure of modulations. The scale resolution does not impact the performance of SR features significantly, and hence it is fixed to 1 bin/octave for the reported results. The rate resolution, on the other hand, has a noticeable effect, giving the worst results when set to 1 bin/octave and best results when set to 8 bins/octave.

3.1.2. Results

QBV task results are presented in Figure 1. To tilt the comparison in favor of the deep representations as far as possible, we show only the results for the **worst** tunable parameter settings found for the signal processing approaches (CQT, 2DFT, and SR). Results shown for 2DFT and CQT use the **worst** frequency resolution tested. Results for SR show the **worst** scale and rate tested. Therefore, Figure 1 compares off-the-shelf signal processing representations that use bad hyperparameter choices to published task-specific deep models, tuned to work well on the kind of data used for evaluation.

It can be clearly observed that the 2DFT and SR features outperformed all other representations, even with their worst parameter selection. This superiority holds for all sizes of dataset tested. CQT is a log-frequency spectrogram. VGGish, M-VGGish, and TL-IMINET all use a log-frequency spectrogram as input. VGGish was neither trained nor designed for the QBV task and serves as a baseline among the deep nets. Therefore, it is not surprising that using the output of VGGish as a representation is roughly equivalent to simply using a constant-Q spectrogram (the CQT).

The deep nets TL-IMINET and M-VGGish were both constructed for the specific QBV task we tested them on. Surprisingly, TL-IMINET, a network designed and trained specifically for the QBV task, shows degraded performance as the dataset grows, to the point where it is would actually be preferable to use a constant-Q spectrogram, which was the worst-performing of the signal processing representations.

As can be seen, only M-VGGish consistently improved upon CQT as the database size increases. This shows it is possible to create a deep representation that is consistently better than its input representation for this task. That said, M-VGGish never achieved the performance of either of the top signal processing representations (2DFT and SR), despite the fact that we compared to the worst parameter settings for both. We hypothesize that the superior performance of the signal processing representations on this task may be due to the fact that SR and 2DFT features inherently capture spectro-temporal modulations and present them in a time/frequency-shift-invariant fashion. While a deep representation may be able to represent such modulations, this comparison illustrates that even a network explicitly designed and trained for this task (e.g. TL-IMINET) may not perform as well as an existing signal processing approach.

3.2. Singing technique classification

For the task of singing technique classification, we used VocalSet [13], a singing voice dataset that includes a large set of voices (9 female and 11 male professional singers) performing 17 different singing techniques. We extracted the samples corresponding to 10 different singing techniques (belt, breathy, inhaled singing, lip trill, spoken excerpt, straight tone, trill, trillo, vibrato, and vocal fry) by all singers, which amounts to 915 samples ranging in length from 1.7 to 21.5 seconds. All audio examples were resampled to 8 kHz.

We compared our results to those of the classifier proposed by Wilkins et al. [13], which was directly trained on VocalSet, and thus is expected to perform very well. Their classifier is a neural network, composed of three convolutional layers followed by two dense layers. The network receives a 3-second time-domain audio excerpt as input and outputs the predicted vocal technique class. We use the same technique classes and the same training/testing data split as in their experiments. The training and testing sets include samples from 15 and 5 singers, respectively.

The signal processing representations compared to Wilkins et al. were CQT, 2DFT, and SR. We also compared to VGGish, a deep net not trained on this specific task. This provided a baseline deep net, much the way the CQT provides a baseline signal processing representation. The singing techniques were classified using the K-Nearest Neighbors algorithm, with the cosine similarity measure and $K = 3$ used as algorithm parameters in all experiments.

Since audio examples are of different lengths, we had to decide whether to zero-pad or cut all files to the same length. Two lengths were tried. First, we extracted the initial 3 seconds of all examples, which is the same length used by the deep net in Wilkins et al. [13]. We expected this to favor their deep net. Next, we found the signal length that maximized the performance of the VGGish deep net (18 seconds) and zero-padded or cut all examples to that length.

We measured the classification performance in terms of precision, recall, and F-measure. Table 1 shows results for 3-second examples and Table 2 the results for 18-second examples.

The frequency range for the CQT and the parameter tuning strategies for the 2DFT and SR features were the same as in Section 3.1. Since the signal processing approaches were not as dominant in this task, we report the results using both the best and the worst parameter settings for these representations. In both tables, the best frequency resolutions for the CQT and 2DFT are 96 and 24 bins/octave, respectively. In both tables, the best scale resolution is 1 bin/octave and the best rate resolution 8 bins/octave.

Representation	Precision	Recall	F-measure
Deep: Wilkins et al.	**0.677**	**0.628**	**0.651**
Deep: VGGish	0.556	0.54	0.529
CQT-best	0.61	0.528	0.519
CQT-worst	0.52	0.468	0.448
2DFT-best	**0.665**	**0.624**	**0.637**
2DFT-worst	**0.660**	**0.58**	**0.597**
SR-best	0.562	0.564	0.554
SR-worst	0.449	0.44	0.434

Table 1: Singing technique classification (10 classes): Results for 3-second excerpts. Higher values are better.

Representation	Precision	Recall	F-measure
Deep: VGGish	0.627	0.6	0.602
CQT-best	0.533	0.488	0.479
CQT-worst	0.43	0.432	0.408
2DFT-best	**0.723**	**0.692**	**0.698**
2DFT-worst	**0.674**	**0.636**	**0.646**
SR-best	0.615	0.612	0.603
SR-worst	0.612	0.6	0.599

Table 2: Singing technique classification (10 classes): Results for 18-second excerpts. Higher values are better.

It can be observed that in the 3-second case, the 2DFT outperforms the VGGish embeddings by a large margin and a simple parameter tuning (frequency resolution) brings its performance close to the network that was specifically trained for the VocalSet data (Wilkins et al.). When applied to excerpts of longer duration (Table 2), the 2DFT is able to capture long-term modulations even more efficiently, yielding a higher F-measure than the state-of-the-art results reported by Wilkins et al. on 3-second examples.

4. CONCLUSION

For query by vocalization, a nearest-neighbor method that applies cosine similarity to either of two off-the-shelf signal processing methods (2DFT and SR applied to a constant-Q spectrogram) outperformed similarity measures built using two different deep approaches designed specifically for this task (M-VGGish and TL-IMINET), as well as a general audio deep representation (VGGish). For singer technique classification, a 2DFT representation was competitive with or outperformed the task-specific deep network that is the current state-of-the-art (Wilkins et al. [13]), depending on the choice of parameters, and also outperformed a general audio representation (VGGish).

The deep networks evaluated here defined the state of the art on both tasks until this study. We hypothesize that the ability of both SR and 2DFT to explicitly represent spectro-temporal modulations in a time/frequency-shift-invariant fashion is key to their effectiveness with non-speech vocal classification. While a deep representation may be able to represent such modulations, this comparison illustrates that even a network explicitly designed and trained for non-speech vocal classification may not perform as well at representing these features. Given our results, it would be premature to abandon traditional signal processing techniques in favor of exclusively using deep networks.

5. REFERENCES

[1] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[2] Y. Luo, Z. Chen, J. R. Hershey, J. Le Roux, and N. Mesgarani, "Deep clustering and conventional networks for music separation: Stronger together," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 61–65.

[3] S. Venkataramani and P. Smaragdis, "End-to-end networks for supervised single-channel speech separation," *arXiv preprint arXiv:1810.02568*, 2018.

[4] M. Cartwright and B. Pardo, "Synthassist: an audio synthesizer programmed with vocal imitation," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 741–742.

[5] G. Lemaitre, O. Houix, F. Voisin, N. Misdariis, and P. Susini, "Vocal imitations of non-vocal sounds," *PloS one*, vol. 11, no. 12, p. e0168167, 2016.

[6] A. Mehrabi, K. Choi, S. Dixon, and M. Sandler, "Similarity measures for vocal-based drum sample retrieval using deep convolutional auto-encoders," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 356–360.

[7] W.-H. Tsai, D. Rodgers, and H.-M. Wang, "Blind clustering of popular music recordings based on singer voice characteristics," *Computer Music Journal*, vol. 28, no. 3, pp. 68–78, 2004.

[8] M. A. Bartsch and G. H. Wakefield, "Singing voice identification using spectral envelope estimation," *IEEE Transactions on speech and audio processing*, vol. 12, no. 2, pp. 100–109, 2004.

[9] J. Schlüter, "Learning to pinpoint singing voice from weakly labeled examples." in *ISMIR*, 2016, pp. 44–50.

[10] Y. Zhang, B. Pardo, and Z. Duan, "Siamese style convolutional neural networks for sound search by vocal imitation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 2, pp. 429–441, 2019.

[11] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017, pp. 131–135.

[12] B. Kim and B. Pardo, "Improving content-based audio retrieval by vocal imitation feedback," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 4100–4104.

[13] J. Wilkins, P. Seetharaman, A. Wahl, and B. Pardo, "Vocalset: A singing voice dataset," in *Proc. 19th Conf. Int. Society for Music Information Retrieval (ISMIR)*, 2018, pp. 468–474.

[14] C. Schörkhuber, A. Klapuri, N. Holighaus, and M. Dörfler, "A matlab toolbox for efficient perfect reconstruction time-frequency transforms with log-frequency resolution," in *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio*. Audio Engineering Society, 2014.

[15] P. Seetharaman, F. Pishdadian, and B. Pardo, "Music/voice separation using the 2d fourier transform," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 36–40.

[16] D. P. Ellis and B.-M. Thierry, "Large-scale cover song recognition using the 2d fourier transform magnitude," 2012.

[17] P. Seetharaman and Z. Rafii, "Cover song identification with 2d fourier transform sequences," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 616–620.

[18] F. Pishdadian and B. Pardo, "Multi-resolution common fate transform," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 2, pp. 342–354, 2019.

[19] T. Chi, P. Ru, and S. A. Shamma, "Multiresolution spectrotemporal analysis of complex sounds," *The Journal of the Acoustical Society of America*, vol. 118, no. 2, pp. 887–906, 2005.

[20] M. Cartwright and B. Pardo, "Vocalsketch: Vocally imitating audio concepts," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 2015, pp. 43–46.

[21] B. Kim, M. Ghei, B. Pardo, and Z. Duan, "Vocal imitation set: a dataset of vocally imitated sound events using the audioset ontology," in *Workshop on Detection and Classification of Acoustic Scenes and Events*, 2018.

SOUND EVENT LOCALIZATION AND DETECTION USING CRNN ARCHITECTURE WITH *MIXUP* FOR MODEL GENERALIZATION

Pranay Pratik [1*], *Wen Jie Jee* [2*†], *Srikanth Nagisetty*[1], *Rohith Mars*[1], *Chong Soon Lim*[1],

[1] Panasonic R&D Center Singapore
{pranay.pratik, srikanth.nagisetty, rohith.mars, chongsoon.lim}@sg.panasonic.com
[2] Nanyang Technological University Singapore,
wjee001@e.ntu.edu.sg

ABSTRACT

In this paper, we present the details of our solution for the IEEE DCASE 2019 Task 3: Sound Event Localization and Detection (SELD) challenge. Given multi-channel audio as input, goal is to predict all instances of the sound labels and their directions-of-arrival (DOAs) in the form of azimuth and elevation angles. Our solution is based on Convolutional-Recurrent Neural Network (CRNN) architecture. In the CNN module of the proposed architecture, we introduced rectangular kernels in the pooling layers to minimize the information loss in temporal dimension within the CNN module, leading to boosting up the RNN module performance. Data augmentation *mixup* is applied in an attempt to train the network for greater generalization. The performance of the proposed architecture was evaluated with individual metrics, for sound event detection (SED) and localization task. Our team's solution was ranked 5th in the DCASE-2019 Task-3 challenge with an F-score of 93.7% & Error Rate 0.12 for SED task and DOA error of $4.2°$ & frame recall 91.8% for localization task, both on the evaluation set. This results showed a significant performance improvement for both SED and localization estimation over the baseline system.

***Index Terms*—** DCASE-2019, SELD, SED, Localization, CRNN, *mixup*

1. INTRODUCTION

Sound event localization and detection (SELD) is a challenging and well-researched topic in the field of acoustic signal processing. There are two sub-tasks for SELD, first: the sound event detection (SED), second: the sound source's direction estimation. An ideal SELD system would be able to detect & classify multiple sound events and for each detected sound event determines its direction of arrival. Signal processing algorithms have been traditionally employed to address this challenging task. However performance achieved by such methods are still limited under practical conditions.

In recent research, deep learning based techniques have been applied individually for both SED and localization part of the SELD task. In [1, 2], it has been shown that CNN based network can detect and classify sound events with high accuracy. In [3], 1D-CNN has been applied for solving the sound localization task. The recent trend in this field has been about developing deep learning techniques for joint localization and classification of multiple sound sources. In [4] authors proposed 2D-CNN based network for joint sound source localization and classification. In [5] authors introduced convolutional recurrent neural network architecture (CRNN), where the CNN module learns the audio spectral information followed by the RNN module, that learn the temporal information. This network architecture has been set as the baseline model in the DCASE2019 Task 3 challenge - Sound Event Localization & Detection. In [6] authors introduced two-stage training approach, which shows improvement in the overall performance over [5]. In this approach the training of the network is split into two branches, i.e., the SED branch and the localization branch.

In this paper, we proposed two deep CRNN architectures with log-mel spectrogram and generalized cross-correlation phase transforms (GCC-PHATs) as input features to the network. In the CNN module of one of the proposed network architecture, we restricted pooling in the frequency domain, this helps in preserving temporal information, boosting the performance of RNN module. Data augmentation technique *mixup* was used in an attempt to generalize the network. We investigated the effect of *mixup* on each of the sub-task, SED and localization and compared our results with baseline system provided by the DCASE-2019 challenge and with other prior-arts.

The rest of the paper is organized as follows. In Section 2, we presented the details on feature extraction, data augmentation technique and our proposed CRNN architectures. In Section 3, we discuss experiments setup & compare our results with prior-arts. Finally, conclusion and future work is presented in Section 4.

2. METHODOLOGY

In this section, we present our methodology starting with input feature extraction description followed by CRNN architecture description. In addition, we also discuss the data augmentation step used during training for improving model generalization. For training the network we adopted the strategy proposed in [6], where the model is first trained on SED task, then on localization task using the same network architecture.

2.1. Features

Input features plays a crucial role in training deep neural network. In this work, the raw data is in the form of four-channel audio signal, recorded at 48kHz sampling rate using a microphone array and was provided by DCASE Task-3 organizers [7]. The time domain multi-channel signals were first down-sampled to 32 kHz and then used

[*]Both authors contributed equally.

[†]This work was done during an internship at Panasonic R&D Center Singapore.

https://doi.org/10.33682/gbfk-re38

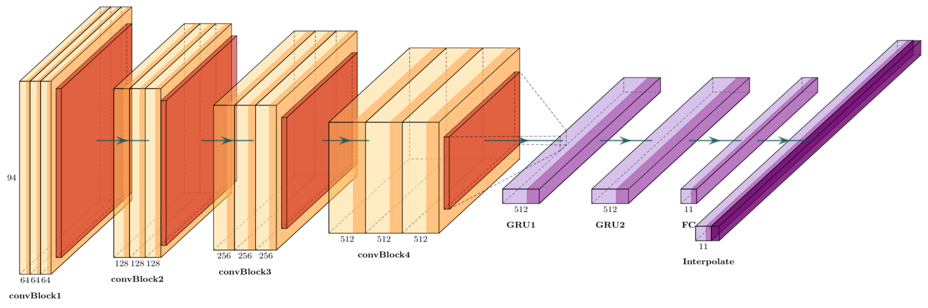


Figure 1: Base architecture **TS-C2Rnn**. Each convBlock contains three Conv2D layers followed by (2×2) average pooling. Each CNN layer is followed by batch normalization and ReLU activation. convBlock1 receives the input features.

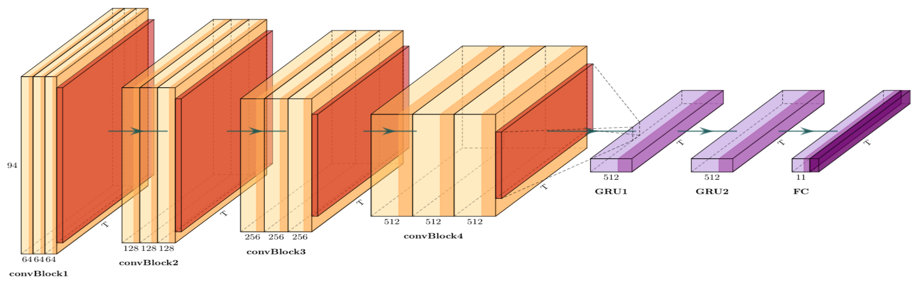


Figure 2: Proposed architecture **TS-C2Rnn-P**, with (2×1) average pooling after each convBlock, with no interpolation layer.

to extract log-mel spectrogram and GCC-PHAT features.

The log-mel spectrogram is commonly used as input feature in speech recognition [8] because of its similarity to frequency decomposition of the human auditory system. To obtain the log-mel spectrogram, time domain audio signal is converted to the time-frequency (TF) domain using short-time Fourier transform (STFT). This ensures that both the temporal and spectral characteristics of the audio data are utilized. After the frequency domain conversion, we extracted the log-mel spectrogram corresponding to each audio clip using 96 Mel bands. For the STFT parameters, we employ a frame length of 1024 points, with a hop length of 10 ms.

GCC-PHAT is widely used for estimation of time difference of arrival (TODA) of acoustic signal between two microphones. A basic frequency domain cross-correlation is estimated by the inverse-FFT of cross power spectrum. GCC is the improved version of cross-correlation, it is estimated by adding a windowing (or filtering) function prior to the inverse transform to improve the estimation of the time delay, depending on the specific characteristics of the signals and noise. GCC-PHAT is the phase-transformed version of GCC, which eliminate the influence of amplitude in the cross power spectrum, hence only preserving the phase information [9].

2.2. CRNN Architecture

The base network architecture introduced in this work is inspired from [6] and named as TS-C2Rnn as shown in figure 1 . The extracted audio features are provided as input to a CRNN architecture. CNN module of TS-C2Rnn consist of 4 convolutional blocks, named convBlock1 to convBlock4. Each convBlock is followed by an average pooling layer. Within each convBlock there are 3 convolutional layers, followed by batch normalization and ReLU activation. For convolutional layers in the convBlocks, 3×3 kernel is used, with stride and padding fixed to 1. The number of filters used in convBlock1 to convBlock4 are $\{convBlock1 : 64, convBlock2 : 128, convBlock3 : 256, convBlock4 : 512\}$. For performing average pooling in convBlocks, we used 2×2 window, with a stride of 2×2 . The CNN module of the network is followed by RNN module, which has two GRU layers, GRU-1 and GRU-2. The output of the GRU-2 layer is fed into fully connected (FC) layer of size N, where N is the number of sound event classes. FC layer is followed by interpolate layer to ensure the final number of the time frames is approximately equal to the original number of time frames of the input clip. This is necessary due to the presence of square kernels in the pooling layers in each convBlock. The output of the interpolate layer contains N class scores, azimuth and elevation values corresponding to each T time frames, where T varies from clip to clip.

We proposed another network architecture TS-C2Rnn-P which is a modified version of TS-C2Rnn architecture as shown in Figure 2. In the CNN module of TS-C2Rnn the 2×2 pooling across time and frequency domain reduces the information both in frequency and temporal dimension of feature maps. In order to preserve the time domain information which may be critical for GRU performance, we introduced 2×1 rectangular kernels in the CNN module pooling layers for TS-C2Rnn-P architecture. This results in restricting the pooling of feature maps in the frequency dimension.

Both the proposed networks TS-C2Rnn & TS-C2Rnn-P), were first trained on SED task and then on the localization task. In the first stage, all features are fed into the network to train for SED task and only the loss of SED is minimized. After SED have been trained, the learned weights from the convBlocks in the SED branch is transferred to the convBlocks in the localization branch to train for the localization task using the reference event labels to mask the localization predictions.

2.3. Data Augmentation: *MIXUP*

For better model generalization, we adopted *mixup* [10] a data augmentation technique which is a popular for image classification tasks. It has been illustrated in [10] that mixup scheme helps in alleviating undesirable behaviors of the deep neural network such as memorization and sensitivity to adversarial examples. *Mixup* is a data-agnostic data augmentation routine. It makes decision boundaries transit linearly from class to class, providing a smoother estimate of uncertainty.

The idea behind *mixup* is that of risk minimization. We wish to determine a function f that describes the relationship between input x_i and target y_i , and follows the joint distribution $P(x, y)$. We can minimize the average of the loss function ℓ (or expected risk) over P in the following manner

$$R(f) = \int \ell(f(x), y)dP(x, y)\,,$$

where $f(x)$ is a function that describes the relationship between input vector x and target vector y, ℓ is the loss function that penalizes the difference between the output of $f(x)$ and target y. While P is unknown is most practical cases, it can be approximated. There are two such approximations raised in [10] namely *empirical risk minimization* [11] and *vicinal risk minimization* [12]. While the vicinal virtual input-target pairs are generated by addition of Gaussian noise in [12], Zhang et al. [10] proposed the generation of virtual input and target pairs as such,

$$\begin{aligned} X &= \lambda \times x_1 + (1 - \lambda) \times x_2, \\ Y &= \lambda \times y_1 + (1 - \lambda) \times y_2, \end{aligned} \tag{1}$$

where λ is a weight drawn from the beta distribution with parameters $\alpha, \beta = 0.2$ and x_1, x_2, y_1 and y_2 are two pairs of input-target pairs drawn randomly from the dataset. The parameters α and β are chosen such that the probability density is denser in the domain $0 < \lambda < 0.1$ and $0.9 < \lambda < 1.0$ which can be seen in Figure 3. The average of the loss function can then be minimized over this probability distribution approximation.

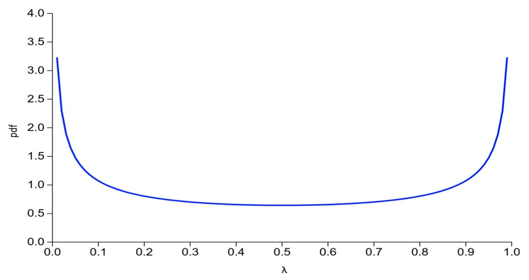


Figure 3: Beta distribution with $\alpha, \beta = 0.2$

3. EXPERIMENTS AND RESULTS

DCASE2019 Task-3 organizers has provided two datasets [7], TAU Spatial Sound Events 2019: Ambisonic, Microphone Array datasets of an identical sound scene with only difference in the format of the audio. In this work, we only used TAU Spatial Sound Events 2019: Microphone array dataset for all our experiments. The dataset consist of multiple audio recordings from 4 channel, directional microphones arranged in a tetrahedral array configuration with overlapping sound events recorded in different environments. Dataset is divided into two sets, development set and evaluation set. The development set consists of 400, one minute long recordings sampled at 48kHz, divided into four cross-validation splits of 100 recordings each. The evaluation set consists of 100, one-minute recordings. There are total 11 isolated classes of the sound events. We trained our network using this 4 pre-defined cross-validation folds and the final results are the overall aggregated from the test data of all 4 folds in the development set. The performance of the architecture is evaluated with individual metrics, for SED F-score and error rate (ER) was considered and for localization task, direction of arrival (DOA) error and frame recall (FR) were used. We trained our network with an objective to achieve lower DOA error & ER and higher FR & F-score.

Below is the list of prior arts and proposed architectures used for experiments and evaluations.

- **Baseline**, which is the benchmark model [5] released by DCASE-2019 Task-3 organizers. This network is based on the CRNN architecture, and take magnitude & phase spectrogram as input features.
- **SELDNet**, this network has the same architecture as in Baseline, but instead of magnitude & phase spectrogram, it takes log-mel spectrograms & GCC-PHAT as input features.
- **Two-Stage (TS)**, this network has CRNN architecture and is based on two stage training methodology [6] .
- **TS-CRnn**, same as our base network architecture TS-C2Rnn except only 1 GRU layer used as the RNN.
- **TS-C2Rnn**, our base network architecture as illustrated in Figure 1 and explained in section 2.2 .
- **TS-C2Rnn-P**, the modified version of our base network architecture TS-C2Rnn, which has 2×1 kernel size for pooling layers, as illustrated in Figure 2 and explained in section 2.2.

Table 1 presents the performance results on development set w.r.t to prior-arts for SED and localization task, with the effect of data augmentation technique, *mixup*. Compared with Baseline, our base network TS-C2Rnn without *mixup* shows 12.6% and 50.2% improvement on ER and F-score respectively for the SED task, while for localization task it show 20° and 2.6% improvement on DOA error and FR respectively. This result shows that deep CRNN based architectures improves the performance for SELD task compared to CNN based architecture.

In addition, TS-C2Rnn-P architecture which uses average pooling with kernel size of 2×1 in the CNN module, shows the best improvements with the best score across all evaluation metrics. For the SED task, TS-C2Rnn-P achieved an error rate of 0.149 and an F-score of 91.9%. For the localization evaluation metrics, it achieved a DOA error of 4.588 °and frame recall of 0.896. It shows improvement of 13% and 4° respectively on ER and DOA error, over the state-of-art Two-Stage(TS) network. This result infers that 2×1 pooling in the CNN module of TS-C2Rnn-P, helps it to learn the spectral information efficiently, and at the same time minimize the loss of information in the temporal dimension. In turn there is more information available to the RNN module, which helps in effectively learning the temporal information. This lead to boosting up

	no *mixup*				with *mixup*			
	ER	**F-score**	**DOA(°)**	**FR**	**ER**	**F-score**	**DOA(°)**	**FR**
Baseline	0.350	0.800	30.800	0.840	—	—	—	—
SELDNet	0.213	0.879	11.300	0.847	—	—	—	—
Two-Stage (TS)	0.166	0.908	9.619	0.863	0.194	0.888	8.901	0.839
TS-CRnn	0.186	0.897	9.450	0.857	0.200	0.888	7.866	0.841
TS-C2Rnn	0.174	0.901	8.881	0.862	0.176	0.903	7.236	0.856
TS-C2Rnn-P	0.147	0.916	5.631	0.902	0.149	0.919	4.588	0.896

Table 1: Performance evaluation of the proposed network architecture on the development set comparing with prior-arts

DCASE-2019 Task-3 Evaluation-result					
Team Name	**Rank**	**ER**	**F-score**	**DOA(°)**	**FR**
Kapka_SRPOL [13]	1	0.08	0.947	3.7	0.968
Cao_Surrey [14]	2	0.08	0.955	5.5	0.922
Xue_JDAI [15]	3	0.06	0.963	9.7	0.923
He_THU [16]	4	0.06	0.967	22.4	0.941
Jee_NTU (our)	5	0.12	0.937	4.2	0.918

Table 2: Comparison of top 5 results of DCASE-2019 Task-3.

of the overall performance of the proposed TS-C2Rnn-P network.

Table 1 also illustrate the effect of data augmentation i.e. *mixup* on the performance of above mention networks. Comparing the results we can infer that upon applying the *mixup* the F-score slightly dropped while the DOA error improved. We realized that the *mixup* is having positive effect on improving the localization task performance but at the same time it is showing a slight drop or no change in performance for the SED task. We applied a new training strategy of applying *mixup* only on the localization task during training, as there is no effect of *mixup* on SED task.

	mixup on localization task only			
	ER	**F-score**	**DOA(°)**	**FR**
Two-Stage	0.175	0.903	8.056	0.861
TS-C2Rnn	0.171	0.903	7.486	0.861
TS-C2Rnn-P	0.144	0.904	4.746	0.902

Table 3: Results from using *mixup* in localization branch training only.

Comparing between the performance of Two-Stage, TS-C2Rnn and TS-C2Rnn-P in Table 1 (no *mixup*) and Table 3 (*mixup* on localization task only), an improvement could be seen across all four evaluation metrics for all of the networks. In contrast while comparing the results of these network in Table 1 (with *mixup*) and Table 3, only three metrics showed an improvement while DOA error increased. This abnormality tell us that, training for localization task is built upon the trained weights of the SED task, therefore for improving the results in the localization branch, SED results are also essential. Although *mixup* appear to slightly drop in the performance score for SED predictions, but its negative effect on SED score, appears to have a positive performance surge on the localization task. The negative effects of *mixup* on the SED branch appeared to be suppressed by increasing the number of layers. This can be seen from the F-score converging to 0.904 for networks tested with *mixup* applications in Table 3.

The DOA error could be improved further by learning from the trained weights of a *mixup*-applied SED task instead of a non-*mixup*-applied SED task although that would adversely affect the results of SED predictions. Thus, a balance must be found in the use of *mixup*, depending on the use case and the allowance for error in SED and DOA predictions.

Table 2 presents the top 5 teams results on the DCASE-2019 Task 3 evaluation set. In this Table *"Jee_NTU"* refers to the results of TS-C2Rnn-P architecture proposed in this work. From the table we can infer that our DOA error performance, which is the rank 5 system has given a positive improvement over rank 2-4 systems. In addition our overall F-score for the SED task is comparable with other systems. With the usage of both the data sets provided by DCASE-2019 Task 3 and including mixup, we believe TS-C2Rnn-P can yield similar results as the top system.

4. CONCLUSION & FUTURE WORK

In this paper, we proposed CRNN architecture with *mixup* as data augmentation technique for SELD task. Experimentally, we have shown that using *mixup* helps in improving the localization performance. In addition, usage of rectangular kernels for the pooling layers helps in overall performance of SED and localization. Experimental results show that our proposed network architecture TS-C2Rnn-P with *mixup* is shown to significantly outperform the baseline system for both SED and localization task. For future studies, the changing of parameters α and β in *mixup* can be investigated. The parameters were chosen so as not to create too many vastly different virtual input-target pairs. There might be a beneficial improvement if the λ is less heavily weighted to one side of the input-target pair.

5. REFERENCES

[1] A. Kumar and B. Raj, "Deep cnn framework for audio event recognition using weakly labeled web data," *arXiv preprint arXiv:1707.02530*, 2017.

[2] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, *et al.*, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017, pp. 131–135.

[3] J. Vera-Diaz, D. Pizarro, and J. Macias-Guarasa, "Towards end-to-end acoustic localization using deep learning: From audio signals to source position coordinates," *Sensors*, vol. 18, no. 10, p. 3418, 2018.

[4] W. He, P. Motlicek, and J.-M. Odobez, "Deep neural networks for multiple speaker detection and localization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 74–79.

[5] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, "Sound event localization and detection of overlapping sources using convolutional recurrent neural networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, March 2019.

[6] Y. Cao, Q. Kong, T. Iqbal, F. An, W. Wang, and M. D. Plumbley, "Polyphonic sound event detection and localization using a two-stage strategy," *arXiv preprint arXiv:1905.00268*, 2019. [Online]. Available: https://arxiv.org/pdf/1905.00268.pdf

[7] S. Adavanne, A. Politis, and T. Virtanen, "A multi-room reverberant dataset for sound event localization and uetection," in *Submitted to Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019. [Online]. Available: https://arxiv.org/abs/1905.08546

[8] B. Logan *et al.*, "Mel frequency cepstral coefficients for music modeling." in *ISMIR*, vol. 270, 2000, pp. 1–11.

[9] J. Hassab and R. Boucher, "Performance of the generalized cross correlator in the presence of a strong spectral peak in the signal," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 3, pp. 549–555, 1981.

[10] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[11] V. N. Vapnik, "Statistical learning theory," vol. 1, 1998.

[12] O. Chapelle, J. Weston, L. Bottou, and V. Vapnik, "Vicinal risk minimization," in *Advances in neural information processing systems*, 2001, pp. 416–422.

[13] S. Kapka and M. Lewandowski, "Sound source detection, localization and classification using consecutive ensemble of crnn models," DCASE2019 Challenge, Tech. Rep., June 2019.

[14] Y. Cao, T. Iqbal, Q. Kong, M. Galindo, W. Wang, and M. Plumbley, "Two-stage sound event localization and detection using intensity vector and generalized cross-correlation," DCASE2019 Challenge, Tech. Rep., June 2019.

[15] W. Xue, T. Ying, Z. Chao, and D. Guohong, "Multi-beam and multi-task learning for joint sound event detection and localization," DCASE2019 Challenge, Tech. Rep., June 2019.

[16] J. Zhang, W. Ding, and L. He, "Data augmentation and prior knowledge-based regularization for sound event localization and detection," DCASE2019 Challenge, Tech. Rep., June 2019.

EXPLOITING PARALLEL AUDIO RECORDINGS TO ENFORCE DEVICE INVARIANCE IN CNN-BASED ACOUSTIC SCENE CLASSIFICATION

Paul Primus[1], *Hamid Eghbal-zadeh*[1,2], *David Eitelsebner*[1]
Khaled Koutini[1], *Andreas Arzt*[1], *Gerhard Widmer*[1,2]

[1]Institute of Computational Perception (CP-JKU) & [2]LIT Artificial Intelligence Lab,
Johannes Kepler University Linz, Austria
paul.primus@jku.at

ABSTRACT

Distribution mismatches between the data seen at training and at application time remain a major challenge in all application areas of machine learning. We study this problem in the context of machine listening (Task 1b of the DCASE 2019 Challenge). We propose a novel approach to learn domain-invariant classifiers in an end-to-end fashion by enforcing equal hidden layer representations for domain-parallel samples, i.e. time-aligned recordings from different recording devices. No classification labels are needed for our domain adaptation (DA) method, which makes the data collection process cheaper. We show that our method improves the target domain accuracy for both a toy dataset and an urban acoustic scenes dataset. We further compare our method to Maximum Mean Discrepancy-based DA and find it more robust to the choice of DA parameters. Our submission, based on this method, to DCASE 2019 Task 1b gave us the 4th place in the team ranking.

Index Terms— Domain Adaptation, Recording Device Mismatch, Parallel Representations, Acoustic Scene Classification

1. INTRODUCTION

Convolutional Neural Networks (CNNs) have become state of the art tools for audio related machine learning tasks, such as acoustic scene classification, audio tagging and sound event localization. While CNNs are known to generalize well if the recording conditions for training and unseen data remain the same, the generalization of this class of models degrades when there is a distribution dissimilarity between the training and the testing data [1].

In the following work we elaborate our findings for subtask 1b of 2019's IEEE DCASE Challenge, which is concerned with a domain mismatch problem. The task is to create an acoustic scene classification system for ten different acoustic classes. A set of labelled audio snippets recorded with a high-quality microphone (known as Device A) is provided for training. Additionally, for a small subset of samples from device A, parallel recordings from two lower quality microphones (devices B and C) are given. Evaluation of methods is done based on the overall accuracy on unseen samples from devices B and C. The acoustic scene, the city, and the device labels are provided for samples of the development set only. The main challenge of task 1b is to develop a model that, although trained mostly on samples from device A, is able to generalize well to samples from devices B and C. Since this problem is related to the field of *Domain Adaptation (DA)*, we refer to the distribution of device A samples as the *source domain*, and the distribution of samples of B and C devices as the *target domain*. In this work we explain how a state-of-the-art CNN model which by itself achieves high accuracy can be further improved by using a simple DA technique designed for problems where parallel representations are given.

2. RELATED WORK

Domain Adaptation (DA) is a popular field of research in transfer learning with multiple areas of application, e.g. bird audio detection [2]. Kouw et al. [3] distinguish between three types of data shifts which lead to a domain mismatches: prior, covariate and concept shift. In this work we focus on domain mismatches which are caused by covariate shifts (i.e., changes in feature distributions).

According to Shen et al. [4] solutions to domain adaptation can be categorized into three types: (i) Instance-based methods: reweight or subsample the source dataset to match the target distribution more closely [5]. (ii) Parameter-based methods: transfer knowledge through shared or regularized parameters of source and target domain learners [6], or by weighted ensembling of multiple source learners [7]. (iii) Feature-based methods: transform the samples such that they are invariant of the domain. Weiss et al. [8] further distinguish between symmetric and asymmetric methods. Asymmetric methods transform features of one domain to match another domain [9] symmetric feature-based methods embed samples into a common latent space where source and target feature distributions are close [10]. Symmetric feature-based methods can be easily incorporated into deep neural networks and therefore have been studied to a larger extent. The general idea is to minimize the divergence between source and target domain distributions for specific hidden layer representations with the help of some metric of distribution difference. For example, the deep domain confusion method [10] and deep adaptaion network [11] use Maximum Mean Discrepancy (MMD) [12] as a non-parametric integral probability metric. Other symmetric feature-based approaches exist that use adversarial objectives to minimize domain differences [13, 4]. These methods learn domain-invariant features by playing a minimax game between the domain critic and the feature extractor where the critic's task is to discriminate between the source and the target domain samples and the feature extractor learns domain-invariant and class-discriminative features. However, training the critic introduces more complexity, and may cause additional problems such as instability and mode collapse.

https://doi.org/10.33682/v9qj-8954

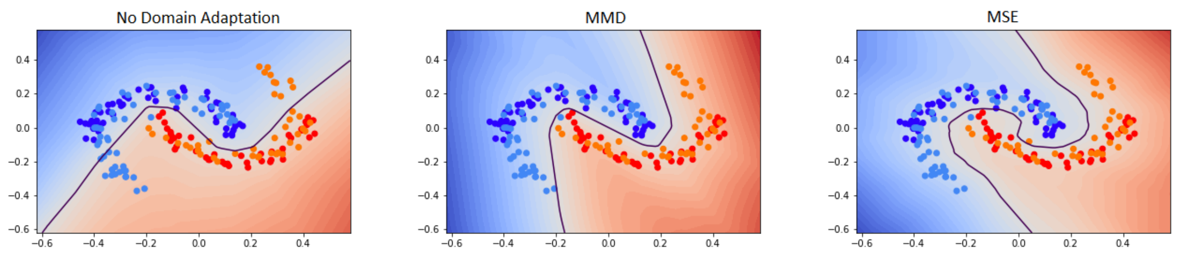


Figure 1: Two Moons dataset: best classifiers found by grid search over λ and n (Tab. 1). The source domain is represented by dark blue and red data points, the shifted target domain by light blue and orange points. The black line shows the decision boundary of a classifier trained without DA (left), with MMD-DA (middle) and with MSE-DA (right). Red and blue shaded areas represent decision areas of the classifiers.

3. DOMAIN-INVARIANT LEARNING

We propose a symmetric feature-based loss function to encourage the network to learn device-invariant representations for parallel samples from the source X^s, and the target domain X^t. This loss exploits the fact that parallel samples contain the same information relevant for classification and differ only due to a covariate shift, e.g. time-aligned spectrograms (x^s, x^t) contain the same information about the acoustic scenes and differ only due to device characteristics. Let $\phi_l(x^s)$ and $\phi_l(x^t)$ be d-dimensional hidden layer activations of layer l for paired samples x^s and x^t from the source and the target domains, respectively. A domain-invariant mapping $\phi_l(\cdot)$ projects both samples to the same activations without losing the class-discriminative power. To achieve this, we propose to jointly minimize classification loss $\mathcal{L}_{CL}$ and the Mean Squared Error (MSE) over paired sample activations, where the latter one is defined as

$$\mathcal{L}_{l,MSE} = \frac{1}{n \cdot d} \sum_{i=1}^{n} \left\| \phi_l(x_i^s) - \phi_l(x_i^t) \right\|_2^2 \tag{1}$$

for some fixed network layer l (this is a hyper-parameter). As we will show in Section 4 the DA mini-batch size n is critical, and our results suggest that bigger n yields better results. The final optimization objective we use for training is a combination of classification loss $\mathcal{L}_{CL}$ and DA loss $\mathcal{L}_{l,MSE}$:

$$\mathcal{L} = \mathcal{L}_{CL} + \lambda \mathcal{L}_{l,MSE} \tag{2}$$

Here, λ controls the balance between the DA loss and the classification loss during training. Note that for $\mathcal{L}_{l,MSE}$ no class label information is required and the labeled samples from all domains can be used for the supervised classification loss $\mathcal{L}_{CL}$.

4. EXPERIMENTS

In the following we evaluate the performance of our approach on the two moons dataset as well as on real-world acoustic data: the *DCASE 2019 Task 1b dataset* on acoustic scene classification [14]. We compare our proposed DA objective to the multi-kernel MMD-based approach used by Eghbal-zadeh et al. [15] for DCASE 2019 Subtask 1b. In all experiments, parallel samples are used without any class-label information. For both datasets, we find that when paired samples are given, MSE achieves higher accuracy on the target set compared to MMD.

	MSE				MMD			
$n \backslash \lambda$	0.1	1	5	10	0.1	1	5	10
8	.999	.999	.999	.999	.805	.862	.760	.749
32	.999	.999	.999	.999	.817	.771	.995	.990
128	.999	.999	.999	.999	.801	.859	.754	.739
256	.999	.999	.999	.999	.804	.861	.997	.744

Table 1: Domain Adaptation (DA) results on the two moons dataset: Accuracy on the target domain for models trained with different choices of DA loss, λ (columns) and n (rows). Baseline without DA is at 0.814.

4.1. Experimental Setup

We compare our approach to a baseline that uses the same CNN architecture and classification loss, but does not incorporate a DA loss. As another baseline, we use multi-kernel MMD-based DA [11], a non-parametric symmetric feature-based approach. MMD represents distances between two distributions as distances between mean embeddings of features in reproducing kernel Hilbert space $\mathcal{H}_k$:

$$d_k^2(X^s, X^t) = \left\| \mathbb{E}_{X^s}\left[k(\phi_l(x^s), \cdot)\right] - \mathbb{E}_{X^t}\left[k(\phi_l(x^t), \cdot)\right] \right\|_{\mathcal{H}_k}^2$$

The kernel k associated with the feature mapping for our experiments is a combination of four equally weighted RBF kernels with $\sigma \in \{0.2, 0.5, 0.9, 1.3\}$. We use the empirical version of this metric as DA loss, for which we randomly sample batches of size n from X^s and X^t. Therefore batches do not necessarily contain parallel representations of samples. Compared to our approach, MMD-based DA matches the distribution between the hidden representations of the source and the target domains, and not between the parallel representations. For both DA methods best results were obtained when applying the DA to the output layer. A plausible explanation is that using higher layer activations gives the network more flexibility for learning domain invariant representations.

4.2. Two Moons

Two moons (see Fig. 1) is a toy dataset often used in the context of transfer learning. It consists of two interleaved class distributions, where each is shaped like a half circle. We use this synthetic dataset to demonstrate our domain adaptation technique under controlled conditions.

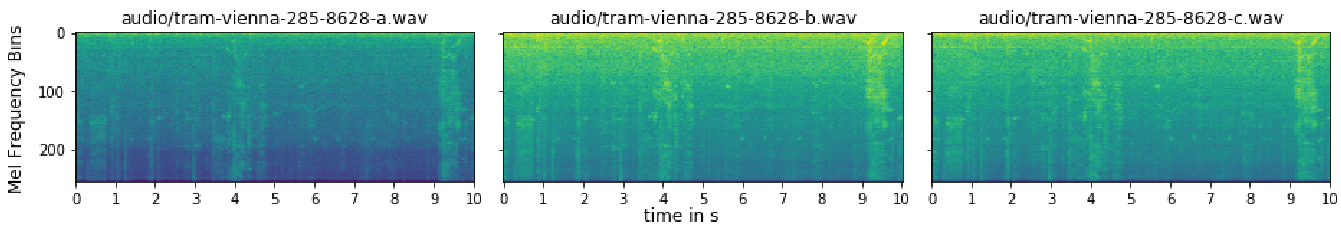


Figure 2: From left to right: Time-aligned recordings from devices A (Soundman OKM II Klassik/Studio A3 Microphone & Zoom F8 Recorder), B (Samsung Galaxy S7) and C (iPhone SE). Spectrograms show microphone-specifics, e.g. samples from devices B and C have more noise in lower Mel bins, compared to those from device A, and samples from device A seem to have fewer energy in all frequency bins.

4.2.1. Dataset & Architecture & Training

We utilize *sklearn* to generate two class-balanced two moons datasets with Gaussian distributed noise ($\mu = 0$ and $\sigma = 0.1$) and 10.000 samples each. Features are normalized to fall into the range of $[-0.5, 0.5]$. Domain-parallel representations are obtained by applying an artificial covariate shift to one of the two datasets. For our initial experiments we use two transformations: a stretching along the y-dimension by a factor of 1.5, and a rotation by -45 degrees (Fig. 1). We assume no label information is available for the parallel dataset. All experiments use a common model architecture, which is a fully connected network with one hidden layer of size 32 and ReLU activations. The output layer consists of one unit with a sigmoid activation function. The weights are initialized with He normal initialization [16]. We train for 250 epochs with mini-batches of size 32, binary cross-entropy loss, ADAM [17] update rule and constant learning rate of 0.001 to minimize Eq. 2.

4.2.2. Results

Without DA the model scores 81.4% accuracy on the target validation dataset (Fig. 1 left). To find a good parameter setting for both domain adaptation techniques we perform grid search over the DA weight $\lambda \in \{0.1, 1, 5, 10\}$ and the DA batch size $n \in \{8, 32, 128, 256\}$. Results are summarized in Table 1. At its best, MMD improves the accuracy on the target dataset to 99.7% (Fig. 1 middle). Regardless of the parameter combination the model with MSE-DA reaches 99.9% accuracy on both source and target domain validation sets. (Fig. 1 right). For all parameter configurations MSE-DA yields better results than MMD-DA.

4.3. Urban Acoustic Scene Dataset

The previous section has demonstrated that MSE-DA can be effectively used when domain-parallel representations are given. It is now necessary to evaluate our prior findings on a real world dataset, in our case the DCASE 2019 Task 1b dataset [14]. As explained in the introduction, our objective is to create a recording device invariant classifier by training it on a larger set of source domain samples and a few time-aligned recordings from the target domain. Fig. 2 shows three time-aligned recordings, for which we can observe the device-specific characteristics. Section 1 describes the DACSE 2019 task 1b in more details. An implementation of the following experiments is available on GitHub [1].

[1]`https://github.com/OptimusPrimus/dcase2019_task1b/tree/Workshop`

4.3.1. Dataset

The dataset contains 12.290 non-parallel device A samples and 3.240 parallel recorded samples (1080 per device). We use the validation setup suggested by the organizers, i.e. 9185 device A, 540 device B, and 540 device C samples for training and 4185 device A, 540 device B, and 540 device C for validation. Preprocessing is done similar to [18]: We resample the audio signals to 22050Hz and compute a mono-channel Short Time Fourier Transform using 2048-sample windows and a hop size of 512 samples. We apply a dB conversion to the individual frequency bands of the power spectrogram and a mel-scaled filterbank for frequencies between 40 and 11025Hz, yielding 431-frame spectrograms with 256 frequency bins. The samples are normalized during training by subtracting the source training set mean and dividing by the source training set standard deviation.

4.3.2. Network Architectures

We use the model architecture introduced by Koutini et al. [19], a receptive-field-regularized, fully convolutional, residual network (ResNet) with five residual blocks (Tab. 2). The receptive field of this architecture is tuned to achieve the best performance in audio-related tasks using spectrograms, as discussed in [19].

ResNet			
Type	#K	KS 1	KS 2
Conv+BN	128	5	
RB	128	3	1
Max Pool	-	2	-
RB	128	3	3
Max Pool	-	2	-
RB	128	3	3
RB	256	3	3
Max Pool	-	2	-
RB	512	3	1
Conv+BN	10	3	-
GAP	-	-	-

Residual Block (RB)	
Type	KS
Conv+BN	KS 1
Conv+BN	KS 2
Add Input	

Table 2: Model Architecture by [19] for experiments with the acoustic scenes dataset. #K and KS are the number of kernels and kernel size, respectively. Residual Blocks (RB) consist of two Convolutional (Conv) layers with #K kernels, each followed by a Batch Normalization (BN) layer. GAP is a Global Average Pooling Layer.

	MSE			MMD		
$n \setminus \lambda$	0.1	1	10	0.1	1	10
1	.494	.525	.488	-	-	-
8	.537	.592	.556	.467	.434	.412
16	.571	.592	.561	.456	.492	.233

Table 3: Domain Adaptation (DA) results on the acoustic scenes dataset: Accuracy on devices B and C for models trained with different choices of DA loss, λ (columns) and n (rows). Baseline model without DA scores .353 accuracy on the provided split.

4.3.3. *Training*

Although scene labels are available for all samples, we minimize $\mathcal{L}_{CL}$ over the 8.645 non-parallel device A samples only. The 1.620 time-aligned samples are used to learn domain-invariant features by minimizing pairwise DA loss $\mathcal{L}_{l,\cdot}$ between the three devices. For each update-step, we draw a batch from the non-parallel samples and a batch from the parallel samples to compute $\mathcal{L}_{CL}$ and $\mathcal{L}_{l,\cdot}$, respectively. We then minimize the sum of these two losses (Eq. 2). Models are trained for 120 epochs with non-parallel mini-batches of size 32, categorical cross-entropy loss, and ADAM [17] update rule to minimize Eq. 2. The initial learning rate is set to 10^{-3} and decreased by a factor 0.5 if the mean accuracy for devices B and C does not increase for 10 epochs. If the learning rate is decreased, we also reset the model parameters to the best model in terms of mean accuracy of device B and C up to the last epoch. We further use MixUp augmentation [20] with parameters of the beta-distribution set to $\alpha = \beta = 0.2$ for classification as well as DA samples.

4.3.4. *Results*

The baseline model without domain adaptation scores 35.3% BC-accuracy. We perform grid search over parameters $\lambda \in \{0.1, 1, 10\}$ and $n \in \{1, 8, 16\}$ to find a good combination for both MMD- and MSE-DA. The best model validation accuracy on device B and C (BC-accuracy) over all 120 epochs for each experiment is reported in Table 3. MMD-DA improves the BC-accuracy compared to the baseline without DA for all except one experiment. At its best MMD-DA achieves an BC-accuracy of 49.2%, which is an improvement by 13.9 $p.p.$ compared to the model trained without DA. Pairwise representation matching improves BC-accuracy even further: The best MSE-DA model scores 59.2% which is 23.9$p.p.$ above the baseline without DA.

4.4. DCASE Challenge 2019 Subtask 1b

In the following section we describe the adjustments made to our challenge submission to be more competitive. Our technical report describes the submitted systems in more detail [21].

4.4.1. *Datset & Cross-Validation & Training*

We split all audio segments into four folds, to have more domain parallel samples available for training. Furthermore, we minimize the classification loss over all available samples, including those from devices B and C. We increase the number of training and patience epochs to 250 and 15, respectively. For each fold, the model that scores the highest device BC-accuracy is selected for prediction on evaluation data. As we train every model on 4 folds, our final submission models are ensembles of the outputs of the 4 folds. For submission 1 and 2 we average the softmax predictions of each fold's best scoring model and select the class with the highest score. Submission 4 combines two independently trained models, again by averaging each of their 4 folds softmax outputs.

	Tr.\Te.	4-CV	K. Priv.	K. Pub.	Eval.
Ensemble	-	-	**.770**	**.766**	**.742**
MSE-DA	.644	.697	.762	.758	.734
No-DA	.612	.669	.705	.737	.713

Table 4: DCASE 2019 Task 1b results for different validation sets, from left to right: Device B and C validation accuracy (%) on the provided (Tr.\Te.) and custom split (4-CV), Kaggle private (K. Priv) and public leaderbord (K. Pub.), and the evaluation set (Eval.).

4.4.2. *Results*

The results of our challenge submission measured in BC-accuracy on unseen samples are reported in Table 4. The convolutional ResNet without DA achieved a BC-accuracy of 71.3% on the evaluation set, training on the suggested split achieved a BC-accuracy of 61.2%. The model used in submission three trained with MSE-DA loss gained an additional 2.1$p.p.$ on the evaluation set over the base model, resulting in an accuracy of 73.4%. A larger gain can be seen for the proposed split, as with 64.35 the model performed 3.15$p.p.$ better than our base model. Our ensemble of eight predictors achieves 74.2% BC-accuracy on the evaluation set which is our best result. The challenge submission by [15] which utilizes MMD-DA to learn device-invariant classifiers scores 74.5% on the final validation set, 0.3 $p.p.$ better than ours. The MM-DA used in [15] incorporates across-device mixup augmentation, is applied on a different architecture, integrates ensemble models, and uses a larger batch size, which explains the performance differences.

5. CONCLUSION & FUTURE WORK

In this report, we have shown how an already well-performing ResNet-like model [19] can be further improved for DCASE 2019 task 1b by using a simple DA technique. Our DA loss is designed to enforce equal hidden layer representations for different devices by exploiting time-aligned recordings. In our experiment we find that pointwise matching of representations yields better results, compared to minimizing the MMD between the hidden feature distributions without utilizing parallel representations. Notably, the MSE-DA increased the performance by 3.15 p.p. on the validation set of the proposed split, and by 2.1 p.p. on the final validation set. Furthermore, acquiring data for our method is cheap as it does not require labels for domain-parallel samples. In future work, we would like to investigate if data from unrelated acoustic scenes, i.e. scenes not relevant for classification, can be used to create device-invariant classifiers, as this would decrease cost even further.

6. ACKNOWLEDGMENT

This work has been supported by the COMET-K2 Center of the Linz Center of Mechatronics (LCM), funded by the Austrian federal government and the Federal State of Upper Austria.

7. REFERENCES

[1] H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "Deep within-class covariance analysis for robust deep audio representation learning," in *Neural Information Processing Systems, Interpretability and Robustness in Audio, Speech, and Language Workshop*, 2018.

[2] F. Berger, W. Freillinger, P. Primus, and W. Reisinger, "Bird audio detection - dcase 2018," DCASE2018 Challenge, Tech. Rep., September 2018.

[3] W. M. Kouw, "An introduction to domain adaptation and transfer learning," *CoRR*, vol. abs/1812.11806, 2018. [Online]. Available: http://arxiv.org/abs/1812.11806

[4] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 2018, pp. 4058–4065. [Online]. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17155

[5] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf, "Correcting sample selection bias by unlabeled data," in *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, 2006, pp. 601–608. [Online]. Available: http://papers.nips.cc/paper/3075-correcting-sample-selection-bias-by-unlabeled-data

[6] A. Rozantsev, M. Salzmann, and P. Fua, "Beyond sharing weights for deep domain adaptation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 4, pp. 801–814, 2019. [Online]. Available: https://doi.org/10.1109/TPAMI.2018.2814042

[7] L. Duan, D. Xu, and S.-F. Chang, "Exploiting web images for event recognition in consumer videos: A multiple source domain adaptation approach," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 1338–1345.

[8] K. R. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, p. 9, 2016. [Online]. Available: https://doi.org/10.1186/s40537-016-0043-6

[9] J. Hoffman, E. Rodner, J. Donahue, B. Kulis, and K. Saenko, "Asymmetric and category invariant feature transformations for domain adaptation," *International journal of computer vision*, vol. 109, no. 1-2, pp. 28–41, 2014.

[10] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," *CoRR*, vol. abs/1412.3474, 2014. [Online]. Available: http://arxiv.org/abs/1412.3474

[11] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, 2015, pp. 97–105. [Online]. Available: http://proceedings.mlr.press/v37/long15.html

[12] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, 2012. [Online]. Available: http://dl.acm.org/citation.cfm?id=2188410

[13] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7167–7176.

[14] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," *CoRR*, vol. abs/1807.09840, 2018. [Online]. Available: http://arxiv.org/abs/1807.09840

[15] H. Eghbal-zadeh, K. Koutini, and G. Widmer, "Acoustic scene classification and audio tagging with receptive-field-regularized CNNs," DCASE2019 Challenge, Tech. Rep., June 2019.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, 2015, pp. 1026–1034. [Online]. Available: https://doi.org/10.1109/ICCV.2015.123

[17] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[18] M. Dorfer, B. Lehner, H. Eghbal-zadeh, H. Christop, P. Fabian, and W. Gerhard, "Acoustic scene classification with fully convolutional neural networks and I-vectors," DCASE2018 Challenge, Tech. Rep., September 2018.

[19] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The Receptive Field as a Regularizer in Deep Convolutional Neural Networks for Acoustic Scene Classification," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*, A Coruña, Spain, 2019.

[20] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *CoRR*, vol. abs/1710.09412, 2017. [Online]. Available: http://arxiv.org/abs/1710.09412

[21] P. Primus and D. Eitelsebner, "Acoustic scene classification with mismatched recording devices," DCASE2019 Challenge, Tech. Rep., June 2019.

MIMII DATASET: SOUND DATASET FOR MALFUNCTIONING INDUSTRIAL MACHINE INVESTIGATION AND INSPECTION

Harsh Purohit, Ryo Tanabe, Kenji Ichige, Takashi Endo,
Yuki Nikaido, Kaori Suefusa, and Yohei Kawaguchi

Research and Development Group, Hitachi, Ltd.
1-280, Higashi-koigakubo, Kokubunji, Tokyo 185-8601, Japan
{harsh_pramodbhai.purohit.yf, yohei.kawaguchi.xk}@hitachi.com

ABSTRACT

Factory machinery is prone to failure or breakdown, resulting in significant expenses for companies. Hence, there is a rising interest in machine monitoring using different sensors including microphones. In scientific community, the emergence of public datasets has been promoting the advancement in acoustic detection and classification of scenes and events, but there are no public datasets that focus on the sound of industrial machines under normal and anomalous operating conditions in real factory environments. In this paper, we present a new dataset of industrial machine sounds which we call a sound dataset for malfunctioning industrial machine investigation and inspection (MIMII dataset). Normal and anomalous sounds were recorded for different types of industrial machines, i.e. valves, pumps, fans and slide rails. To resemble the real-life scenario, various anomalous sounds have been recorded, for instance, contamination, leakage, rotating unbalance, rail damage, etc. The purpose of releasing the MIMII dataset is to help the machine-learning and signal-processing community to advance the development of automated facility maintenance.

Index Terms— Machine sound dataset, Acoustic scene classification, Anomaly detection, Unsupervised anomalous sound detection

1. INTRODUCTION

The increasing demand for automatic machine inspection is attributable to the need for a better quality of factory equipment maintenance. The discovery of malfunctioning machine parts mainly depends on the experience of field engineers. However, shortage of field experts due to the increased number of requests for inspection has become an important problem in the industry. Therefore, an efficient and affordable solution to this problem is highly desirable.

In the past decade, industrial Internet of Things (IoT) and data-driven techniques have been revolutionizing the manufacturing industry, and different approaches have been undertaken for monitoring the state of machinery; for example, vibration sensor-based approaches [1–4], temperature sensor-based approaches [5], pressure sensor-based approaches [6], etc. Another approach is to detect anomalies from sound by using technologies for acoustic scene classification and event detection [7–13]. A remarkable advancement has been made in classification of acoustic scenes and detection of acoustic events, and there are many promising state-of-the-art studies [14–16]. We know that the emergence of numerous open benchmark dataset [17–20] is essential for the advancement of the research field. However, to the best of our knowledge, there is no public dataset which contains different types of machine sounds in real factory environments.

In this paper, we introduce a new dataset of machine sounds in normal and anomalous operating conditions in real factory environments. We include the sound of four machine types: (i) valves, (ii) pumps, (iii) fans, and (iv) slide rails. For each type of machine, we consider seven kinds of product models. We assume that the main task is to find an anomalous condition of the machine during a 10-second sound segment in an unsupervised learning situation. In other words, only normal machine sounds can be used in the training phase, and one has to correctly distinguish between a normal machine sound and an abnormal machine sound in the test phase. The main contributions of this paper can be summarized as follows: (1) We created an open dataset for malfunctioning industrial machine investigation and inspection (MIMII), first of its kind. (We will release this dataset by the workshop.) This dataset contains a total of 26,092 sound files for normal conditions of four different machine types. It also contains real-life anomalous sound files for each category of the machines. (2) Using our developed dataset, we have explored an autoencoder-based model for each type of machine with various noise conditions. These results can be taken as a benchmark to improve the accuracy of anomaly detection in the MIMII dataset.

The rest of the paper is organized as follows. In Section 2, we describe the recording environment and the setup. The details of the dataset content are given in Section 3. The autoencoder-based detection benchmark and results are discussed in Section 4. Section 5 concludes the paper.

2. RECORDING ENVIRONMENT AND SETUP

The dataset was collected using TAMAGO-03 microphone, manufactured by *System In Frontier Inc* [21]. It is a circular microphone array which consists of eight distinct microphones; the details of the microphone array are shown in Figure 1. By using the microphone array, not only single-channel-based approaches but also multi-channel-based ones can be evaluated. The microphone array was kept at a distance of 50 cm from the machine (10 cm in case of valves); 10-second sound segments were recorded. The dataset contains eight separate channels for each segment. Figure 2 depicts the recording setup with the direction and distance for each kind of machine. It should be noted that each machine sound was recorded in separate session. In running condition, the sound of the machine was recorded as 16-bit audio signals sampled at 16 kHz in a reverberant environment. Apart from the target machine sound, background noise in multiple real factories was continuously recorded

https://doi.org/10.33682/m76f-d618

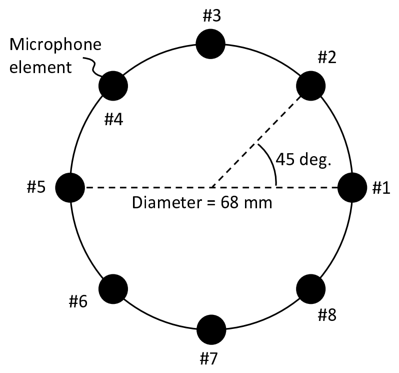


Figure 1: Circular microphone array

to mix it with the target machine sound for simulating real environments. For recording the background noise, we used the same microphone array as for the target machine sound.

3. DATASET CONTENT

The MIMII dataset contains the sound of four different types of machines: valves, pumps, fans, and slide rails. The valves are solenoid valves that are repeatedly opened and closed. The pumps are water pumps, which drained water from a pool and discharged water to the pool continuously. The fans represent industrial fans, which are used to provide a continuous flow or gas of air in factories. The slide rails in this paper represent linear slide systems, which consist of a moving platform and a stage base. The types of the sounds produced by the machines are stationary and non-stationary, have different features, and different degrees of difficulty. Figure 3 depicts a power spectrogram of the sound of all four types of machines, clearly showing that each machine has its unique sound characteristics.

The list of sound files for each machine type is reported in the Table 1. Each type of machines consists of seven individual machines. Individual machines may be of a different product model. We know that large datasets incorporating real-life complexity are needed to effectively train the models, so we recorded a total of 26,092 normal sound segments for all individual machines. In addition to this, different real-life anomalous scenarios have been considered for each kind of machine, for instance, contamination, leakage, rotating unbalance, rail damage, etc. Various running conditions are listed in Table 2. The number of sound segments for each anomalous sound for each different type of machine is small because we regard the main target of our dataset as an unsupervised learning scenario and regard the anomalous segments as a part of test data.

As explained in Section 2, the background noise recorded in multiple real factories was mixed with the target machine sound. Eight channels are considered separately during mixing the original sounds with the noise. For a certain signal-to-noise ratio (SNR) γ dB, the noise-mixed data of each machine model was made by the following steps:

1. The average power over all segments of the machine models, a, was calculated.
2. For each segment i from the machine model,

Table 1: MIMII dataset content details

Machine type/	model ID	Segments for normal condition	Segments for anomalous condition
Valve	00	991	119
	01	869	120
	02	708	120
	03	963	120
	04	1000	120
	05	999	400
	06	992	120
Pump	00	1006	143
	01	1003	116
	02	1005	111
	03	706	113
	04	702	100
	05	1008	248
	06	1036	102
Fan	00	1011	407
	01	1034	407
	02	1016	359
	03	1012	358
	04	1033	348
	05	1109	349
	06	1015	361
Slide rail	00	1068	356
	01	1068	178
	02	1068	267
	03	1068	178
	04	534	178
	05	534	178
	06	534	89
Total		26092	6065

Table 2: List of operations and anomalous conditions

Machine type	Operations	Examples of anomalous conditions
Valve	Open/close repeat with different timing	More than two kinds of contamination
Pump	Suction from/ discharge to a water pool	Leakage, contamination, clogging, etc.
Fan	Normal work	Unbalanced, voltage change, clogging, etc.
Slide rail	Slide repeat at different speeds	Rail damage, loose belt, no grease, etc.

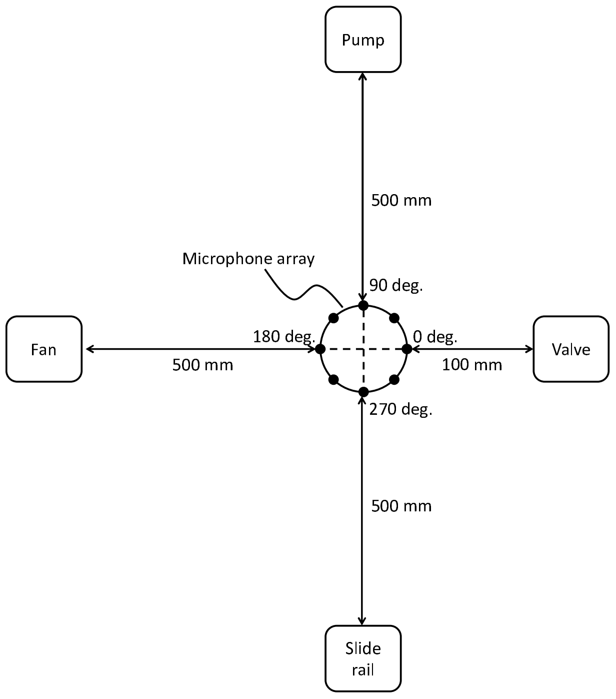


Figure 2: Schematic experimental setup for dataset recording

(a) a background-noise segment j is randomly selected, and its power b_j is tuned so that $\gamma = 10\log_{10}(a/b_j)$.

(b) The noise-mixed data is calculated by adding the target-machine segment i and the power-tuned background-noise segment j.

4. EXPERIMENT

An example of benchmarking is shown in this section. Our main goal is to detect anomalous sounds in an unsupervised learning scenario as discussed in Section 1, and several studies have successfully used autoencoders for unsupervised anomaly detection [12, 22–24], so an autoencoder-based unsupervised anomaly detector was evaluated.

We consider log-Mel spectrogram as an input feature. To calculate the Mel spectrogram, we consider a frame size of 1024, a hop size of 512 and a mel filter banks of 64 in our experiment. Five frames have been combined to initiate our 320 dimensional input feature vector $\mathbf{x}$. The parameters of the encoder and decoder neural networks (i.e. $\theta = (\theta_e, \theta_d)$) are trained to minimize the loss function given as follows:

$$L_{AE}(\theta_e, \theta_d) = \|\mathbf{x} - D(E(\mathbf{x} \mid \theta_e) \mid \theta_d)\|_2^2 \qquad (1)$$

Our basic assumption is that this trained model will give high reconstruction error for anomalous machine sounds. The autoencoder network structure for the experiment is summarized as follows: The encoder network ($E(\cdot)$) comprises $FC(Input, 64, ReLU)$; $FC(64, 64, ReLU)$; and $FC(64, 8, ReLU)$, and the decoder network ($D(\cdot)$) incorporates $FC(8, 64, ReLU)$; $FC(64, 64, ReLU)$ and $FC(64, Output, none)$ where $FC(a, b, f)$ means a fully-connected layer with a input neurons, b output neurons, and activation function f. The ReLUs are Rectified Linear Units [25]. The network is trained with Adam [26] optimization technique for 50 epochs.

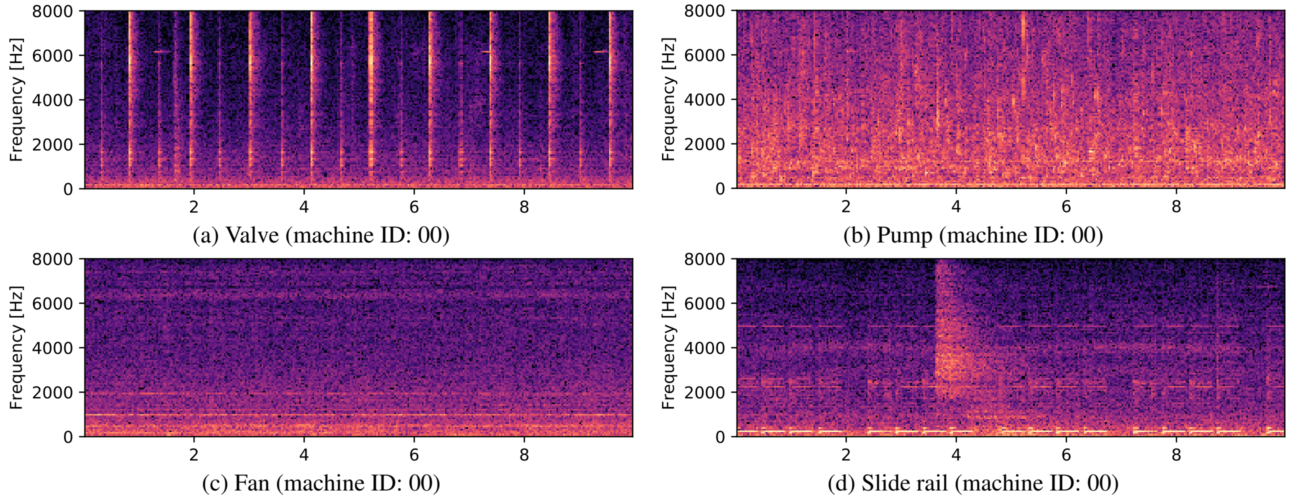
For each machine type and model ID, all the segments were split into a training dataset and a test dataset. All the anomalous segments were regarded as the test dataset, the same number of normal segments were randomly selected and regarded as the test dataset, and all the rest normal segments were regarded as the training dataset. By using the training dataset consisting only of normal ones, different autoencoders were trained for each machine type and model ID. Anomaly detection was performed for each segment by thresholding the reconstruction error averaged over 10 seconds, and the area under the curve (AUC) values were calculated for the test dataset for each machine type and model ID. In addition to this, we also considered different levels of SNR (with factory noise) in the experiment, for example, 6 dB, 0 dB, and -6 dB.

Table 3 shows the AUCs averaged over three training runs with independent initializations. In Table 3, It is clear that the AUCs for valves are lower than the other machines. Sound signals of valves are non-stationary, in particular, impulsive and sparse in time, and the reconstruction error averaged over time tends to be small. So, it is difficult to detect anomalies for valves. In contrast, it is easier to detect anomalies for fans than the other machines because sound signals of fans are stationary. Moreover, for some machine models, the AUC decreases rapidly as the noise level increases. These results indicate that it is important to solve the degradation caused by non-stationarity and noise for unsupervised anomalous sound detection.

5. CONCLUSION AND PERSPECTIVES

In this paper, we introduced the MIMII dataset, a real-world dataset for investigating the malfunctioning behavior of the industrial machines. We collected 26,092 sound segments of normal condition and 6,065 sound segments of anomalous condition and mixed the background noise recorded in multiple real factories with the machine-sound segments for simulating real environments. In addition, using the MIMII dataset, we showed an example of evaluation for autoencoder-based unsupervised anomalous sound detection. We observed that non-stationary machine sound signals and noise are the key issues for developing the unsupervised anomaly detector. These results can be taken as a benchmark to improve the accuracy of anomaly detection in the MIMII dataset.

We will release this dataset by the workshop. To the best of our knowledge, this dataset is the first of its kind to address the problem of detecting anomalous conditions in industrial machinery through machine sounds. As benchmarking is an important aspect in data driven methods, we strongly believe that our MIMII dataset will be very useful to the research community. We are releasing this data to accelerate research in the area of audio event detection, specifically for machine sounds. This dataset can be used for other use cases, for example, to restrict the training on specific number of machine models and then test on the remaining machine models. This study will be useful for measuring the domain adaptation capability of the different methods applied on machines from different manufactures. If the community finds interest in our dataset and validates its usage, we will improve the current version with the additional meta-data related to different anomalies.



(a) Valve (machine ID: 00) (b) Pump (machine ID: 00)

(c) Fan (machine ID: 00) (d) Slide rail (machine ID: 00)

Figure 3: Examples of power spectrograms on a normal condition at 6 dB SNR.

Table 3: AUCs for all machines

Machine type/		**Input SNR**		
model ID		**6 dB**	**0 dB**	**-6 dB**
Valve	00	0.68	0.55	0.62
	01	0.77	0.71	0.61
	02	0.66	0.59	0.57
	03	0.70	0.65	0.44
	04	0.64	0.65	0.50
	05	0.52	0.48	0.44
	06	0.70	0.66	0.53
	Avg.	0.67	0.61	0.53
Pump	00	0.84	0.65	0.58
	01	0.98	0.90	0.73
	02	0.45	0.46	0.52
	03	0.79	0.81	0.75
	04	0.99	0.95	0.93
	05	0.66	0.66	0.64
	06	0.94	0.76	0.61
	Avg.	0.81	0.74	0.68
Fan	00	0.75	0.63	0.57
	01	0.97	0.90	0.70
	02	0.99	0.83	0.68
	03	1.00	0.89	0.70
	04	0.92	0.75	0.57
	05	0.95	0.90	0.83
	06	0.99	0.97	0.83
	Avg.	0.94	0.84	0.70
Slide rail	00	0.99	0.99	0.93
	01	0.94	0.90	0.83
	02	0.93	0.79	0.74
	03	0.99	0.85	0.71
	04	0.88	0.78	0.61
	05	0.84	0.70	0.60
	06	0.71	0.56	0.52
	Avg.	0.90	0.80	0.70

6. REFERENCES

[1] M. Yu, D. Wang, and M. Luo, "Model-based prognosis for hybrid systems with mode-dependent degradation behaviors," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 546–554, 2013.

[2] T. Ishibashi, A. Yoshida, and T. Kawai, "Modelling of asymmetric rotor and cracked shaft," in *Proceedings of the 2nd Japanese Modelica Conference*, no. 148, 2019, pp. 180–186.

[3] E. P. Carden and P. Fanning, "Vibration based condition monitoring: A review," *Structural health monitoring*, vol. 3, no. 4, pp. 355–377, 2004.

[4] G. S. Galloway, V. M. Catterson, T. Fay, A. Robb, and C. Love, "Diagnosis of tidal turbine vibration data through deep neural networks," in *Proceedings of the 3rd European Conference of the Prognostics and Health Management Society*, 2016.

[5] G. Lodewijks, W. Li, Y. Pang, and X. Jiang, "An application of the IoT in belt conveyor systems," in *Proceedings of the International Conference on Internet and Distributed Computing Systems (IDCS)*, 2016, pp. 340–351.

[6] R. F. Salikhov, Y. P. Makushev, G. N. Musagitova, L. U. Volkova, and R. S. Suleymanov, "Diagnosis of fuel equipment of diesel engines in oil-and-gas machinery and facilities," *AIP Conference Proceedings*, vol. 2141, no. 1, p. 050009, 2019.

[7] Y. Koizumi, S. Murata, N. Harada, S. Saito, and H. Uematsu, "SNIPER: Few-shot learning for anomaly detection to minimize false-negative rate with ensured true-positive rate," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 915–919.

[8] Y. Kawachi, Y. Koizumi, S. Murata, and N. Harada, "A two-class hyper-spherical autoencoder for supervised anomaly detection," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3047–3051.

[9] M. Yamaguchi, Y. Koizumi, and N. Harada, "AdaFlow: Domain-adaptive density estimator with application to anomaly detection and unpaired cross-domain translation," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3647–3651.

[10] Y. Kawaguchi, R. Tanabe, T. Endo, K. Ichige, and K. Hamada, "Anomaly detection based on an ensemble of dereverberation and anomalous sound extraction," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 865–869.

[11] Y. Koizumi, S. Saito, H. Uematsu, Y. Kawachi, and N. Harada, "Unsupervised detection of anomalous sound based on deep learning and the Neyman–Pearson lemma," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 212–224, 2018.

[12] Y. Kawaguchi and T. Endo, "How can we detect anomalies from subsampled audio signals?" in *Proceedings of the IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2017, pp. 1–6.

[13] Y. Kawaguchi, "Anomaly detection based on feature reconstruction from subsampled audio signals," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*, 2018, pp. 2524–2528.

[14] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 Challenge," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 26, no. 2, pp. 379–393, 2018.

[15] S. S. R. Phaye, E. Benetos, and Y. Wang, "SubSpectralNet–using sub-spectrogram based convolutional neural networks for acoustic scene classification," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 825–829.

[16] Z. Podwinska, I. Sobieraj, B. M. Fazenda, W. J. Davies, and M. D. Plumbley, "Acoustic event detection from weakly labeled data using auditory salience," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 41–45.

[17] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio Set: An ontology and human-labeled dataset for audio events," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 776–780.

[18] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound datasets: A platform for the creation of open audio datasets," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2017, pp. 486–493.

[19] G. Dekkers, S. Lauwereins, B. Thoen, M. W. Adhana, H. Brouckxon, B. V. den Bergh, T. van Waterschoot, B. Vanrumste, M. Verhelst, and P. Karsmakers, "The SINS database for detection of daily activities in a home environment using an acoustic sensor network," in *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2017, pp. 32–36.

[20] Y. Koizumi, S. Saito, H. Uematsu, N. Harada, and K. Imoto, "ToyADMOS: A dataset of miniature-machine operating sounds for anomalous sound detection," in *Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, to appear.

[21] System In Frontier Inc. (http://www.sifi.co.jp/system/modules/pico/index.php?content_id=39&ml_lang=en).

[22] T. Tagawa, Y. Tadokoro, and T. Yairi, "Structured denoising autoencoder for fault detection and analysis," in *Proceedings of the Asian Conference on Machine Learning (ACML)*, 2015, pp. 96–111.

[23] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, "A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 1996–2000.

[24] D. Oh and I. Yun, "Residual error based anomaly detection using auto-encoder in SMD machine sound," *Sensors*, vol. 18, no. 5, p. 1308, 2018.

[25] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proceedings of the 12th IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 2146–2153.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

SOUND EVENT DETECTION AND DIRECTION OF ARRIVAL ESTIMATION USING RESIDUAL NET AND RECURRENT NEURAL NETWORKS

Rishabh Ranjan[1], *Sathish s/o Jayabalan*[1], *Thi Ngoc Tho Nguyen*[1], *Woon-Seng Gan*[1]

[1] Nanyang Technological University, Singapore, 679798
{rishabh001, sathishj, nguyenth003, ewsgan}@ntu.edu.sg

ABSTRACT

This paper presents deep learning approach for sound events detection and localization, which is also a part of detection and classification of acoustic scenes and events (DCASE) challenge 2019 Task 3. Deep residual nets originally used for image classification are adapted and combined with recurrent neural networks (RNN) to estimate the onset-offset of sound events, sound events class, and their direction in a reverberant environment. Additionally, data augmentation and post processing techniques are applied to generalize and improve the system performance on unseen data. Using our best model on validation dataset, sound events detection achieves F1-score of 0.89 and error rate of 0.18, whereas sound source localization task achieves angular error of 8° and 90% frame recall.

***Index Terms*—** Sound events detection, directional of arrival, residual net, recurrent neural networks

1. INTRODUCTION

Sound events localization and detection (SELD) system allows one to have automated annotation of a scene in spatial dimension and can assist stakeholders to make informed decisions. It is an important tool for various applications like identifying critical events like gunshots, accidents, noisy vehicles, mixed reality audio where spatial scene information enhanced the augmented listening, robots that listens just like humans and tracks the sound source of interest, smart homes and surveillance systems [1-5]. The three main objectives of SELD system are namely, (1) first, to detect presence of sound events, (2) second, to classify active sound events as textual labels, and (3) third, to estimate directions of active sound events.

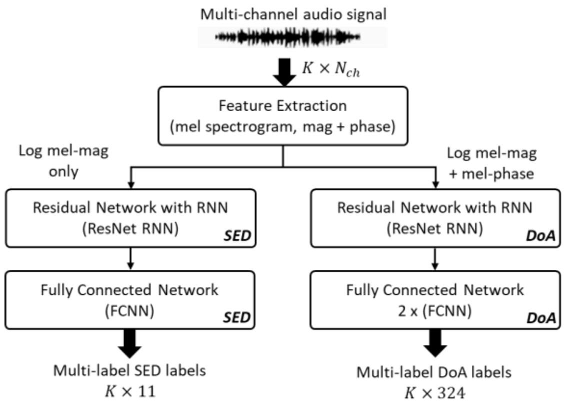
The first key component of the SELD system is sound event detection (SED), which assigns pre-defined labels to the active sound events every frame [6]. In the past, many signal processing and machine learning methods have been extensively applied to the SED problem using supervised classification approach. The most popular methods include, dictionary learning [7], gaussian or and hidden markov model [8-9], non-negative matrix factorization (NMF) [10-11], principal component analysis [12], and deep learning methods like fully connected neural network (FCNN) [13], convolutional neural network (CNN) [14-15], recurrent neural networks (RNN) [16], residual network (ResNet) [17]. Most recently, combination of the CNN, RNN and FCNN networks were also proposed to improve the SED performance and present state-of-art results [18-20]. Furthermore, multi-channel audio inputs as well as ambisonics data has been employed in SED task to exploit the spatial nature of the data [20-21].

The second key component of SELD system is direction of arrival (DoA), which estimates the directions of active sound events in terms of azimuth and/or elevations angles. DoA problem is commonly dealt using various traditional signal processing based methods: time-difference [22], subspace methods such as multiple signal classification (MUSIC) [23], cross-correlation methods such as generalized cross-correlation with phase transform (GCC-PHAT) [24], steered response with phase transform (SRP-PHAT) [25], multichannel cross-correlation coefficient (MCCC) [26]. However, some of the common practical challenges with these methods is performance degradation in presence of noisy and reverberant environment as well as high computational cost. Recently, deep learning based methods is also being extensively employed to improve the DoA performance and outperforms the traditional methods in challenging environments [27-35]. DNN based approaches vary in terms of microphone array geometry- circular, linear, binaural, ambisonics. In addition, different input features like GCC [33], magnitude and phase transform [21] [31], eigen vectors [34], inter-aural cross-correlation features [32] and most recently raw temporal features [35] have been used to improve the DoA performance. Furthermore, most of these works have been shown to work on only azimuthal plane sources and/or single static sources except [31], which demonstrates working in both azimuth and elevation as well as for overlapping sound sources.

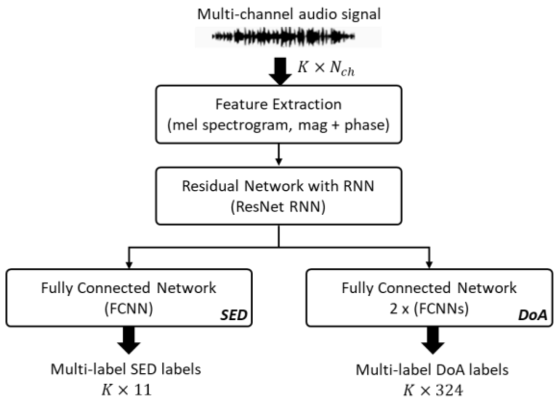
There are very few works jointly solving the SELD task using deep learning. Hirvonen [28] used spectral power of the multichannel audio signals from circular array and used CNN based classifier to predict one of the 8 source directions on azimuthal plane for each sound event. In contrast, Adavanne [21] employed regression based continuous DoA output in both azimuth and elevation for 11 different type of overlapping sound classes. The authors employed a joint network using CRNN network with two branches each for SED and DoA to perform the combined SELD task.

In this paper, we employ a ResNet architecture combined with RNN, referred as ResNet RNN, for the joint estimation of respective labels for SED and DoA for sound events in a reverberant scene with one or two active sound sources. In contrast to the baseline model [21], a classification-based output is employed for DoA and additional post-processing techniques are employed for both SED and DoA to further improve the overall SELD performance. The proposed model significantly outperforms the baseline model [21] using convolutional recurrent neural network (CRNN) specifically for the DoA task. In the next section, we give a detailed description of the proposed methodology and training set up.

https://doi.org/10.33682/93dp-f064


Multi-channel audio signal
$K \times N_{ch}$
Feature Extraction (mel spectrogram, mag + phase)
Log mel-mag only
Log mel-mag + mel-phase
Residual Network with RNN (ResNet RNN) SED
Residual Network with RNN (ResNet RNN) DoA
Fully Connected Network (FCNN) SED
Fully Connected Network 2 x (FCNN) DoA
Multi-label SED labels $K \times 11$
Multi-label DoA labels $K \times 324$

(a)


Multi-channel audio signal
$K \times N_{ch}$
Feature Extraction (mel spectrogram, mag + phase)
Residual Network with RNN (ResNet RNN)
Fully Connected Network (FCNN) SED
Fully Connected Network 2 x (FCNNs) DoA
Multi-label SED labels $K \times 11$
Multi-label DoA labels $K \times 324$

(b)

Figure 1: Proposed system overview (a) Individually trained models for SED and DoA (b) Jointly trained model

2. METHODOLOGY

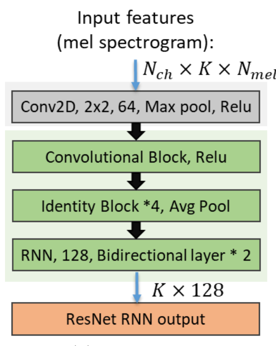
For the SELD task, two different configurations using a modified version of ResNet architecture combined with RNN are employed. Figure 1 shows overview block diagrams of the two system configurations. First system is using individually trained models for SED and DoA, where input is log mel magnitude spectrogram for the SED task, while log mel magnitude and linear mel phase spectrogram for the DoA as shown in Figure 1(a). Second system is using a jointly trained model, where ResNet RNN architecture is common for both SED and DoA and subsequently, divided into two branches using FCNN layers as shown in Figure 1(b). One key advantage of joint model in this work is that they share the common resources of ResNet RNN and therefore, would need less computational resource when implementing on embedded devices. DoA branch for both the configurations is further divided into two parallel branches consisting of FCNN layers and the two network outputs are combined as post-processing step to enhance the DoA accuracy. For both the systems, SED and DoA is predicted as continuous output in range [0 1] as probabilities for 11 distinct sound events and 324 unique directions, respectively. In the next subsections, we explain the dataset, feature extraction, model architecture, training set up, data augmentations and post-processing techniques used.

2.1. Development Dataset

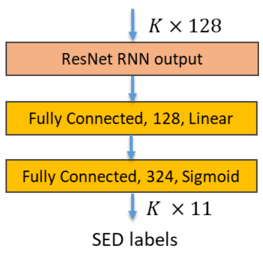
The development dataset is taken from detection and classification of acoustic scenes and events (DCASE) challenge 2019 task 3 for SELD task [36]. It consists of 4 splits and each split contains 100 audio files of length 60 sec and contains overlapping as well as non-overlapping sound events. Audio files is synthesized using 11 isolated sound labels taken from [37] and convolved with impulse responses (IR) measured from 5 different rooms at 504 unique combinations of azimuth-elevation-distance and finally, mixed with natural ambient noise collected at IR recording locations. In terms of unique target directions, there are 36 azimuths and 9 elevations resulting in total 324 directions. All the IRs were recorded using Eigenmike [38], a 32 microphone spherical array with only 4 of the microphones forming a tetrahedral shape were used for synthesis of DCASE 2019 task 3 dataset.

2.2. Feature Extraction

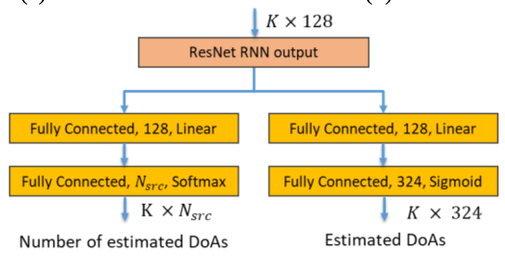
Each of the audio file is sampled at 48kHz and short-time Fourier transform (STFT) is applied with hop size of 20 msec. Next, STFT


Input features (mel spectrogram):
$N_{ch} \times K \times N_{mel}$
Conv2D, 2x2, 64, Max pool, Relu
Convolutional Block, Relu
Identity Block *4, Avg Pool
RNN, 128, Bidirectional layer * 2
$K \times 128$
ResNet RNN output

(a) ResNet RNN



(b) SED model


$K \times 128$
ResNet RNN output
Fully Connected, 128, Linear
Fully Connected, 128, Linear
Fully Connected, N_{src}, Softmax
Fully Connected, 324, Sigmoid
$K \times N_{src}$
$K \times 324$
Number of estimated DoAs
Estimated DoAs

(c) DoA model

Figure 2: Model architectures (a) ResNet RNN (b) SED: FCNN (c) DoA: Two parallel FCNN branch

spectrogram is converted to log mel magnitude spectrogram from amplitude of STFT and linear mel phase spectrogram from phase component of STFT using dot product of STFT component and mel-filter banks. After converting into mel spectrogram features, low and high frequency components are removed and finally, resized to match the input shape of the neural network before training.

2.3. Model Architecture

Figure 2(a) shows the architecture of proposed modified ResNet combined with RNN. The ResNet model is adapted from residual net model originally designed for image recognition and described in [39]. As shown in the figure, output of the feature extraction is fed to the ResNet RNN model with feature dimension of $N_{ch} \times K \times N_{mel}$, where N_{ch} is the number of channels (= 4 when only magnitude is used and 8 when both magnitude and phase is used as input feature), K is the number of frames used as sequence, and N_{mel} is the number of mel filter banks. The ResNet architecture consists of many 2D convolutional (conv) layers, however the distinct feature of ResNet architecture is the use of identity and convolutional block with skip connection to solve the vanishing gradient problem in deeper networks [39]. In this work, ResNet is a 2-stage architecture with first stage being a 2D conv layer with 64 filters, followed by batch normalization of outputs [40], 'ReLU' activation function and dimensionality reduction using max pooling of 2 along the mel frequency axis. Second stage consists of one convolutional block with three filters with output size as (64, 64, 256), and 4 identity blocks with three filters and same number of filters as in convolutional block. Finally, average pooling of size 16 is applied along the mel frequency axis. Subsequently, output from stage 2 is reshaped on the last two dimensions before feeding to two RNN layers to learn the contextual information from temporal sequence data of K frames. Each RNN layer consists of 128 nodes of either gated recurrent units (GRU) or long short-term memory (LSTM) with 'tanh' activation function. RNN block is followed by fully connected dense layers for both SED and DoA as shown in Figure 2(b) and (c). First FC layer in both the tasks consists of 128 nodes with linear activation function and dropout of 0.5 to improve the generalization ability of network. Final FC layer in SED consists of 11 nodes corresponding to 11 unique target sound classes with sigmoid activation function as shown in Figure 2(b). DoA, however, consists of two parallel branches of FC layers with one branch estimating number of active sources and other branch estimating actual direction estimates as probabilities. Final FC layer in first branch consists of N_{src} nodes corresponding to maximum number of active sources with 'softmax' activation function. For the second DoA branch, final FC layer consists of 324 nodes corresponding to 324 unique directions with 'sigmoid' activation function.

2.4. Model Training

For model development, 4 cross-fold sets from DCASE challenge 2019 task 3 dataset [4] is used with 3 of the splits used for training and one split for validation as shown in Table 1. During training, each processed audio feature file is split into sequence length of 128 frames and resized with fixed batch size of 96. For SED, binary cross-entropy loss function is used for model weights adaptation. For DoA second branch, weighted binary cross-entropy loss function is used to strongly penalize the false negatives because at most only two out of 324 DoA labels are true at any time frame in the ground truth. For both SED and DoA, adam optimizer is used with learning rate of 0.0005. Best model is saved using the combined SELD loss metric computed using the evaluation metrics provided by DCASE task 3 organizers and briefly explained in sub-section 2.7.

2.5. Data Augmentation

To improve model generalization capability on unseen test data, data augmentation using frame shifting is applied to each of the processed audio file. Each audio feature set is shifted in negative time by 32, 64 and 96 frames across temporal dimension before splitting into sequence of 128 frames. In this way, we create 3 shifted copies of audio segments, which helps in generalizing the model performance. Therefore, total data after augmentation is 4 times larger than the original dataset size and each audio feature file including shifted copies are selected randomly for training in each epoch.

Table 1: Cross-fold configuration for model evaluation

Fold	Training sets	Validation sets
1	Split 2, 3, 4	Split 1
2	Split 3, 4, 1	Split 2
3	Split 1, 2, 4	Split 3
4	Split 1, 2, 3	Split 4

2.6. Output Post-processing

First post-processing technique applied to both SED and DoA outputs is by predicting on frame shifted audio feature sequences and then, taking geometric mean of the shifted probability estimates:

$$\mathbf{p}_{avg} = \sqrt[3]{\mathbf{p}(t_0) \cdot \mathbf{p}(t_1) \cdot \mathbf{p}(t_2)} \, . \tag{1}$$

where $\mathbf{p}(t_i)$ is the probabilities predicted using the final trained model weights for each audio feature file $\mathbf{X}(t)$ shifted by t_i frames and padding zeros in front and excluding first t_i frames from the predicted probabilities as final estimates:

$$\mathbf{p}(t_i) = predict([\mathbf{0}(t_i) \quad \mathbf{X}(t - t_i)]). \tag{2}$$

Above averaging method helps in averaging out the spurious outliers in the final prediction. It is also found that geometric mean gives slightly better results than arithmetic mean and thus, were used to compute SED and DoA output probabilities.

Final SED labels were obtained frame wise by comparing the output probabilities for each label with a given threshold. Those labels with probabilities more than the threshold are selected as active sound events and in the case of none of the labels' probabilities more than threshold no activity, i.e., ambience is assigned. For DoA estimations, we merge the outputs of two branches as explained in following sub subsection.

2.6.1. DoA Post-processing

As explained earlier in sub-section 2.3 and Figure 2(c), there are two outputs from DoA model as number of active sources and 324 direction labels probabilities. To obtain final estimated directions per frame, we take the following steps:

1. Convert the DoA output 324 probabilities estimate into 2D array with size 36 azimuths × 9 elevations
2. Find the local peaks in the 2D array above a given threshold and a minimum neighboring distance between two peaks
3. Compute n_{src} as number of active sources by selecting label with maximum probability in the first DoA branch.
4. Select n_{src} peaks from the output of second step as final DoA estimate.

By using above post-processing steps of peak finding with minimum neighboring constraint, we filter out the redundant DoA peaks which are close by and also improve the DoA frame recall by capping the number estimated DoAs based on first branch output. Finally, both SED and DoA outputs are combined together frame wise based on the presence of active sound events or directions in any of the SED or DoA outputs. In the case of multiple sources, to match the DoA and SED outputs, we take into account the precedence of single source SED and DoA outputs in previous time frames and use this prior information to match the second source outputs in current time frame.

Table 2: Proposed ResNet RNN model Vs Baseline CRNN model performance fold-wise for training on 3-splits

Fold	Model	ER	F-Score	DoA Error (°)	FR (%)
1	**Proposed-I**	**0.1640**	**89.83**	**8.72**	**91.36**
	Proposed-J	0.2479	85.10	12.08	89.77
	Baseline	0.3055	82.96	30.13	85.42
2	**Proposed-I**	**0.1903**	**89.13**	**8.55**	**90.0**
	Proposed-J	0.2716	84.32	12.58	86.57
	Baseline	0.3273	82.17	31.32	82.44
3	**Proposed-I**	**0.1611**	**90.71**	**7.76**	**91.28**
	Proposed-J	0.2543	85.33	13.35	89.62
	Baseline	0.2676	84.93	31.75	86.12
4	**Proposed-I**	**0.2143**	**86.77**	**7.72**	**90.7**
	Proposed-J	0.3084	82.03	12.09	88.47
	Baseline	0.2937	82.64	31.05	87.23
over-all	**Proposed-I**	**0.1824**	**89.10**	**8.19**	**90.84**
	Proposed-J	0.2706	84.18	12.53	88.61
	Baseline	0.2986	83.16	31.06	85.30

2.7. Evaluation Metrics

Model performance is evaluated using 4 metrics, 2 each for SED and DoA. SED is evaluated using error rate (ER) and F-score. ER is the total error based on total number of insertions (I), deletions and substitutions [41]. F-score is calculated as harmonic mean of precision and recall [41]. DoA is evaluated using average angular error and frame recall (FR). DoA error is defined as average angular error in degrees between estimated and ground truth directions and computed using Hungarian algorithm [42] to account for the assignment problem of matching the individual estimated direction with respective reference direction. DoA FR is defined as percentage of frames where number of estimated and reference directions are equal out of total frames. In addition, combined SED, DoA and SELD metrics were computed using mean of respective error metrics and used for evaluating models.

3. RESULTS

Table 2 shows the performance of two proposed models: individually trained models (Proposed-I) and jointly trained models (Proposed-J). Clearly, the Proposed-I models outperforms the baseline model in terms of all the 4 metrics and for all validation splits. Specifically, there is significant overall improvement in terms of DoA angular error from 31° for baseline to 8.2° for the individually trained models. However, jointly trained models do not perform as good as the Proposed-I models but yet provides noticeable improvement over baseline, especially for DoA. Poor performance for Proposed-J model can be explained by the fact that by using shared ResNet layers' trained weights may not be optimal for either SED and DoA because of joint training. On the other hand, for individual models, respective weights for both SED and DoA ResNet layers are optimally trained and thus, giving better performance. Additionally, joint model incurred around 1.4 million parameters against 3 million parameters for combined individual SED and DoA model. Clearly as mentioned earlier, joint models require less computational resource and therefore, would ensure faster prediction time as compared to individual models for an SELD system running in real-time on an embedded device.

Table 3: Proposed ResNet RNN model Vs Baseline CRNN model performance for Ov1 and Ov2

Fold	Model	ER	F-Score	DoA Error (°)	FR (%)
Ov1	Proposed-I	**0.1571**	**91.26**	**3.90**	**97.07**
	Proposed-J	0.2077	88.86	6.6	93.96
	Baseline	0.2834	85.32	26.41	93.23
Ov2	Proposed-I	**0.1954**	**87.93**	**10.49**	**84.60**
	Proposed-J	0.3029	81.64	15.59	83.25
	Baseline	0.3064	82.00	33.70	77.38

Table 4: Proposed-I model performance for 5 RIRs

IR	ER	F-Score	DoA Error (°)	FR (%)
IR1	0.1728	89.49	8.31	90.46
IR2	0.1940	88.57	7.85	92.06
IR3	0.1737	89.83	7.69	90.22
IR4	0.1788	89.33	8.36	91.49
IR5	0.1937	88.24	8.73	89.95

Table 3 shows the proposed models performance for single source (Ov1) and two overlapping sources (Ov2). Proposed models performs much better for single source scenario as compared to two sources, especially with DoA error as low as 3.9° and FR as high as 97 %. Proposed model performance for 5 different room impulse responses is also summarized in Table 4. Except for the IR5 and IR3 in terms of SED ER, proposed models perform similar across all the IRs.

4. CONCLUSION

In this paper, a 2-stage ResNet architecture combined with RNN is used for both sound events classification and localization task. With data augmentation and post-processing techniques, the proposed model performance is significantly improved, especially for the DoA task with error as low as 8^o and frame recall of 90 %. The proposed work is also demonstrated in DCASE challenge 2019 Task 3 and showed superior performance over baseline on evaluation dataset. Jointly trained model is useful for edge implementations because of lower complexity but at the cost of sub-optimal performance. This needs to be further investigated and has been identified as future work to further improve the performance of joint model.

5. ACKNOWLEDGMENT

This research was conducted in collaboration with Singapore Telecommunications Limited and supported by the Singapore Government through the Industry Alignment Fund - Industry Collaboration Projects Grant.

The Authors are also thankful to Maggie Leong at amazon web services Singapore to generously provide the resources for model development on the cloud.

6. REFERENCES

[1] N. Yalta, K. Nakadai, and T. Ogata, "Sound source localization using deep learning models," in *Journal of Robotics and Mechatronics*, vol. 29, no. 1, 2017

[2] R. Radhakrishnan, A. Divakaran, and P. Smaragdis, "Audio analysis for surveillance applications," in *IEEE Worksh. on Apps. of Signal*

Processing to Audio and Acoustics (WASPAA'05), New Paltz, NY, USA, Oct. 2005, pp. 158–161.

[3] C. Mydlarz, J. Salamon, and J. P. Bello, "The implementation of low-cost urban acoustic monitoring devices," *Applied Acoustics*, vol. In Press, 2016.

[4] W. He, P. Motlicek, and J.-M. Odobez, "Deep neural networks for multiple speaker detection and localization," in *International Conference on Robotics and Automation (ICRA)*, 2018.

[5] C. Grobler, C. Kruger, B. Silva, and G. Hancke, "Sound based localization and identification in industrial environments," in *IEEE Industrial Electronics Society (IECON)*, 2017.

[6] J.-J. Aucouturier, B. Defreville, and F. Pachet, "The bag-of-frames approach to audio pattern recognition: A sufficient model for urban soundscapes but not for polyphonic music," *J. Acoust. Soc. America*, vol. 122, no. 2, pp. 881–891, 2007.

[7] J. Salamon and J. P. Bello, "Unsupervised feature learning for urban sound classification," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Brisbane, Australia, Apr. 2015, pp. 171–175.

[8] A. Mesaros, T. Heittola, A. Eronen, and T. Virtanen, "Acoustic event detection in real life recordings," in *Signal Processing Conference, 2010 18th European. IEEE*, 2010, pp. 1267–1271.

[9] X. Zhuang, X. Zhou, M. A. Hasegawa-Johnson, and T. S. Huang, "Real-world acoustic event detection," *Pattern Recognition Letters*, vol. 31, no. 12, pp. 1543–1551, 2010.

[10] B. Cauchi, "Non-negative matrix factorisation applied to auditory scenes classification," M.S. thesis, ATIAM, Paris Tech, Paris, France, Aug. 2011

[11] J. F. Gemmeke, L. Vuegen, P. Karsmakers, B. Vanrumste, et al., "An exemplar-based nmf approach to audio event detection," in *2013 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. IEEE, 2013, pp. 1–4.

[12] E. Benetos, "Automatic transcription of polyphonic music exploiting temporal evolution," Ph.D. dissertation, School of Electron. Eng. And Comput. Sci., Queen Mary University of London, London, U.K., Dec. 2012.

[13] E. Cakır, T. Heittola, H. Huttunen, and T. Virtanen, "Polyphonic sound event detection using multi-label deep neural networks," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2015.

[14] H. Phan, L. Hertel, M. Maass, and A. Mertins, "Robust audio event recognition with 1-max pooling convolutional neural networks," in *INTERSPEECH*, 2016.

[15] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[16] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 6440–6444.

[17] S. Hershey, C. Sourish, E. P. W. Daniel, G. F. Jort, J. Aren, M. R. Channing, P. Manoj. "CNN architectures for large-scale audio classification." In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pp. 131-135. IEEE, 2017.

[18] S. Adavanne, A. Politis, and T. Virtanen, "Multichannel sound event detection using 3D convolutional neural networks for learning inte channel features," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2018.

[19] E. C¸ akır, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, 2017.

[20] S. Adavanne, P. Pertila, and T. Virtanen, "Sound event detection using ¨spatial features and convolutional recurrent neural network," in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017.

[21] S. Adavanne, P. Pertila, N. Joonas and T. Virtanen. "Sound Event Localization and Detection of Overlapping Sources Using Convolutional Recurrent Neural Networks." *IEEE Journal of Selected Topics in Signal Processing* (2018).

[22] Y. Huang, J. Benesty, G. Elko, and R. Mersereati, "Real-time passive source localization: a practical linear-correction least-squares approach," in *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 8, 2001.

[23] R. O. Schmidt, "Multiple emitter location and signal parameter estimation," in *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 3, 1986.

[24] C. Knapp, and C. Glifford. "The generalized correlation method for estimation of time delay." *IEEE transactions on acoustics, speech, and signal processing* 24, no. 4 (1976): 320-327.

[25] DiBiase, Joseph Hector. *A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays*. PhD thesis, Brown University, 2000.

[26] J. Benesty, J.D. Chen, and Y.T.Huang, "Time delay estimation via linear interpolation and cross correlation," *IEEE Transactions on speech and audio processing*, vol. 12, no. 5, pp. 509–519, 2004.

[27] Q. Li,Z. Xueliang, and L. Hao. "Online Direction of Arrival Estimation Based on Deep Learning." In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2616-2620. IEEE, 2018.

[28] T. Hirvonen. "Classification of spatial audio location and content using convolutional neural networks." In *Audio Engineering Society Convention 138*. Audio Engineering Society, 2015.

[29] C. Pang, L. Hong, and L. Xiaofei. "Multitask Learning of Time-Frequency CNN for Sound Source Localization." *IEEE Access* 7 (2019): 40725-40737.

[30] S. Chakrabarty, and H. AP. Emanuël. "Broadband DOA estimation using convolutional neural networks trained with noise signals." In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 136-140. IEEE, 2017.

[31] S. Adavanne, A. Politis, and T. Virtanen. "Direction of arrival estimation for multiple sound sources using convolutional recurrent neural network." In *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 1462-1466. IEEE, 2018.

[32] M. Yiwere and E. J. Rhee, "Distance estimation and localization of sound sources in reverberant conditions using deep neural networks," in International Journal of Applied Engineering Research, vol. 12, no. 22, 2017.

[33] X. Xiao, S. Zhao, X. Zhong, D. L. Jones, E. S. Chng, and H. Li, "A learning-based approach to direction of arrival estimation in noisy and reverberant environments," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[34] R. Takeda and K. Komatani, "Sound source localization based on deep neural networks with directional activate function exploiting phase information," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[35] J. Vera-Diaz, P. Daniel, and M. G. Javier. "Towards End-to-End Acoustic Localization Using Deep Learning: From Audio Signals to Source Position Coordinates." *Sensors* 18, no. 10 (2018): 3418.

[36] DCASE Challenge Task 3: http://dcase.community/challenge2019/task-sound-event-localization-and-detection

[37] TUT audio dataset: http://www.cs.tut.fi/sgn/arg/dcase2016/task-sound-event-detection-in-synthetic-audio#audio-dataset

[38] Eigenmike: https://mhacoustics.com/products

[39] K. He, Z. Xiangyu, R. Shaoqing, and S. Jian. "*Deep residual learning for image recognition."* In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *International Conference on Machine Learning*, 2015.

[41] A. Mesaros, T. Heittola, and T. Virtanen. "*Metrics for polyphonic sound event detection."* Applied Sciences 6, no. 6 (2016): 162.

[42] H. W. Kuhn, "The hungarian method for the assignment problem," *in Naval Research Logistics Quarterly*, no. 2, 1955, p. 8397.

OPEN-SET EVOLVING ACOUSTIC SCENE CLASSIFICATION SYSTEM

Fatemeh Saki, Yinyi Guo, Cheng-Yu Hung

Lae-Hoon Kim, Manyu Deshpande, Sunkuk Moon, Eunjeong Koh, and Erik Visser

Qualcomm Technologies, Inc., San Diego, USA

ABSTRACT

Most audio recognition/classification systems assume a static and closed-set model, where training and testing data are drawn from a prior distribution. However, in real-world audio recognition/classification problems, such a distribution is unknown, and training data is limited and incomplete at training time. As it is difficult to collect exhaustive training samples to train classifiers. Datasets at prediction time are evolving and the trained model must deal with an infinite number of unseen/unknown categories. Therefore, it is desired to have an open-set classifier that not only accurately classifies the known classes into their respective classes but also effectively identifies unknown samples and learns them. This paper introduces an open-set evolving audio classification technique, which can effectively recognize and learn unknown classes continuously in an unsupervised manner. The proposed method consists of several steps: a) recognizing sound signals and associating them with known classes while also being able to identify the unknown classes; b) detecting the hidden unknown classes among the rejected sound samples; c) learning those novel detected classes and updating the classifier. The experimental results illustrate the effectiveness of the developed approach in detecting unknown sound classes compared to extreme value machine (EVM) and Weibull-calibrated SVM (W-SVM).

***Index Terms*—**Acoustic scene classification, open-set recognition, support vector data description

1. INTRODUCTION

Research on acoustic scene classification (ASC) has been receiving increased attention over the past decade, which has led to a considerable amount of new sound modeling and recognition techniques. ASC plays a major role in machine hearing systems. Where, the primary goal is achieving human-like auditory recognition of ambient sound signals [1, 2]. Some example applications include context-aware devices that automatically adjust their operation mode according to surrounding sounds, such as hearing aid devices that their speech enhancement parameters are adjusted depending on the background noise type [3]. Some other applications are robotics [4], monitoring elderly people, and acoustic monitoring systems in smart homes for detecting events such as glass breaking, baby crying, and gunshot [5].

One limitation of the existing ASC systems is their closed-set nature, that is a fixed and limited number of known classes are used during the training. In closed-set classifiers, it is assumed that during test time, the test data is drawn from the same set of classes as the training data. This guarantees that every input samples are classified into exactly one of the training classes. However, most applications for ASCs in nature are open-set problems. In other words, in an open-set framework, the test data could include samples associated with unknown classes as well. Therefore, it is necessary for an ASC to detect if a sound signal is associated with an unknown category. It is also desired to learn the unknown instances that appear more frequently. To our knowledge, the following contributions are the only existing ASC systems that partially implement the open-set framework. In [6] an open-set ASC is proposed to detect unknown classes utilizing support vector data description (SVDD). This method is only able to detect unknown samples without being able to learn them. In [7], a real-time unsupervised model for learning environmental noise signals is developed. This model can detect unknown classes in the stream of input sound signals and learn them on the fly. However, it can only store data from one unknown class and create one class at a time.

In this work, we propose a solution to overcome the limitations of both systems. Our proposed technique can identify the unknown sound signals and classify them into multiple micro-clusters based on the similarity of their characteristics. We then prune the micro-clusters according to a popularity measure, such that the micro-clusters larger than a certain threshold are classified into new classes on the fly. Importantly, our proposed method has no limitation on the number of classes created on the fly.

The rest of the paper is organized as follows. A brief overview of the open-set problem is covered in Section 2. The proposed open-set evolving acoustic scene classification model is then presented in Section 3. Section 4 presents the experimental results followed by the conclusion in Section 5.

2. OPEN-SET RECOGNITION MODEL

In this section, we first briefly state the preliminaries related to open-set recognition (OSR), following which we formally define the evolving open-set problem in Section 3.

Traditional recognition/classification algorithms are closed-set problems where all training and testing data are known a priori. Closed-set classifiers have been developed that maximize the optimal posterior probability, $p(C_i|x; C_1, C_2, \ldots C_M), i \in \{1,2, \ldots, M\}$, where x is an input sample, i is the index of the known class C_i, and M in the number of the known classes. However, a practical automatic recognition/classification problem is an open-set problem,

https://doi.org/10.33682/en2t-9m14

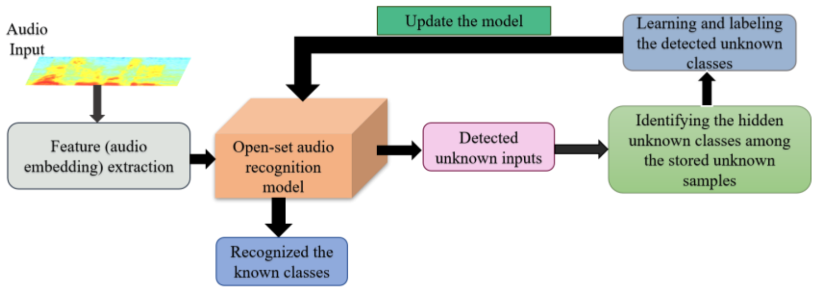

Audio Input
Feature (audio embedding) extraction
Open-set audio recognition model
Recognized the known classes
Detected unknown inputs
Identifying the hidden unknown classes among the stored unknown samples
Learning and labeling the detected unknown classes
Update the model

Figure 1: Block diagram of the developed open-set evolving acoustic scene classification system

where during testing, data from unknown classes can emerge at any time unexpectedly. Therefore, during the test, the optimal posterior probability becomes $p(C_j|x; C_1, C_2, \dots C_M, C_{M+1}, \dots C_{M+Q}), j \in \{1,2,\dots,M+Q\}$, with Q being the number of unknown classes [8]. This posterior probability cannot be modeled as classes C_{M+1} through C_{M+Q} are unknown.

In [9], the OSR problem was formalized for the first time and a preliminary solution was proposed. It incorporates an open space O risk term to account for the space beyond the reasonable support of the known classes. Open space is a space that is sufficiently far from the known classes. Let f be a measurable function where $f_y(x) \geq 0$ implies recognition of the known class y and $f_y(x) = 0$ implies, input x does not belong to class y. The open space risk $R_o(f)$ for a class y can be defined as follows [9]:

$$R_o(f_y) = \frac{\int_O f_y(x)dx}{\int_{S_o} f_y(x)dx} \qquad (1)$$

where S_o is the space that contains all the known training classes as well as the open space O. The objective OSR function must balance the open space risk against empirical error.

The OSR problem has been studied in various frameworks [10-12]. In [12] the existing open-set techniques are categorized into five main categories. 1) deep neural network-based, 2) adversarial learning-based, 3) extreme value theory-based, 4) Dirichlet process-based, and 5) traditional machine learning-based models. Although much effort has been made to develop promising solutions for OSR problem, the flexibility of these methods in continuously learning new classes, at a lower computational complexity, is still a challenge. Addressing this challenge using traditional machine learning approaches is the focus of this paper.

3. MULTI-CLASS OPEN-SET EVOLVING ACOUSTIC SCENE CLASSIFICATION MECHANISM

A dynamically evolving classification system needs to continuously detect unknown classes and learn them at a low training cost. The key components of such a system are as follows: 1) accurately assigning the input samples from the known classes into their respective labels, 2) rejecting samples that are from unknown classes, 3) keeping track of the rejected unknown samples to identify potential new classes among all the rejected samples in 2), and 4) learning/labeling the detected new classes and expanding the existing model at a low training cost.

In this work, similar to [7], known classes are labeled with positive integers and unknown samples are temporarily labeled as 0. Assuming $x \in \mathbb{R}^d$ to be an input sample in the feature domain, the proposed open-set evolving model is defined as follows.

Definition (Multi-class Open-set Evolving Recognition (MCOSR) Model): A solution to a multi-class open-set evolving recognition:

1. A multi-class open-set model $F(x): \mathbb{R}^d \mapsto \mathbb{Z}^{\geq 0}$ is an ensemble of multiple OSR functions, $f_{C_i}(x) : \mathbb{R}^d \mapsto \mathbb{R}$, $i = 1, \dots, M$, where M is the number of known classes.
2. Let $\mathcal{X}_{Nov} := \{x_k | x_k \in \mathbb{R}^d, k = 1, \dots, \mathcal{K}\}$ be a dataset of $\mathcal{K}$ samples, that are detected as unknown. $L(\mathcal{X}_{Nov}): \mathbb{R}^d \mapsto \mathbb{N}$ is a novelty detection process that is applied to $\mathcal{X}_{Nov}$ to determine the existence of new classes in $\mathcal{X}_{Nov}$.
3. If $L(\mathcal{X}_{Nov})$ discovers Q potential new classes, the existing MCOSR model is expanded by adding the discovered classes to the previously learned class set. Thus, the set of known classes becomes $\{C_1, C_2, \dots, C_M\} \cup \{C_{M+1}, C_{M+2}, \dots, C_{M+Q}\}$. It is worth noting that M varies in time as a result of the evolution process.

The diagram of the proposed MCOSR is shown in Fig. 1. Details of each step are introduced in the following subsections.

3.1. Multi-class open-set evolving recognition function

This section states the details of the proposed open-set acoustic scene classification. The algorithm consists of four main steps, feature extraction, classification/rejection, new class detection, and model evolution as follows.

3.1.1. Feature extraction:

The spectrogram of the input audio signal is first passed through a pre-trained neural network (details discussed in section 4) to extract the embedding representation. In this work, an L3-Net-based [13] audio embedding network is used as the feature extractor. The extracted embeddings are used as the input feature vectors to the MCOSR model.

3.1.2. Classification/rejection

The extracted embeddings from an input sound file are passed into the multi-class open-set recognition model $F(x): \mathbb{R}^d \mapsto \mathbb{Z}^{\geq 0}$ to determine if the input sound signal belongs to any of the known classes, C_i, or it is an unknown sample (0). $F(x)$ is an ensemble of multiple OSR functions, $f_{C_i}(x) : \mathbb{R}^d \mapsto \mathbb{R}$, $i = 1, \dots, M$, where M is the number of known classes. Each of the OSR functions characterizes one of the known classes utilizing a support vector data description (SVDD) model [14].

SVDD is a kernel-based sphere-shaped data description method that provides an effective description of the data boundary in the feature space. SVDD has been investigated in the context of various open-set problems [15,16]. The objective of SVDD is to find the smallest hypersphere that encloses most of the data in feature space $\mathcal{X}$. Let $\mathcal{X} \coloneqq \{x_j | x_j \in \mathbb{R}^d, j = 1, \dots, \mathcal{J}\}$ be a dataset of $\mathcal{J}$ points. Using a nonlinear transformation φ from $\mathcal{X}$ to a high-dimensional kernel feature space, the smallest enclosing hypersphere of radius R and center α can be stated as:

$$\min_{R,\alpha,\xi} R^2 + \frac{1}{\gamma}\sum_j \xi_j \tag{2}$$

$$s.t. \left\|\varphi(x_j) - \alpha\right\|^2 \leq R^2 + \xi_j, \quad \xi_j \geq 0, \quad \forall j. \tag{3}$$

The slack variables $\xi_j \geq 0$, associated with each training sample x_j, allow a soft boundary and hyperparameter $\gamma \in (0,1]$ establishes a trade-off between the sphere volume and the accuracy of data description. To optimize α, R and ξ_j [14] a Lagrangian procedure is used. The local maximum of the Lagrange function can be written as:

$$\begin{aligned} \mathcal{L} = \sum_{j=1}^{\mathcal{J}} \beta_j \varphi(x_j).\varphi(x_j) - \sum_{j,k=1}^{\mathcal{J}} \beta_j \beta_k \varphi(x_j).\varphi(x_k) \\ s.t. \sum_j \beta_j = 1, \ \alpha = \sum_j \beta_j \varphi(x_j), 0 \leq \beta_j \leq \gamma \end{aligned} \tag{4}$$

where $\beta_j \geq 0$ represent Lagrange multipliers. Samples with $\beta_j = 0$ lie inside the sphere surface, while those with $\beta_j = \gamma$ fall outside. Samples with $0 < \beta_j < \gamma$ are on the boundary of the corresponding hypersphere. It can be seen from (4), the center of the sphere (α) is a linear combination of the data samples. To describe the hypersphere, only samples with $\beta_j > 0$ are needed, hence they are called support vectors. R^2 is the distance from the center of the sphere (α) to (any of the support vectors on) the boundary, excluding the ones outside the sphere.

Therefore, given a set of $\mathcal{J}$ data samples, the open-set recognition function for representing it as a class/hypersphere C_i is defined as:

$$f_{C_i}(x) = \|\varphi(x) - \alpha_i\|^2 - R_i^2 \tag{5}$$

Input x is associated with class C_i, if its distance to the center of the sphere C_i ,i.e. α_i, is equal or smaller than the radius R_i^2, i.e. $f_{C_i}(x) \leq 0$. Therefore, the decision mechanism of identifying the label of the input sample x in MCOSR is as follows:

$$C^* = \begin{cases} arg \min_i f_{C_i}(x) & f_{C_i}(x) \leq 0, \forall\, i = 1, \dots, M \\ 0 & otherwise \end{cases} \tag{6}$$

The C^* is the output label, where 0 stands for unknown samples.

To minimize misclassifications, majority voting of 3 decisions is considered. This way if inputs at time t and $t+2$, x_t and x_{t+2} are both from class ℓ, then x_{t+1} is expected to be from the same class, noting that signal data occur in a streaming manner and the interest is primarily on the sustained type of sound.

3.1.3. Model evolution with detected new classes

Samples that are detected as unknown are stored in a buffer. Length of this buffer should be larger than a minimum number of samples, $\mathcal{D}$, required for establishing a new class. Let $\mathcal{X}_{Nov} \coloneqq \{x_k | x_k \in \mathbb{R}^d, k = 1, \dots, \mathcal{K}\}$ be the set of $\mathcal{K}$ samples in the buffer. Among these stored unknown samples, we need to determine if there is any consolidated ensemble that should be declared as a new separate class. A similarity measure is used to assess such consolidation. Let $L(.): \mathbb{R}^d \mapsto \mathbb{N}$ to be the process of detecting the existence of a new class using that similarity measure. We use cosine similarity measure and denote a pair of data points as similar if their cosine similarity is greater than a predefined threshold value λ. In this work, this value is set empirically to $\lambda = 0.85$. An ensemble of similar samples is called micro-cluster. A sample is assigned to a micro-cluster if its similarity not only with the center of the micro-cluster, but also with each sample from a randomly selected set, is greater than λ. If not assigned to any of the existing micro-clusters, the sample will form a sporadic micro-cluster. Micro-clusters with size less than two are considered sporadic. Once the size of a micro-cluster exceeds $\mathcal{D}$, an open-set recognition function, e.g. SVDD, is used to model it as a new class. To set $\mathcal{D}$, one needs to specify how long an occurring sound to be sustained in order to establish a new class. In other words, how long sound data from a scene class is needed to establish a new class. Finally, when a new class is created, the pre-trained model gets updated accordingly. The newly created class is labeled as the number of existing classes plus one.

4. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of MCOSR algorithm in terms of OSR accuracy, and qualitatively analyze its evolving aspects. We then perform a comparative study of the OSR accuracy of the proposed method against the existing state-of-the-art OSR algorithms such as EVM [8], Weibull-calibrated SVM (W-SVM) [17], PI-SVM and PI-OSVM (one-class SVM) [18]. EVM has a well-grounded interpretation derived from the statistical extreme value theory (EVT) and is the first classifier capable of performing nonlinear kernel-free learning. In W-SVM, the decision scores are used to fit the data into a single Weibull distribution, and a specific threshold is set to reject the unknown classes [8].

Data from the DCASE 2018 task5, a subset of SINS [19], TUT Acoustic Scenes 2017 [20], and Audioset [21] are used to evaluate the proposed algorithm. The dataset comprises 12 acoustic scenes, which are “absence”, “cooking”, “dishwashing”, “eating”, “social activity”, “watching TV”, “car”, “music”, “restaurant”, “transport”, “vacuum cleaner”, and “working”.

OpenL3 [22] is used for extracting the audio embeddings, considering the default settings and a 512-embedding dimension. The input to the L3-Net is a 10-second audio signal and the extracted embedding feature is a matrix of size 96×512. The averages of the extracted embeddings from the 10-second sound files are used as inputs to the MCOSR.

We begin by pre-training a model using data from the following five acoustic scenes “cooking”, “dishwashing”, “eating”, “social activity”, and “watching TV”. This leads to five pre-trained classes, which we will refer to as C1, C2, …, C5, respectively. For pre-training, we used 30 audio embedding samples, i.e. $\mathcal{D}$, extracted from five minutes of sound signals per acoustic scene. The remaining duration of data associated with the five pre-training scenes, mentioned above, along with the full duration of the data associated with the other seven acoustic scenes, i.e. “absence”, “car”, “music”, “restaurant”, “transport”, “vacuum cleaner”, and “working”, are used for testing.

4.1. Multi-class open-set recognition

We evaluate the OSR aspect of the MCOSR in two steps. In the first step, the proposed system identifies if an input sample is associated with the pre-trained classes and if so, it labels them as known samples. In the second step, the system classifies the known samples into their respective classes, which is known as closed-set recognition. Denote by TP, FP, TN, and FN the true-positive, false-positive, true-negative and false-negative, respectively. We measure the

Table 1. Accuracy of MCOSR, EVM, W-SVM(Linear), PI-OSVM, and PI-SVM in detecting the known classes and rejecting the unknown classes (%), in terms of TPR (higher, better) and 1-FPR (higher, better)

Methods	Detecting known: TPR	Rejecting unknown: 1-FPR
MCOSR	**88.22**	**93.03**
W-SVM (Linear)	**59.58**	**96.61**
EVM	**97.08**	**91.9**
PI-OSVM	21.7	15.68
PI-SVM	60.8	51.41

accuracy of the first step in the form of true-positive rate, given by $TPR = TP/(TP + FN)$, and specificity, that is $1 - FPR = TN/(TN + FP)$. The results are reported in Table 1, where each entry is an average of a 10-fold cross-validation process.

Aside from MCOSR results, Table 1 also reports the results for comparable algorithms in the literature, including EVM, W-SVM, PI-SVM, and PI-OSVM. The rejection threshold, i.e. δ, for EVM is set to 0.05, which is deduced empirically. At first glance, it may appear that W-SVM outperforms other algorithms since its specificity is slightly higher than that of MCOSR and EVM. However, it drastically underperforms, in terms of TPR, compared to MCOSR and EVM. It is worth mentioning that, EVM has the highest TPR while its specificity is close to the MCOSR. Importantly, as reported in Table 2, EVM suffers from large confusion error in close-set recognition. Our analyses suggest that PI-OSVM and PI-SVM provide the lowest performance.

4.2. The performance of the open-set technique as a function of class-set evolution

In this section, we study the performance of our proposed algorithm while changing the order in which the data are fed to the system. Our contribution is two-fold. First, we investigate the effect of choosing the initial sound samples on the number of created classes during the test and their accuracy. Next, we measure the overall post-evolution accuracy of the system, in terms of a confusion matrix.

To evaluate the effect of the initial sound samples on the performance of the proposed MCOSR technique, we used a pre-trained model comprising C1, C2, C3, C4, and C5. During the test, the inputs to the pre-trained model are the samples from known and unknown classes. The experiment is repeated 10 times while randomly shuffling the sound files associated with each class. Samples that are detected as unknown are stored in a buffer. The $L(.)$ process is continuously assessing the existence of a new class among the stored unknown samples. Once it detects a new class, MCOSR to be updated with this new class and the set of known classes is expanded. It is expected the forthcoming samples from the newly added class to be assigned to it and not identified as unknown samples anymore. However, in some of the experiments, MCOSR creates multiple classes for "music" class. Because the music data is not coherent enough, the initially created music class by MCOSR does not provide enough representative information of the music class. Therefore, other incoming music samples are identified as unknown samples and later will be learned as a new class. Data from the "working" sound scene is also a challenging one. In some experiments, no new class is created for this scene, as data from this scene is always misclassified by one of the known classes, mostly the "absence" class. Also, the samples from this scene that are identified as unknown are not similar enough to create meaningful micro-clusters to be declared as a new class.

As mentioned earlier, we next measured the post-evolution accuracy of the system. The results are reported in Table 3 in terms of a confusion matrix. In this table "absence", "car", "music", "restaurant", "transport", "vacuum cleaner", and "working" scenes are referred as C6, C7, …, C12, respectively. It was found as the number of classes increases in the model; the confusion rate increases. Because SVDD ignores the discriminative information between the known classes, which leads to poor classification performance. This issue needs to be addressed to achieve a higher classification accuracy when dealing with more complex acoustic scenes.

5. CONCLUSION

This paper provides an open-set evolving audio scene classification technique, MCOSR, which can effectively recognize and learn unknown acoustic scenes in an unsupervised manner. The developed model is evaluated utilizing the DCASE challenge dataset, TUT Acoustic Scenes 2017, and music files from Audioset. Experimental results demonstrate the effectiveness of the developed approach in identifying unknown samples compared to EVM, W-SVM, PI-OSVM, and PI-SVM. This paper exemplifies how the proposed MCOSR method can be used as a proper evolving open-set system for sound classification applications. Future research will focus on addressing practical issues during run time.

Table 2. Confusion matrices of the proposed MCOSR and EVM for Table 1.

	cooking		dishwashing		eating		social activity		watching TV		unknown	
	MCOSR	EVM	MCOSR	EVM	MCOSR	EVM	MCOSR	EVM	MCOSR	EVM	MCOSR	EVM
cooking	**81.54**	**58.33**	3.97	27.08	0	4.17	0	0	0	0	14.49	10.42
dishwashing	3.08	16.67	**88.21**	**62.5**	3.72	20.83	0	0	0	0	5	0
eating	0	8.33	1.03	12.5	**96.41**	**79.17**	0	0	0	0	2.56	0
social activity	0	6.25	0	0	1.28	4.17	**86.28**	**87.5**	0	0	12.44	2.08
watching TV	0.13	2.08	0	0	0	2.08	0.9	6.25	**89.1**	**87.5**	9.87	2.08
unknown	0.04	7.79	0.04	0.31	5.48	0	0	0	0.15	0	**94.29**	**91.9**

Table 3. Accuracy of the developed open-set evolving acoustic scene recognition model (%), after all classes have been learnt on the fly

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	unknown
C1	**73.21**	4.36	0	0	0	0.13	1.03	0	0	0	0	0.51	20.77
C2	2.05	**88.46**	3.08	0	0	0	0	0	0	0	0	0	6.41
C3	0	1.67	**81.67**	0	0	4.36	0	0	0	0	0	3.08	9.23
C4	0.13	0.38	0.51	**84.62**	0	0	0	0	1.15	0.38	0	0	12.82
C5	0	0	0	0.13	**89.74**	0	0	0	0	0	0.51	0.51	9.1
C6	0.13	0	0.26	0	0	**94.49**	0	0	0	0	0	0.13	5
C7	0	0	0	0	0	1.28	**92.95**	0	0	0.26	0	0	5.51
C8	0	0	0.9	0	0	0	0	**94.62**	0.64	0	0	0	3.85
C9	0	0	0	0	0	0	0	0.9	**97.56**	0	0	0	1.54
C10	0	0	0	0.77	0	0	0.26	0	0	**92.56**	0	0	6.41
C11	0	0	0	0	0	0	0	0	0	0.77	**91.15**	0	8.08
C12	0.63	2.7	13.81	0	0.63	29.05	0	0	0	0	0	**19.68**	33.49

6. REFERENCES

[1] S. McAdams, “Recognition of sound sources and events,” *in Thinking in Sound,: the Cognitive Psychology of Human Audition*. London: Oxford Univ. Press, 1993.

[2] A. S. Bregman. *Auditory Scene Analysis: The Perceptual Organization of Sound*. Cambridge, MA, USA: MIT Press, 1990.

[3] I. Panahi, N. Kehtarnavaz, and L. Thibodeau, “Smartphone-based noise adaptive speech enhancement for hearing aid applications,” *Proceedings of the 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 85-88, Orlando, August 2016.

[4] S. Aziz , M. Awais, T. Akram, U. Khan, M. Alhussein, and K. Aurangzeb, “Automatic scene recognition through acoustic classification for behavioral robotics,” *Electronics* 2019, vol. 8, no.5, 483.

[5] T .Virtanen, M. D. Plumbley, and D. Ellis. “Introduction to sound scene and event analysis,” *Computational Analysis of Sound Scenes and Events*, pp. 3-12, Springer, Cham, 2018.

[6] D. Battaglino, L. Lepauloux, and N. Evans, “The open-set problem in acoustic scene classification,” *International Workshop on Acoustic Signal Enhancement (IWAENC)*, pp. 1-5, 2016.

[7] F. Saki, N. Kehtarnavaz “Real-time unsupervised classification of environmental noise signals,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pp. 1657-1567, 2017.

[8] E. M. Rudd, L. P. Jain, W. J. Scheirer, and T. E. Boult, “The extreme value machine,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 762-768, Mar. 2018.

[9] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, “Toward open set recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1757-1772, Jul. 2013.

[10] P. Phillips, P. Grother, and R. Micheals, “Evaluation methods on face recognition,” *Handbook of Face Recognition*, A. Jain and S. Li, eds., pp. 329-348, Springer, 2005.

[11] L. Fayin and H. Wechsler, “Open set face recognition using transduction,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1686-1697, Nov. 2005.

[12] Ch. Geng, Sh. Huang, and S. Chen, “Recent advances in open set recognition: a survey,” http://arxiv.org/abs/1811.08581.

[13] R. Arandjelovi´c and A. Zisserman, “Look, listen and learn,” *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 609–617, 2017.

[14] D. M. Tax and R. P. Duin, “Support vector data description,” *Machine learning*, vol. 54, no. 1, pp. 45-66, 2004.

[15] A. Banerjee, P. Burlina, and C. Diehl. “A support vector method for anomaly detection in hyperspectral imagery,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 8, pp. 2282-2291, 2006.

[16] F. Alegre, A. Amehraye, and N. Evans. “A one-class classification approach to generalised speaker verification spoofing countermeasures using local binary patterns,” *In Proceedings of the IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pp. 1-8, 2013.

[17] W. J. Scheirer, L. P. Jain, and T. E. Boult, “Probability models for open set recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2317-2324, 2014.

[18] L. P. Jain, W. J. Scheirer, and T. E. Boult, “Multi-class open set recognition using probability of inclusion,” *European Conference on Computer Vision*, pp. 393–409, Springer, Cham, 2014.

[19] G. Dekkers, S. Lauwereins, B. Thoen, M. W. Adhana, H. Brouckxon, T. van Waterschoot, B. Vanrumste, M. Verhelst, and P. a. Karsmakers, “The SINS database for detection of daily activities in a home environment using an acoustic sensor network,” *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, November 2017.

[20] A. Mesaros, T. Heittola, and T. Virtanen. TUT database for acoustic scene classification and sound event detection. *In 24th European Signal Processing Conference*, pp.1128-1132, Budapest, Hungary, 2016.

[21] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, “Audio Set: An ontology and human-labeled dataset for audio events,” *Proceedings of the EEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp.776-780, 2017.

[22] J. Cramer, H. Wu, J. Salamon, and J. Pablo Bello “Look, listen and learn more: design choices for deep audio embeddings,” *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3852–3856, Brighton, UK, May 2019.

HODGEPODGE: SOUND EVENT DETECTION BASED ON ENSEMBLE OF SEMI-SUPERVISED LEARNING METHODS

Ziqiang Shi, Liu Liu, Huibin Lin, Rujie Liu

Fujitsu Research and Development Center
Beijing, China
shiziqiang@cn.fujitsu.com

Anyan Shi

ShuangFeng First
Beijing, China

ABSTRACT

In this paper, we present a method called HODGEPODGE[1] for large-scale detection of sound events using weakly labeled, synthetic, and unlabeled data in the Detection and Classification of Acoustic Scenes and Events (DCASE) 2019 challenge Task 4: Sound event detection in domestic environments. To perform this task, we adopted the convolutional recurrent neural networks (CRNN) as our backbone network. In order to deal with the small amount of tagged data and the large amounts of unlabeled in-domain data, we aim to focus primarily on how to apply semi-supervise learning methods efficiently to make full use of limited data. Three semi-supervised learning principles have been used in our system, including: 1) Consistency regularization applies data augmentation; 2) MixUp regularizer requiring that the predictions for a interpolation of two inputs is close to the interpolation of the prediction for each individual input; 3) MixUp regularization applies to interpolation between data augmentations. We also tried an ensemble of various models, which are trained by different semi-supervised learning principles. Our approach significantly improved the performance of the baseline, achieving a event-based f-measure of 42.0% compared to 25.8% of the baseline on the official evaluation dataset. Our submissions ranked third among 18 teams in the task 4.

Index Terms— DCASE 2019, convolutional recurrent neural networks, sound event detection, weakly-supervised learning, semi-supervised learning

1. INTRODUCTION

Sound carries a lot of information about our everyday environment and the physical events that take place there. We can easily perceive the sound scenes we are in (busy streets, offices, etc.) and identify individual sound events (cars, footsteps, etc.). The automatic detection of these sound events has many applications in real life. For example, it's very useful for intelligent devices, robots, etc., in the environment awareness. Also a sound event detection system can help to construct a complete monitoring system when the radar or video system may not work in some cases.

To contribute to the sound event detection task, the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge has been organized for four years since 2013 [1, 2, 3]. DCASE is a series of challenges aimed at developing sound classification and detection systems [1, 2, 3]. This year, the DCASE 2019 challenge comprises five tasks: acoustic scene classification, audio tagging with noisy labels and minimal supervision, sound event localization and detection, sound event detection in domestic environments, and urban sound tagging [3]. Among them, this paper describes a method for the task 4 of the DCASE 2019 challenge, large-scale detection of sound events in domestic environments using real data either weakly labeled or unlabeled, or synthetic data that is strongly labeled (with time stamps). The aim is to predict the presence or absence and the onset and offset times of sound events in domestic environments. This task is the follow-up to DCASE 2018 task 4, which aims at exploring the possibility to exploit a large amount of unbalanced and unlabeled training data together with a small weakly annotated training set to improve system performance. The difference is that there is an additional training set with strongly annotated synthetic data is provided in this year's task 4. Thus it can be seen that we are faced with three difficult problems: 1) there is no real strongly labeled and only too few weakly labeled data, 2) the synthetic data is obviously different from the real one, and how is the effect of synthetic data on the detection results? and 3) there is too much unlabeled data. Although this task is difficult , there have been a variety of methods proposed to solve this problem [4, 5, 6]. Furthermore, a baseline system that performs the task is provided in the DCASE 2019 challenge [7, 5].

Based on these previous studies, we propose to apply a convolutional recurrent neural network (CRNN), which is used as the backbone network in the baseline system for task 4 of DCASE 2019 [3]. In order to make full use of small amount of weakly labeled and synthetic data, the principles in interpolation consistency training (ICT) [8] and MixMatch [9] has been adopted in the 'Mean Teacher' [7, 5] framework. To avoid overfitting, consistency regularization on the provided unlabeled data is incorporated.

The rest of this paper is organized as follows: Section 2 introduces details of our proposed HODGEPODGE. The experiment settings and results are displayed and discussed in Section 3. We conclude this paper in Section 4.

2. PROPOSED METHOD

Herein, we present the method of our submissions for task 4 of DCASE 2019. In the following sections, we will describe the details of our approach, including feature extraction, network structure, how to use ICT and MixMatch in the context of 'Mean Teacher', and how to use unlabeled data.

[1]HODGEPODGE has two layers of meanings. The first layer is the variety of training data involved in the method, including weakly labeled, synthetic, and unlabeled data. The second layer refers to several semi-supervised principles involved in our method.

https://doi.org/10.33682/9kcj-bq06

2.1. Feature extraction

The dataset for task 4 is composed of 10 sec audio clips recorded in domestic environment or synthesized to simulate a domestic environment. No preprocessing step was applied in the presented frameworks. The acoustic features for the 44.1kHz original data used in this system consist of 128-dimensional log mel-band energy extracted in Hanning windows of size 2048 with 431 points overlap. Thus the maximum number of frames is 1024. In order to prevent the system from overfitting on the small amount of development data, we added random white noise (before log operation) to the melspectrogram in each mini-batch during training. The input to the network is fixed to be 10-second audio clip. If the input audio is less than 10 seconds, it is padded to 10 seconds; otherwise it is truncated to 10 seconds.

2.2. Neural network architecture

Figure 1 presents the CRNN network architecture employed in our HODGEPODGE. The audio signal is first converted to [128×1024] log-melspectrogram to form the input to the network. The first half of the network consists of the seven convolutional layers, where we use gated linear units (GLUs) instead of commonly rectified linear units (RELUs) or leaky ReLUs as nonlinear activations.

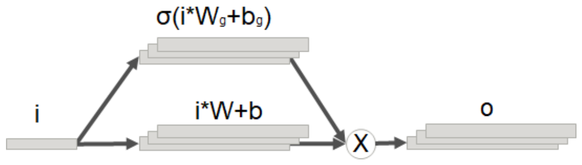


Figure 2: Architecture of a GLU.

Figure 2 shows the structure of a GLU :

$$o = (i * W + b) \otimes \sigma(i * W_g + b_g),$$

where i and o are the input and output, W, b, W_g, and b_g are learned parameters, σ is the sigmoid function and $\otimes$ is the element-wise product between vectors or matrices. Similar to LSTMs, GLUs play the role of controlling the information passed on in the hierarchy. This special gating mechanism allows us to effectively capture long-range context dependencies by deepening layers without encountering the problem of vanishing gradient.

For the seven gated convolutional layers, the kernel sizes are 3, the paddings are 1, the strides are 1, and the number of filters are [16, 32, 64, 128, 128, 128, 128] respectively, and the poolings are [(2, 2), (2, 2), (1, 2), (1, 2), (1, 2), (1, 2), (1, 2)] respectively. Pooling along the time axis is used in training with the clip-level and frame-level labels.

The gated convolutional blocks are followed by two bidirectional gated recurrent units (GRU) layers containing 64 units in the forward and backward path, their output is concatenated and passed to the attention and classification layer which are described below.

As depicted in Figure 1, the output of the bidirectional GRU layers is fed into both a frame-level classification block and an attention block respectively. The frame-level classification block uses a sigmoid activation function to predict the probability of each occurring class at each frame. Thus bidirectional GRUs followed by a dense layer with sigmoid activation to compute posterior probabilities of the different sounds classes. In that case there are two outputs in this CRNN. The output from bidirectional GRUs followed by dense layers with sigmoid activation is considered as sound event detection result. This output can be used to predict event activity probabilities. The other output is the weighted average of the element-wise multiplication of the attention, considering as audio tagging result. Thus the final prediction for the weak label of each class is determined by the weighted average of the element-wise multiplication of the attention and classification block output of each class c.

2.3. Semi-supervised learning

Inspired by the DCASE 2018 task 4 winner solution [5] and the baseline system [10], in which it uses the 'Mean Teacher' model [7], we also used 'Mean Teacher' as the main framework of our system.. 'Mean Teacher' is a combination of two models: the student model and the teacher model. At each training step, the student model is trained on synthetic and weakly labeled data with binary cross entropy classification cost. While the teacher model is an exponential moving average of the student models. The student model is the final model and the teacher model is designed to help the student model by a consistency mean-squared error cost for frame-level and clip-level predictions of unlabeled audio clips. That means good student should output the same class distributions as the teacher for the same unlabeled example even after it has been perturbed by Gaussian noise augmentation. The goal of 'Mean Teacher' is to minimize:

$$L = L_w + L_s + w(t)L_{cw} + w(t)L_{cs}$$

where L_w and L_s are the usual cross-entropy classification loss on weakly labeled data with only weak labels and synthetic data with only strong labels respectively, L_{cw} and L_{cs} are the teacher-student consistence regularization loss on unlabeled data with predicted weak and strong labels respectively, and $w(t)$ is the balance of classification loss and the consistency loss. Generally the $w(t)$ changes over time to make the consistency loss initially accounts for a very small proportion, and then the ratio slowly becomes higher. Since in the beginning, neither the student model nor the teacher model were accurate on predictions, and the consistency loss did not make much sense. $w(t)$ has a maximum upper bound, that is, the proportion of consistent loss does not tend to be extremely large. With different maximum upper bound of consistence weight $w(t)$, the trained model has different performances. In the next section, we ensemble the models trained under different maximum consistence weights to achieve better results.

HODGEPODGE did not change the overall framework of the baseline. It only attempts to combine several of the latest semi-supervised learning methods under this framework.

The first attempt is the interpolation consistency training (ICT) principle [8]. ICT teaches the student network in a semi-supervised manner. To this end, ICT uses a 'Mean Teacher' $f_{\theta'}$. During training, the student parameters θ are updated to encourage consistent predictions

$$f_\theta(\text{Mix}_\lambda(u_j, u_k)) \approx \text{Mix}_\lambda(f_{\theta'}(u_j), f_{\theta'}(u_k)),$$

and correct predictions for labeled examples, where

$$\text{Mix}_\lambda(a, b) = \lambda a + (1 - \lambda)b$$

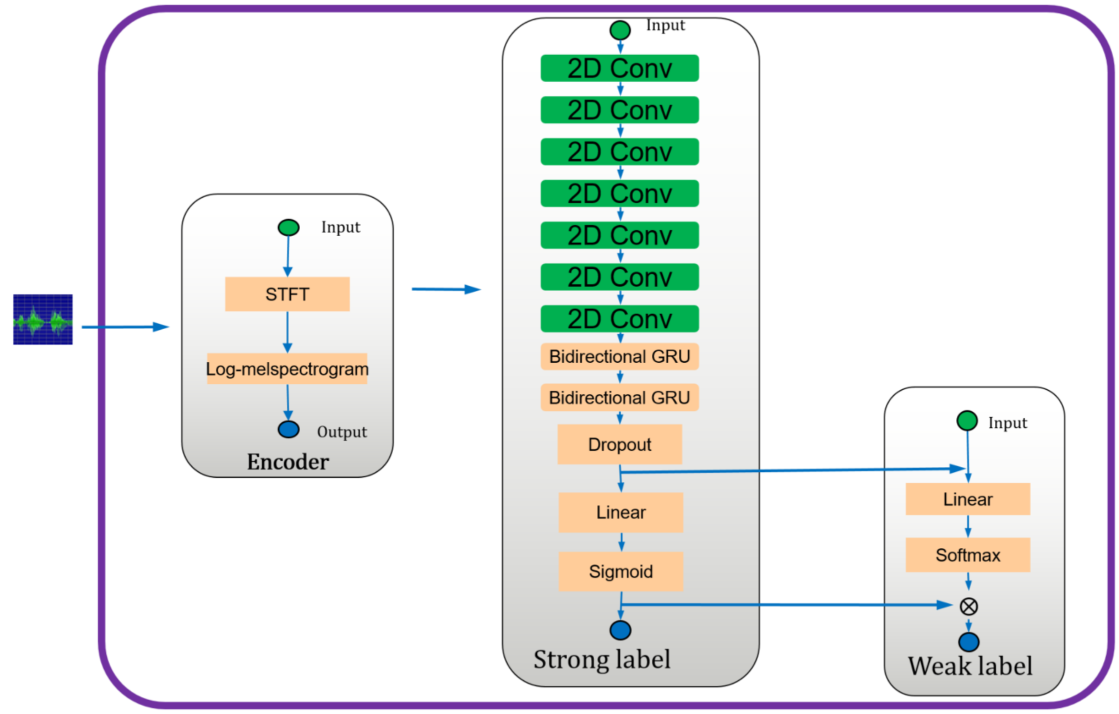


Figure 1: Architecture of the CRNN in HODGEPODGE.

is called the interpolation or MixUp [11] of two log-melspectrograms u_j and u_k, and on each batch we sample a random λ from Beta(α; α) (e.g. $\alpha = 1.0$ in all our settings). In sum, the population version of our ICT term can be written a. In our system, we perform interpolation of sample pair and their corresponding labels (or pseudo labels predicted by the CRNNs) in both the supervised loss on labeled examples and the consistency loss on unsupervised examples. In each batch, the weakly labeled data, synthetic data, and unlabeled data are shuffled separately to form a new batch. There are 24 audio log-melspectrograms in each batch, of which 6 are weakly labeled, 6 are synthetic audio data, and the remaining 12 are unlabeled. Then we use the ICT principle to generate new augmented data and labels with the corresponding clips in the original and new batches. It should be noted that λ is different for each batch. Thus the loss

$$L_{ict} = L_{w,ict} + L_{s,ict} + w(t)L_{cw,ict} + w(t)L_{cs,ict}$$

where $L_{w,ict}$ and $L_{s,ict}$ are the classification loss on weakly labeled data with only weak labels and synthetic data with only strong labels using ICT respectively, $L_{cw,ict}$ and $L_{cs,ict}$ are the teacher-student consistence regularization loss on ICT applied on unlabeled data with predicted weak and strong labels respectively.

The second try draws on some of the ideas in MixMatch [9], but not exactly the same. MixMatch introduces a single loss that unifies entropy minimization, consistency regularization, and generic regularization approaches to semi-supervised learning. Unfortunately MixMatch can only be used for one-hot labels, not suitable for task 4, where may be several events in a single audio clip. So we didn't use MixMatch in its original form. In each batch, $K(> 1)$ different augmentations are generated, then the original MixMatch does mixup on all data, regardless of whether the data is weakly labeled, synthetic or unlabeled. In our experiments, we found that the effect is not good, so we fine-tuned the MixMatch to do MixUp only between the augmentations of the same data type. The loss function is similar to the loss in the ICT case.

2.4. Model ensemble and submission

To further improve the performance of the system, we use some ensemble methods to fuse different models. The main differences of the single models have two dimensions, one is the difference of the semi-supervised learning method, and the other is the difference of the maximum value of the consistency loss weight. For this challenge, we submitted 4 prediction results with different model ensemble:

- HODGEPODGE 1: Ensemble model is conducted by averaging the outputs of 9 different models with different maximum consistency coefficients in 'Mean Teacher' principle. The F-score on validation data was 0.367. (Corresponding to Shi_FRDC_task4_1 in official submissions)
- HODGEPODGE 2: Ensemble model is conducted by averaging the outputs of 9 different models with different maximum consistency coefficients in ICT principle. The F-score on validation data was 0.425. (Corresponding to Shi_FRDC_task4_2 in official submissions)
- HODGEPODGE 3: Ensemble model is conducted by averaging the outputs of 6 different models with different maximum consistency coefficients in MixMatch principle. The F-score on validation data was 0.389. (Corresponding to

Shi_FRDC_task4_3 in official submissions)

- HODGEPODGE 4: Ensemble model is conducted by averaging the outputs of all the 24 models in Submission 1, 2, and 3. The F-score on validation data was 0.417. (Corresponding to Shi_FRDC_task4_4 in official submissions)

3. EXPERIMENTS AND RESULTS

3.1. Dataset

Sound event detection in domestic environments [11] is a task to detect the onset and offset time steps of sound events in domestic environments. The datasets are from AudioSet [12], FSD [13] and SINS dataset [14]. The aim of this task is to investigate whether real but weakly annotated data or synthetic data is sufficient for designing sound event detection systems. There are a total of 1578 real audio clips with weak labels, 2045 synthetic audio clips with strong labels, and 14412 unlabeled in domain audio clips in the development set, while the evaluation set contains 1168 audio clips. Audio recordings are 10 seconds in duration and consist of polyphonic sound events from 10 sound classes.

3.2. Evaluation Metric

The evaluation metric for this task is based on the event-based F-score [15]. The predicted events are compared to a list of reference events by comparing the onset and offset of the predicted event to the overlapping reference event. If the onset of the predicted event is within 200 ms collar of the onset of the reference event and its offset is within 200 ms or 20% of the event length collar around the reference offset, then the predicted event is considered to be correctly detected, referred to as true positive. If a reference event has no matching predicted event, then it is considered a false negative. If the predicted event does not match any of the reference events, it is considered a false positive. In addition, if the system partially predicts an event without accurately detecting its onset and offset, it will be penalized twice as a false positive and a false negative. The following equation shows the calculation of the F-score for each class.

$$F_c = \frac{2TP_c}{2TP_c + FP_c + FN_c},$$

where F_c, TP_c, FP_c, FN_c are the F-score, true positives, false positives, false negatives of the class c respectively. The final evaluation metric is the average of the F-score for all the classes.

3.3. Results

First we did some experiments to determine the best size of the median window. The median window is used in the post-processing of posterior probabilities to results in the final events with onset and offset. Table 1 shows the performance of HODGEPODGE systems on validation data set under different median window size. Coincidentally, all methods achieve the best performance when the window size is 9.

Table 2 shows the final macro-averaged event-based evaluation results on the test set compared to the baseline system. In fact, HODGEPODGE 1 is the ensemble of baselines, the only difference is that we use a deeper network, as well as higher sampling rate and larger features. It can be seen that both ICT and MixMatch principles can improve performance, especially ICT, which performs best in all HODGEPODGE systems.

Table 1: The performance of HODGEPODGE systems on validation data set under different median window size.

Median window size	5	7	9	11	13
HODGEPODGE 1	35.7%	36.4%	36.7%	36.5%	36.1%
HODGEPODGE 2	41.4%	42.1%	42.5%	42.2%	42.1%
HODGEPODGE 3	38.1%	38.7%	38.9%	38.3%	37.9%
HODGEPODGE 4	40.8%	41.5%	41.7%	41.3%	40.9%

Table 2: The performance of our approach compared to the baseline system.

Method	Evaluation	Validation
HODGEPODGE 1	37.0%	36.7%
HODGEPODGE 2	42.0%	42.5%
HODGEPODGE 3	40.9%	38.9%
HODGEPODGE 4	41.5%	41.7%
Baseline	25.8%	23.7%

4. CONCLUSIONS

In this paper, we proposed a method called HODGEPODGE for sound event detection using only weakly labeled, synthetic and unlabeled data. Our approach is based on CRNNs, whereby we introduce several latest semi-supervised learning methods, such as interpolation consistence training and MixMatch into the 'Mean Teacher' framework to leverage the information in audio data that are not accurately labeled. The final F-score of our system on the evaluation set is 42.0%, which is significantly higher than the score of the baseline system which is 25.8%.

5. ACKNOWLEDGEMENT

Many thanks to the anonymous reviewers for their comments, which make our description better.

6. REFERENCES

[1] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the dcase 2016 challenge," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, vol. 26, no. 2, pp. 379–393, 2018.

[2] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, "Dcase 2017 challenge setup: Tasks, datasets and baseline system," in *DCASE 2017-Workshop on Detection and Classification of Acoustic Scenes and Events*, 2017.

[3] http://dcase.community/challenge2019/.

[4] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. P. Shah, "Large-scale weakly labeled semi-supervised sound event detection in domestic environments," *arXiv preprint arXiv:1807.10501*, 2018.

[5] L. JiaKai, "Mean teacher convolution system for dcase 2018 task 4," DCASE2018 Challenge, Tech. Rep., September 2018.

[6] Q. Kong, T. Iqbal, Y. Xu, W. Wang, and M. D. Plumbley, "Dcase 2018 challenge baseline with convolutional neural networks," *arXiv preprint arXiv:1808.00773*, 2018.

[7] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in neural information processing systems*, 2017, pp. 1195–1204.

[8] V. Verma, A. Lamb, J. Kannala, Y. Bengio, and D. Lopez-Paz, "Interpolation consistency training for semi-supervised learning," *arXiv preprint arXiv:1903.03825*, 2019.

[9] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," *arXiv preprint arXiv:1905.02249*, 2019.

[10] N. Turpault, R. Serizel, A. P. Shah, and J. Salamon, "Sound event detection in domestic environments with weakly labeled data and soundscape synthesis," 2019.

[11] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[12] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[13] E. Fonseca, J. Pons Puig, X. Favory, F. Font Corbera, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound datasets: a platform for the creation of open audio datasets," in *Hu X, Cunningham SJ, Turnbull D, Duan Z, editors. Proceedings of the 18th ISMIR Conference; 2017 oct 23-27; Suzhou, China.[Canada]: International Society for Music Information Retrieval; 2017. p. 486-93.* International Society for Music Information Retrieval (ISMIR), 2017.

[14] G. Dekkers, S. Lauwereins, B. Thoen, M. W. Adhana, H. Brouckxon, T. van Waterschoot, B. Vanrumste, M. Verhelst, and P. Karsmakers, "The sins database for detection of daily activities in a home environment using an acoustic sensor network," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017), Munich, Germany*, 2017, pp. 32–36.

[15] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.

DEEP MULTI-VIEW FEATURES FROM RAW AUDIO FOR ACOUSTIC SCENE CLASSIFICATION

Arshdeep Singh, Padmanabhan Rajan, Arnav Bhavsar

MANAS LAB, School of Computing and Electrical Engineering,
Indian Institute of Technology, (IIT), Mandi, India
d16006@students.iitmandi.ac.in, [padman,arnav]@iitmandi.ac.in

ABSTRACT

In this paper, we propose a feature representation framework which captures features constituting different levels of abstraction for audio scene classification. A pre-trained deep convolution neural network, SoundNet, is used to extract the features from various intermediate layers corresponding to an audio file. We consider that the features obtained from various intermediate layers provide the different types of abstraction and exhibits complementary information. Thus, combining the intermediate features of various layers can improve the classification performance to discriminate audio scenes. To obtain the representations, we ignore redundant filters in the intermediate layers using analysis of variance based redundancy removal framework. This reduces dimensionality and computational complexity. Next, shift-invariant fixed-length compressed representations across layers are obtained by aggregating the responses of the important filters only. The obtained compressed representations are stacked altogether to obtain a supervector. Finally, we employ the classification using multi-layer perceptron and support vector machine models. We comprehensively perform the validation of the above assumption on two public datasets; Making Sense of Sounds and open set acoustic scene classification DCASE 2019.

***Index Terms*—** Acoustic scene classification, Deep neural network, SoundNet.

1. INTRODUCTION

Acoustic scene classification (ASC) aims to utilize the audio information in everyday soundscapes to recognise the underlying physical environment (commonly referred to as scene). Traditionally, most of the work in audio scene classification, inspired from the closely related fields such as speech recognition and music analysis, employed hand-crafted time-frequency based representations such as spectrogram, log-mel energy, mel-frequency cepstral coefficients, constant-Q-transform etc. However, the hand-crafted features are often not able to adapt to acoustic scenes data owing to the complexity which arises mostly from many independent unknown sources which produces unstructured sounds. Moreover, the audio information in the scene spans whole audio spectrum . To circumvent this, feature learning based approaches are being applied to learn relevant information directly from time-frequency representations. For examples, the work in [1] applied matrix factorization based representations. [2] used i-vector and deep convolution neural network (CNN) based features. The study [3] used a dictionary learning framework which captures the rare and most frequently occurring sound events. Apart from this, ensemble based methods which combines multiple channels and models are also being reported [4].

A few studies explored the feature representations from raw audio directly. For example, [5] demonstrated that deep CNNs trained directly on very long raw acoustic sound waveforms can outperform than CNNs with similar architecture on handcrafted features. The study [6] proposed a pre-trained deep convolutional neural network, SoundNet, that accepts raw audio as input. The work [7] performed a layer-wise analysis on SoundNet layers and proposed an ensemble framework in decision space.

In this paper, we propose a representations framework for ASC by utilizing the intermediate layer representations obtained using SoundNet, from raw audio. Our underlying assumption is that the intermediate representations of different layers in SoundNet, correspond to different details of an audio. To illustrate this, we show the frequency response for some of the learned filters in the first and second convolution layers of SoundNet in Figure 1. It can be observed that the filters (a) and (b) in the first convolution layer have different bandpass frequency characteristics and hence, produce different details of an audio. In the subsequent layers, the output of the filters from the previous layer is being operated with a different set of filters, which also posses different bandpass information. As shown in the Figure 1, the learned filters (c)-(e) in the second convolution layer have different bandpass characteristic and operate on the output of the filter (a) (learned in the first convolution layer). Similarly, the filters (f)-(h) in the second convolution layer operate on the output of the filter (b) which is being learned in the first convolution layer. Therefore, an audio signal is operated upon by filters having different frequency responses along the layers. Moreover, the non-linear operations such as batch normalization and ReLU transformation, project the data into different subspaces [8]. Therefore, the representations obtained from different layers can be considered as exhibiting different characteristics of an audio.

Henceforth, such intermediate representations are further transformed into compressed features as explained in subsection 2.2 and concatenated altogether to build a supervector, which captures the multiple details of an audio. Empirically, we analyse the validity of the proposed feature representation approach for two publicly available datasets.

The rest of the paper is organized as follows. Section 2 gives the main idea of our proposed method in which we describe the feature extraction, the compressed feature representation and the classification methods. Section 3 shows the experimental setup and findings. In Section 4, we conclude this paper.

https://doi.org/10.33682/05gk-pd08

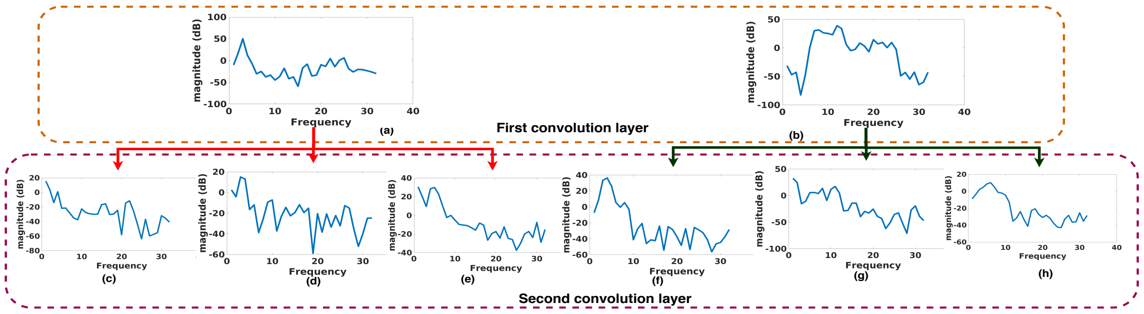


Figure 1: Single-sided magnitude of frequency spectrum for some of the learned filters in SoundNet. (a) and (b) shows the frequency response for fifth and sixteenth filters respectively in the first convolution layer. (c)-(h) shows frequency response of filters in second convolution layer. The filters (c)-(e) (here, only three are shown) operate on the output produced by filter (a). Similarly, (f)-(h) shows frequency response for filters which operate on response of (b) filter. Here, the frequency spectrum is computed using 64-point DFT and the magnitude (dB) is $20 \log |X(f)|$. $|X(f)|$ is the magnitude of frequency spectrum.

2. PROPOSED METHODOLOGY

2.1. A brief on pre-trained 1-D CNN

SoundNet is a 1-D CNN, trained on large-scale weakly labeled video datasets, ultimately performing transfer learning from video to audio. The architecture has 8-layers namely convolution, pooling layers and operates on the raw audio directly. Each convolution layer (denoted as C) output is computed by convolution operation followed by batch normalization (denoted as p-C) and non-linear activation operations (ReLU).

An audio signal x, of duration t seconds, and sampled at f_s frequency, can be represented into a 2-D representations $\in \mathbb{R}^{N \times s}$ using any intermediate layer of SoundNet. Here, N and s represent the number of feature maps and their size respectively in a given layer.

2.2. Compressed feature representation and classification

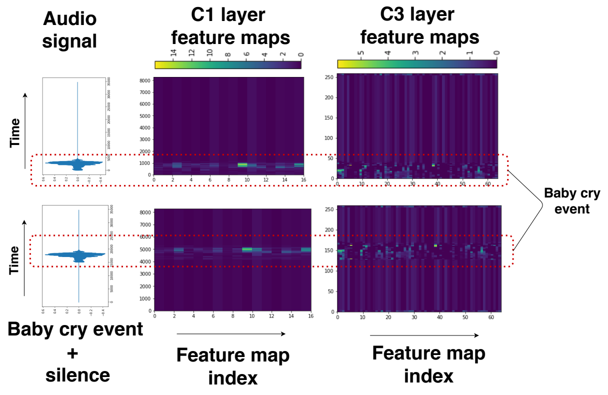


Figure 2: 2-D intermediate layer representations for first (C1) and third (C3) convolutional layer in SoundNet corresponding to 5 seconds audio. Here, the audio is a baby cry event and a silence.

The intermediate 2-D feature representations obtained from SoundNet have very high dimensionality of the order of approx. 16M and 27k for an audio of length 5 seconds sampled at 44.1kHz for first (C1) and third (C3) convolution layer respectively. In addition, the size of representations depend on the input audio length and the representations are not time-invariant. Figure 2 demonstrates the time-variance of the intermediate layers representations as the input shifts in time.

We reduce the dimensionality in two ways: first, since all the learned filters in SoundNet do not provide discriminatory response [9], some of the filter responses can be ignored. We employ analysis of variance method based pruning procedure as proposed in [10] to identify the filters, which generates discriminating response across scene classes. This is done for each of layers independently. Ignoring the non-discriminating responses result into a reduced dimension 2-D representations $\in \mathbb{R}^{N' \times s}$, where $N' \leq N$. Second, to compute the time-invariant representations and compress the intermediate representation further, global sum pooling is applied across the response of filters . This results into a fixed-length representations $\in \mathbb{R}^{N'}$ of an audio for a particular intermediate layer. Henceforth, we call these fixed-length representations as **compressed features**.

We utilize the compressed features from various layers to build a global super-vector ξ, representing different details of an audio by concatenating the compressed features from various layers. A variable length audio of very high dimensionality can now be represented using ξ-features. Since these features represent different characteristics of an audio, therefore we call them **"multi-view features"**. Finally, we employ multilayer perceptron (MLP) model and support vector machine (SVM) as a classifier. The flow diagram of the overall proposed framework is shown in Figure 3.

3. PERFORMANCE EVALUATION

3.1. Dataset and Experimental setup

We use the following audio scene classification (ASC) datasets for evaluation: first, the Making Sense of Sounds (MSOS) challenge dataset [11], comprising of a development dataset consists of 1500 audio files divided into the five categories, each containing 300 files. The number of different sound types within each category is not balanced. The evaluation dataset consists of 500 audio files, 100 files per category. F-measure and accuracy metrics are used to measure the performance.

Second, the TAU Urban Scenes openset development and

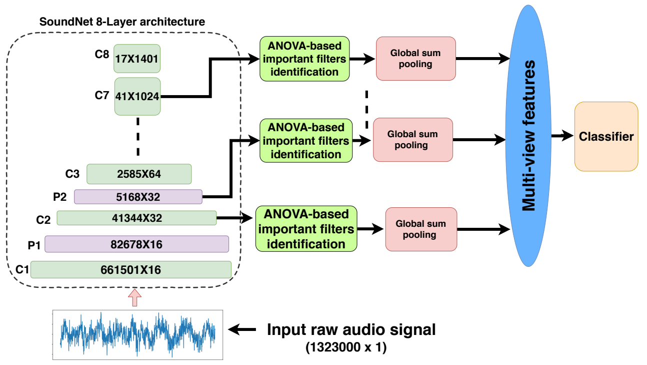


Figure 3: Overall proposed framework. The size and number of feature maps for each layer are shown corresponding to input of 30 seconds length sampled at 44.1kHz. C1, P1, C2 etc. represents first convolution, first pooling, second convolution and so on layers.

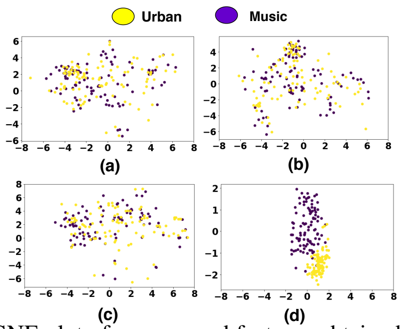


Figure 4: t-SNE plot of compressed features obtained from (a) P2, (b) C3 (c) C7 and (d) the proposed multi-view features from various layers of two classes namely, urban and music from MSOS Evaluation dataset.

leaderboard dataset [12] (DCASE), comprising of ten known target classes and one unknown class is used. The "unknown" class has additional data of several unknown acoustic scenes. The training and testing development data is divided as described in the task protocol for all 11 classes. We handle the out-of-set classification as follows. A given test recording is being classified as a particular scene class if the class-specific probability is greater than threshold (τ), $\tau \in [0, 1)$. Otherwise, if all classes have lesser than τ, the sample is assigned as an unknown class label. For DCASE dataset, the weighted average accuracy (α_w), the accuracy of known classes (α_k) and unknown class (α_u) is used as metric as given in the Equation 1. We report the performance for leaderboard dataset using the online portal.

$$\alpha_w = 0.5 * \alpha_k + 0.5 * \alpha_u \qquad (1)$$

The compressed features are computed from C2 to C7 layer (a total of 4256 filter responses) including the p-C layers. We obtain a total of 1307 non-redundant filter responses, as explained in section 2.2. This leads to give multi-view feature $\xi \in \mathbb{R}^{1307}$ obtained from 12 intermediate layers after performing global sum pooling on each filter response.

The classification model parameters such as number of hidden layers, neurons in MLP and hyper-parameters of non-linear SVM are selected empirically. The MLP is trained with Adam optimizer

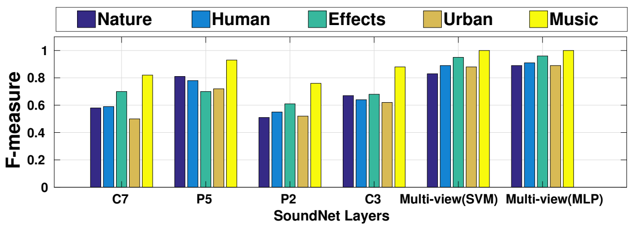
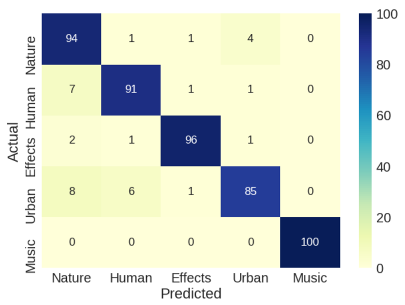


Figure 5: F-measure obtained with compressed features from C7, P5, P2, C3 layers using SVM and multi-view features using SVM and MLP for MSOS dataset.


Actual
Predicted
Nature
Human
Effects
Urban
Music

	Nature	Human	Effects	Urban	Music
Nature	94	1	1	4	0
Human	7	91	1	1	0
Effects	2	1	96	1	0
Urban	8	6	1	85	0
Music	0	0	0	0	100

Figure 6: Confusion matrix using multi-view features with MLP for MSOS dataset.

and cross-entropy categorical loss for 100 epochs. Empirically, we find that single hidden layer having 30 neurons with hyperbolic tangent activation function suits well for our classification task.

3.2. Results and Analysis

3.2.1. Dataset (A): Making sense of Sounds

Figure 4 shows the t-SNE plot for compressed features obtained from various intermediate layers and ξ-features for two sound classes. It can be observed that the ξ-feature space shows lesser inter-class overlap as compared to the feature space generated from the individual intermediate layers.

In addition to give more separability, the ξ-feature space also utilizes complementary information given by various layers. This can be observed from Figure 5, which shows the F-measure, computed using the compressed features obtained from various intermediate layers and ξ-features. The F-measure for a given class varies across layers, for example, "Nature" has larger F-measure for C3 layer as compared to C7 and P2 layers. This is valid for "Human" as well. However, C7 layer has larger F-measure for "Effects" as compared to P2 and C3 layers. This empirically shows that the feature space generated from various layers constitute complementary information. Utilizing the compressed features from various layers to build multi-view features improve the F-measure of all scene class significantly. The accuracy for compressed features from C7, P5, P2, C3 and ξ-features using SVM is 64.2%, 79%, 59.6%, 70% and 91% respectively. Using MLP, the accuracy obtained with ξ-features is 93.2%. Figure 6 gives the confusion matrix obtained using ξ-features with MLP as a classifier. It can be observed that "Urban" is most frequently confused as "Nature" and "Human".

Comparison with existing methods: Figure 7 shows the comparison of class-wise accuracy for baseline, state-of-the-art [11] and

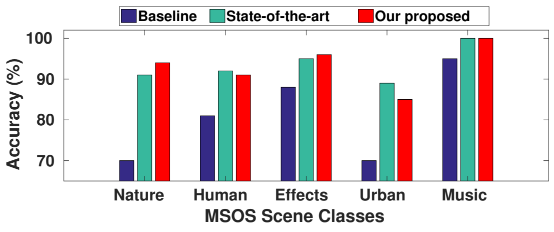


Figure 7: Comparison of class-wise accuracy with the existing methods for MSOS dataset.

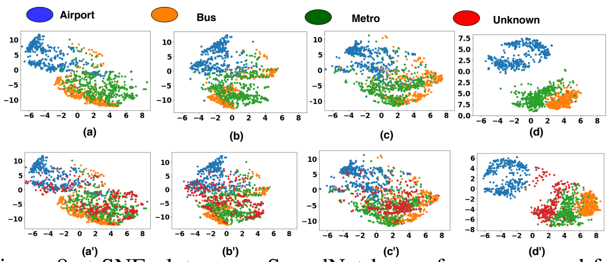


Figure 8: t-SNE plot across SoundNet layers for compressed features from (a) P2 (b) C3 (c) C7 and (d) the proposed multi-view features for three scene class, namely airport, bus and metro. Here, (a'), (b'), (c') and (d') shows the same when unknown class is also being considered.

our proposed approach with MLP. For all classes, our proposed approach provides significant improvement as compared to the baseline. On the other hand, the performance is equivalent to state-of-the-art with an improvement for "Nature" and Effects by 3% and 1% respectively, however, the performance degrades for "Human" and "Urban" by 1% and 4% respectively. The accuracy across classes for baseline, the state-of-the-art and our proposed approach is 81%, 93.4% and 93.2% respectively.

3.2.2. *Dataset (B): TAU Urban Acoustic Scenes 2019 Openset*

Figure 8 shows the t-SNE plot obtained using compressed features from various layers and ξ-features for three known classes, airport, bus, metro and one unknown class. It is notable that the proposed multi-view features are able to provide better separation among known classes as compared to the classes considering both known and unknown.

Figure 9 gives the performance obtained using MLP and SVM classifier as τ varies from 0 to 1. For τ close to 0, the classifiers are able to classify the known classes significantly well. However, the unknown class has poor accuracy. As τ increases, the unknown class accuracy increases, however, when τ is very close to 1, the known class accuracy is poor. It can be observed that there is a trade-off in accuracy of the known and unknown class with threshold. Our proposed framework improves α_w significantly by 12% to 31% (choosing $0.5 < \tau < 1$) as compared to that of baseline [12] which has α_w equals to 48.7%.

For leaderboard dataset, α_w is computed through the public leaderboard online portal. Figure 10 shows α_w as a function of τ with MLP and SVM classifiers. In case of MLP, as the τ increases towards 1, the α_w also increases and approaches to the baseline performance which is around 44% (private leaderboard). Utilizing SVM, the α_w remains constant at 17.5% beyond 0.5 threshold. This may be due to the over-fitting of the SVM classifier towards the

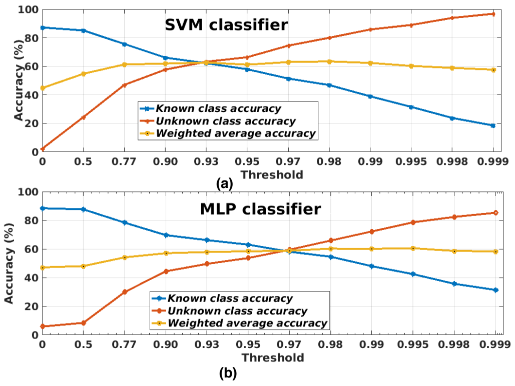


Figure 9: Known, unknown and weighted average accuracy as a function of threshold (τ) using (a) SVM and (b) MLP as a classifier for DCASE development dataset.

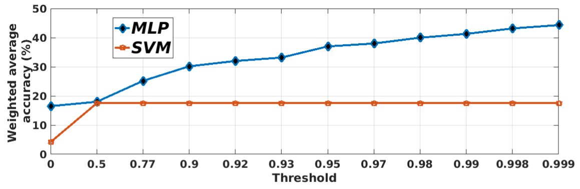


Figure 10: Weighted average accuracy (α_w) as a function of threshold using MLP and SVM for DCASE leaderboard dataset.

training dataset. The predicted scene labels obtained using different threshold for leaderboard dataset can be found on this link.

3.2.3. *Discussion*

The proposed approach is performing well for MSOS dataset. However, the overall performance for DCASE dataset especially the leaderboard dataset (recorded at different locations and time instants), is not that overwhelming. We speculate that this might be caused because the resulting latent space obtained from a pre-trained model is not able to discriminate each of the classes, especially the "unknown" class. SoundNet is trained using transfer learning from 2 million Flicker videos [6]. The MSOS dataset contains the audio files collected from Freesound and the other online sources [11]. This may lead to give similar distributions between the learned parameters of SoundNet and the MSOS dataset, hence, the model shows good representation strength. However, the DCASE dataset is recorded at various locations in an uncontrolled environment and with more confusing classes. Hence, DCASE dataset shows more domain mismatch to the pre-trained SoundNet. In addition, the complexity of DCASE dataset can not be ignored. Therefore, we experiment to adapt the existing model with new datasets such as DCASE and expecting to perform better than the approach proposed in this paper in future.

4. CONCLUSION

We propose a feature representation framework using various intermediate levels of the pre-trained deep CNN SoundNet, for acoustic scene classification. The combined features from the intermediate layers are able to provide better discrimination as compared to the features from each of the individual layers.

5. REFERENCES

[1] V. Bisot, R. Serizel, S. Essid, and G. Richard, "Supervised nonnegative matrix factorization for acoustic scene classification," DCASE2016 Challenge, Tech. Rep., September 2016.

[2] H. Eghbal-Zadeh, B. Lehner, M. Dorfer, and G. Widmer, "CP-JKU submissions for dcase-2016: a hybrid approach using binaural i-vectors and deep convolutional neural networks," *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2016.

[3] A. Singh, A. Thakur, and P. Rajan, "APE: Archetypal-prototypal embeddings for audio classification," in *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2018, pp. 1–6.

[4] S. Mun, S. Park, D. Han, and H. Ko, "Generative adversarial network based acoustic scene training set augmentation and selection using SVM hyper-plane," DCASE2017 Challenge, Tech. Rep., September 2017.

[5] T. Purohit and A. Agarwal, "Acoustic scene classification using deep CNN on raw-waveform," DCASE2018 Challenge, Tech. Rep., September 2018.

[6] Y. Aytar, C. Vondrick, and A. Torralba, "Soundnet: Learning sound representations from unlabeled video," in *Advances in neural information processing systems*, 2016, pp. 892–900.

[7] A. Singh, A. Thakur, P. Rajan, and A. Bhavsar, "A layer-wise score level ensemble framework for acoustic scene classification," in *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, 2018, pp. 837–841.

[8] S. Yu and J. C. Principe, "Understanding autoencoders with information theoretic concepts," *Neural Networks*, vol. 117, pp. 104–123, 2019.

[9] A. RoyChowdhury, P. Sharma, E. Learned-Miller, and A. Roy, "Reducing duplicate filters in deep neural networks," in *NIPS workshop on Deep Learning: Bridging Theory and Practice*, vol. 1, 2017.

[10] A. Singh, P. Rajan, and A. Bhavsar, "Deep hidden analysis: A statistical framework to prune feature maps," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 820–824.

[11] C. Kroos, O. Bones, Y. Cao, L. Harris, P. J. Jackson, W. J. Davies, W. Wang, T. J. Cox, and M. D. Plumbley, "Generalisation in environmental sound classification: the making sense of sounds data set and challenge," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 8082–8086.

[12] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 9–13. [Online]. Available: https://arxiv.org/abs/1807.09840

AUDIO TAGGING USING A LINEAR NOISE MODELLING LAYER

Shubhr Singh, Arjun Pankajakshan and Emmanouil Benetos
{s.singh@se17. , a.pankajakshan@ , emmanouil.benetos@}qmul.ac.uk

School of Electronic Engineering and Computer Science, Queen Mary University of London, UK

ABSTRACT

Label noise refers to the presence of inaccurate target labels in a dataset. It is an impediment to the performance of a deep neural network (DNN) as the network tends to overfit to the label noise, hence it becomes imperative to devise a generic methodology to counter the effects of label noise. FSDnoisy18k is an audio dataset collected with the aim of encouraging research on label noise for sound event classification. The dataset contains ∼42.5 hours of audio recordings divided across 20 classes, with a small amount of manually verified labels and a large amount of noisy data. Using this dataset, our work intends to explore the potential of modelling the label noise distribution by adding a linear layer on top of a baseline network. The accuracy of the approach is compared to an alternative approach of adopting a noise robust loss function. Results show that modelling the noise distribution improves the accuracy of the baseline network in a similar capacity to the soft bootstrapping loss.

Index Terms— Audio tagging, noisy labels, noise adaptation layer, noise robust loss function.

1. INTRODUCTION

Audio tagging refers to the classification task which involves predicting the presence of one or more acoustic events in a particular audio recording. Humans are able to perform this task effortlessly, however modelling this cognitive process through computational methods is non-trivial [1] and is an active research area which has received increased attention in recent years. The majority of current approaches to the problem involve supervised training of a deep neural network (DNN) on the labels associated with each audio recording [2],[3]. The basic assumption that comes along with these approaches is that the label provided with the audio recording is correct, i.e. the presence of the acoustic event associated with the label corresponding to the audio recording has been manually verified. This assumption does not always hold true as manual verification of data labels is a costly affair, effectively limiting the size of the data sets.

As described in [4], a large amount of audio data accumulation comes at the cost of imprecise labels, especially in cases such where labels have been inferred based on user provided metadata, i.e. tags. As the labels are noisy, there is a high probability of misleading information, which in turn subverts the training of a DNN. Recent studies [5], [6] have shown that the generalization capability of a DNN reduces on datasets with noisy labels, i.e. the model overfits the training data.

In this paper, we explore the FSDnoisy18k dataset [4] for sound event classification, which contains a small subset (10%) of accurately labelled data and a large subset (90%) consisting of data samples with noisy labels. We use a linear noise distribution modelling layer approach and compare its performance on the test set with that of the baseline model and also with the soft bootstrapping loss function approach [7]. Both approaches have been implemented for the problem of noisy labels in computer vision [6], [7], however to the authors' knowledge, experiments on audio data with a linear noise modelling layer is yet to be explored. We adopt the MobilenetV2 architecture as our baseline model without any pretrained weights and train the network on the training set of FSDnoisy18k. The model weights with the least categorical cross entropy loss (CCE) on the validation set are selected and a dense layer with softmax activation is placed on top of the network and re-trained on the training set. The purpose of the re-training is to learn the weight parameters of the noise modelling layer. The noise modelling layer is removed during prediction on the test set. We observed that the noise modelling layer approach improves the accuracy of the baseline network on test set by approximately 2% and the soft bootstrapping loss function approach improves the accuracy score of the baseline network by approximately 1.5%.

The paper is organized as follows. Section 2 introduces related work, Section 3 discusses the characteristics of the dataset used in this paper, Section 4 discusses the type of label noise present in the dataset, Section 5 details the MobilenetV2 architecture, Section 6 discusses the two approaches to the problem, Section 7 covers the experimental setup and the evaluation metrics adopted for the paper, Section 8 discusses the results, and Section 9 concludes the paper and discusses future work.

2. RELATED WORK

Various approaches have been proposed to deal with the problem of noisy labels in the computer vision domain. One line of approach involves modelling the distribution of noisy and true labels using DNNs [6], [8]. The noise model is used to infer the true labels from the noisy labels. These methods explicitly require a small subset of the data with trustworthy labels. The true label is considered to be a latent random variable and the noise processes is modelled by a linear layer with unknown parameters. Reasoning for using a linear layer is explained in section 6. The expectation-maximization (EM) algorithm [9] is applied to find the parameters of both the linear layer and the neural network to find the correct labels.

A different line of approach involves the soft bootstrapping loss function [7], where the target label is dynamically updated to a convex combination of the original noisy label and the label predicted by the model at that point in time. The updated target label is used for calculating the CCE loss against the predicted label. The underlying concept behind the custom loss function is that label noise causes high deviation between the label predicted by the model and the observed label, due to which the loss is artificially inflated and

AP is supported by a QMUL Principal's studentship. EB is supported by RAEng Research Fellowship RF/128 and a Turing Fellowship.

https://doi.org/10.33682/zyc0-jw35

to reduce this loss component, the model memorizes the noisy label, hence to rectify this situation, current model prediction is added as a consistency objective to the observed label and as learning progresses during training, the predictions of the network tend to become more reliable, negating the effect of label noise to an extent. The soft and hard bootstrapping methods have been evaluated in [4] using audio data and have shown to improve the accuracy of the network. To the authors' knowledge, [4] is the only work in the audio domain with a soft bootstrapping loss function implemented for noise robustness.

3. DATASET

The FSDnoisy18k dataset [4] consists of audio recordings unequally distributed across 20 acoustic event classes: Acoustic guitar, Bass guitar, Clapping, Coin-dropping, Crash cymbal, Dishes pots and pans, Engine, Fart, Fire, Fireworks, Footsteps, Glass, Hi-hat, Piano, Rain, Slam, Squeak, Tearing, Walk, Wind, and Writing.

Audio recordings are of varying lengths, ranging from 30ms to 300ms and can be broadly divided into two categories of labels - noisy and clean. The proportion of noisy/clean labels in terms of the number of audio recordings is 90%/10% and in terms of duration, the proportion is 94%/6%.

The training set consists of 17,585 clips whereas the test set comprises 947 clips. The test set has been formed entirely from the clean label dataset, with the remaining data forming the training set. The number of clips per class ranges from 51 to 170 in the clean subset and 250 to 1000 in the noisy subset. The dataset contains a single label per audio recording.

4. LABEL NOISE

Label noise can either be synthetically injected into the dataset [7] or can already be present in the dataset (real world noise). FSDnoisy18k [4] contains real world label noise since the class label have been annotated based on the associated user tags from freesound [10]. Before elucidating the label noise types, it is important to discuss the data collection and annotation process adopted for the dataset. First a number of freesound user-generated tags were mapped to classes based on Audio set ontology definition [11], post which, for each class, audio clips were selected from freesound, tagged with at least one of the selected user tags. This process generated a number of potential annotations, each of which indicated the presence of a particular class in the given audio recording. The potential annotations were verified via a validation task hosted on FSD online platform [10], where users were required to validate the presence or absence of each of the potential annotations by choosing one of the following options [10]:

1. Present & Predominant (PP) - The sound event is clearly present and predominant.
2. Present but not predominant (PNP) - The sound event is present, but the audio recording also contains other types of sound events and/or background noise.
3. Not Present (NP) - The sound event is not present in the audio recording.
4. Unsure (U) - Note sure if the sound event is present or not.

The audio recordings with annotations rated as PP by a majority of users were included in the training set with curated labels and the test set. The remaining audio clips are included in the training set with noisy labels. The label noise types found in the dataset can be characterized into the following categories:

1. Incorrect/out of vocabulary (OOV) - The accurate label describing the sound event does not correspond to any of the Audio set [11] classes.
2. Incomplete/OOV - Some audio clips contain acoustic events in addition to their accurate labels, however only one sound event is mentioned in the label since the other sound events do not belong to any of the Audio set [11] classes.
3. Incorrect/In vocabulary (IV) - This type of noise consists of classes which are closely related to each other, (e.g. "wind" and "rain") and have been interchanged.
4. Incomplete/In vocabulary (IV) - Two sound events are cooccurring on the audio recording, despite only a single label reported.
5. Ambiguous labels - It is not clear whether the label is correct or not.

The distribution of label noise types in random 15% of per class data in the dataset is shown in Table1. The analysis of noisy labelled training dataset revealed that approximately 60% of the labels contain one or multiple types of label noise and 40% of the labels are correct [4]. As can be seen from Table1, OOV noise constitutes a major portion of the label noise across different classes, either in form of incorrect labels or incomplete labels.

5. BASELINE MODEL

MobilenetV2 [12] is selected as the baseline model. It builds upon the MobilenetV1 [13] architecture which uses depth wise separable convolution layers as the building block. MobilenetV1 consists of a single convolutional layer followed by 13 separable convolution layers. An average pooling layer follows the last separable convolution layer. In separable convolution, the kernel step is divided into depthwise and pointwise convolution operations. A depthwise convolution acts on each channel independently, post which a pointwise convolution acts across all the channels. This factoring reduces the weights of each layer, making the model compact without loss of accuracy. The MobilenetV2 architecture has two new features on top of its predecessor:

1. Linear bottleneck between layers,
2. Residual connection between the bottlenecks.

A pointwise convolution operation in a separable convolution layer leads to an increase in the number of channels. A linear bottleneck layer does the exact opposite. It reduces the amount of data flowing through the network. The residual connection between the linear bottlenecks work in the same manner as Residual Nets [14], where the skip connection serves to assist the flow of gradients through the network. MobilenetV1 was used as the baseline network for the DCASE 2019 Challenge Task 2 [15], a multi label audio tagging task with a large amount of noisy labelled training data and a small amount of manually curated training data. The test data was free of label noise. The source of the manually curated training dataset and test data was the Freesound dataset [10]. This inspired us to explore the MobilenetV2 architecture for our experiments.

6. METHODS

We explore the approaches proposed in [6] and [4] for our experiments on audio tagging with noisy labels.

Table 1: The table has been adopted from [4] and depicts the distribution of label noise types in a random 15% of the noisy data of FSDnoisy18k. The most predominant noise type is Incorrect OOV which refers to the noise type where user tags were not mapping to any existing class.

Label noise type	Amount
Incorrect/OOV	38%
Incomplete/OOV	10%
Incorrect/IV	6%
Incomplete/IV	5%
Ambiguous labels	1%

6.1. Noise modelling with linear layer

This approach intends to find the latent clean label from the noisy labels [6]. A linear noise modelling layer is added on top of the softmax layer. The parameters of the noise layer are denoted by:

$$\theta(i,j) = p(z=j|y=i). \quad (1)$$

z is the observed noisy label and y is the latent clean label. i and j belong to the class set $\{1, 2....k\}$. This parameter representation denotes the probability of observing a noisy label $z = j$ given the latent true label $y = i$. The equation assumes that the noise generation is independent of the input vector and only depends upon the latent clean labels. This is a simple version of the noise adaption layer and can be implemented using a dense layer with softmax activation.

For the complex version of the noise adaptation layer, given the input vector x, network parameters w, noisy label z, noise distribution parameter θ defined in equation (1) and number of classes n, the probability of observing noisy label z given the feature vector x can be denoted by the equation:

$$p(z=j|x;w,\theta) = \sum_{k=1}^{n} p(z=j|y=i;\theta)p(y=i|x;w) \quad (2)$$

The block diagram of the noise adaptation layer approach is shown in Fig. 1. h in the figure denotes the non linear function $h = h(x)$ applied on the input x. w_{noise} refers to the weights of the noise modelling layer. For our model, which is identical to the simple model proposed in [6], the weights are initialized from the prediction output of the baseline model on the training set and are learnt along with the weights of the neural network (w) during the training phase. The linear noise modelling layer is not used during the test phase.

The experimental procedure is as follows: First, the baseline network (MobilenetV2) is trained on the training set containing both noisy and clean labels. The weights of the model are learnt from scratch during the training phase. The prediction output of the baseline network on the training set is used to initialize the weights of the noise modelling layer which is basically a dense layer with softmax activation. The noise modelling layer is added on top of the baseline network and retrained on the training set to learn the weights of the noisy channel.

The noise modelling layer is removed during prediction on the test set, the reason being that we want to see how the transformed baseline network performs on the test set as compared to the original baseline architecture.

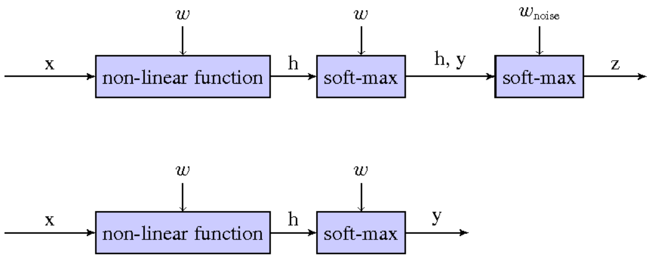


Figure 1: The figure has been adapted from [6]. It is an illustration of the architecture for the training phase (above) and the test phase (below).

6.2. Soft bootstrapping loss

The soft bootstrapping loss was originally introduced in [7] and has been implemented in the audio domain in [4]. The loss function dynamically updates the target labels based on the models' current output. The idea is to pay less attention to the noisy labels, in favour of the model predictions, which are more reliable as the learning progresses. This approach can be expressed by:

$$L_{soft} = -\sum_{k=1}^{n}[\beta y_k + (1-\beta)\hat{y}_k]\log(\hat{y}_k), \quad \beta \in [0,1] \quad (3)$$

$\hat{y}_k$ is the k'th element of the network predictions (the predicted class probabilities), and n is the number of classes. The parameter β is used to assign the weightage of each component in the total loss. The updated target label is a convex combination of the current model's prediction and the (potentially noisy) target label.

7. EXPERIMENTAL SETUP AND METRICS

In this section we discuss the experimental setup and the evaluation metrics adopted for the paper.

7.1. Experimental Setup

Given the nature of the dataset with noisy labels, we are interested in exploring how well the baseline and baseline & linear noise modelling layer would perform on the dataset. The incoming audio is transformed to a 128 band log-mel spectogram using a window size of 1764 (44100(sampling rate) x 0.04(40 ms for a frame)), samples and a hop length of 882 (44100 *0.02(20 ms for overlap)) samples. Since each audio recording is of different length, the duration of each recording is fixed to 2s. The longer recordings were clipped whereas the shorter ones were replicated to obtain a uniform length across the dataset. Both the training set and the test set are scaled using the mean and standard deviation of the training set, post which the class distribution is balanced by oversampling the classes with less samples using the oversampling function from the imblearn library [16].

Data augmentation is also applied as a part of preprocessing. We use mix up data augmentation [17] where new samples are created through a weighted linear interpolation of two existing samples. (x_i, y_i) and (x_j, y_j) are two samples randomly selected from the training set and a convex combination using the parameter $\lambda \in [0, 1]$ which decides the mixing proportions. A new pair of samples (x_k, y_k) is formed using the equations:

$$x_k = \lambda x_i + (1-\lambda)x_j \quad (4)$$

$$y_k = \lambda y_i + (1 - \lambda) y_j \tag{5}$$

The training data is split into training and validation sets with the entire manually verified data used as validation set. The cross entropy loss is used in all experiments except one, where the soft bootstrapping loss is used. An initial learning rate of 0.001 and batch size of 64 samples is used. Each model is trained for 250 Epochs.

The following experimental scenarios are evaluated:

1. Using the baseline network without data augmentation.
2. Using the baseline network with data augmentation.
3. Using the best model from step 2 and adding a dense layer with softmax activation on top of the network. The new network is retrained on the training data with the same setup for training and validation. The weights of the linear layer are initialized using the prediction output from the baseline network. This is in line with the simple model proposed in [6].
4. Training the baseline network from scratch using the soft bootstrapping loss.

7.2. Evaluation Metrics

Classification accuracy was used as the evaluation metric for all the experiments and the results for training, validation and test accuracy have been reported in the results section.

8. RESULTS

Table 2 presents the results of all the experimental approaches mentioned in Section 7.1. Adding a linear noise modelling layer increases the accuracy of the baseline network by approximately 2%. The soft bootstrapping loss function approach also improves the accuracy of the baseline model by approximately 1.5%, indicating that both the approaches are to an extent, helpful in dealing with label noise.

Table 2: Results

Approach	Test Accuracy
Baseline w/o data augmentation	0.649
Baseline with data augmentation	0.667
Baseline with linear noise layer	**0.686**
Baseline with soft bootstrapping loss	0.6825

As can be seen from Fig. 2, the noise modelling approach improves the performance of the baseline network in certain classes such as Coin_dropping, Dishes_and_pots_and_pans,Crash cymbal,Wind,fireworks,Hi-hat and Writing, however the accuracy either decreased or was equivalent to the baseline accuracy for Walk_or_footsteps,Rain,Engine,Glass,Fire,Fart and Tearing. From analysing the FSDnoisy18k [4], it can be inferred that the noise modelling layer improved accuracy in certain classes with significant amount of noisy labels, such as clapping (68% noisy labels), coin dropping (71% noisy labels), crash cymbal (86% noisy labels) and wind (75% noisy labels), however this is not the case with all the classes with high label noise. For some labels such as piano (60% noisy labels), Engine (68% noisy labels), Fire (89% noisy labels), the accuracy score of the noise modelling layer either dropped or stayed constant as compared to the baseline model.

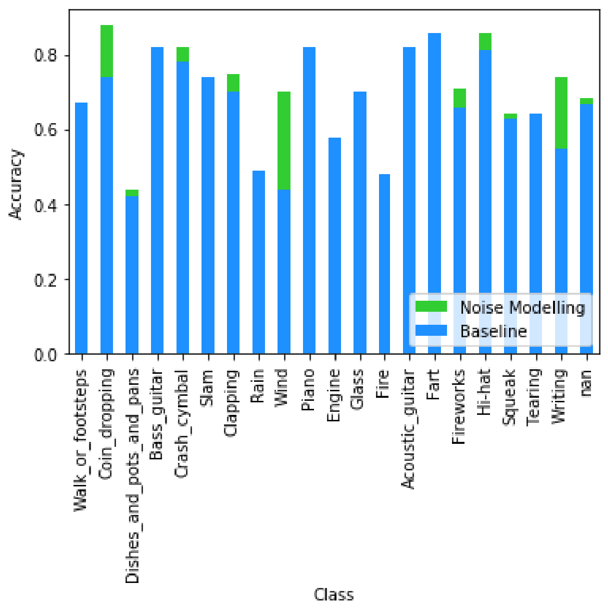


Figure 2: Classwise accuracy score for baseline model(blue) and noise modelling layer (green) is shown. Both graphs have been merged for the purpose of comparison. For all the classes with only blue bars, the accuracy achieved by the noise modelling layer is either equivalent or less than the baseline model and the green bar is a visual indication of the classes where the noise modelling layer was able to improve the accuracy of the baseline model.

This inconsistency in performance improvement indicates towards the hypothesis that the noise modelling approach might only be effective against certain label noise types and not so much against other types. We intend to explore this hypothesis in a more detailed manner in the future.

9. FUTURE WORK & CONCLUSION

In this work, we experimented with an intuitive approach to model the noise distribution of dataset labels and compared it with a noise robust loss function approach. The accuracy increase over the baseline model is encouraging, however we believe that a higher accuracy can be obtained by further tuning of the network and implementing the complex model from [6]. Although the accuracy of the system is lower than the one reported in [4], we consider this our first step in exploration of modelling label noise distribution and hope to achieve better results in the future.

From a future work perspective, we intend to understand as to why the noise modelling layer only can rectify certain kinds of label noise and fails to do so on other kinds of noise, post which we intend to implement the complex model from [6] to evaluate its performance against the baseline model, baseline with simple noise modelling layer and the soft bootstrapping loss model. Our future road map also includes implementing the approach on a multi-label noisy audio dataset and evaluate the performance of the model from different evaluation metrics other than accuracy to gain a better understanding of the underlying concepts.

10. REFERENCES

[1] M. D. P. T. Virtanen and D. Ellis, *Computational Analysis of Sound Scenes and Events*. Springer, 2018.

[2] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. W. Wilson, "CNN architectures for large-scale audio classification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017.

[3] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, X. Favory, J. Pons, and X. Serra, "General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, 2018.

[4] E. Fonseca, M. Plakal, D. P. W. Ellis, F. Font, X. Favory, and X. Serra, "Learning sound event classifiers from web audio with noisy labels," in *Proceedings of ICASSP 2019*, 2019.

[5] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien, "A closer look at memorization in deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[6] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," in *ICLR*, 2017.

[7] S. E. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015.

[8] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. J. Belongie, "Learning from noisy large-scale datasets with minimal supervision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 839–847.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society, series B*, 1977.

[10] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound datasets: a platform for the creation of open audio datasets," in *Proceedings of the 18th International Society for Music Information Retrieval Conference (ISMIR 2017)*, 2017.

[11] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, 2017.

[12] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," in *CVPR*, 2018.

[13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017. [Online]. Available: http://arxiv.org/abs/1704.04861

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[15] "DCASE 2019 Task2." [Online]. Available: http://dcase.community/challenge2019/task-audio-tagging

[16] "Imbalanced learning." [Online]. Available: https://imbalanced-learn.readthedocs.io/en/stable/api.html

[17] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations*, 2018.

ROBUSTNESS OF ADVERSARIAL ATTACKS IN SOUND EVENT CLASSIFICATION

Vinod Subramanian[1,2*], *Emmanouil Benetos*[1†], *Mark Sandler*[1],

[1] Centre for Digital Music, Queen Mary University of London, London, UK
[2] ROLI Ltd., London, UK
{v.subramanian,emmanouil.benetos,mark.sandler}@qmul.ac.uk

ABSTRACT

An adversarial attack is a method to generate perturbations to the input of a machine learning model in order to make the output of the model incorrect. The perturbed inputs are known as adversarial examples. In this paper, we investigate the robustness of adversarial examples to simple input transformations such as mp3 compression, resampling, white noise and reverb in the task of sound event classification. By performing this analysis, we aim to provide insights on strengths and weaknesses in current adversarial attack algorithms as well as provide a baseline for defenses against adversarial attacks. Our work shows that adversarial attacks are not robust to simple input transformations. White noise is the most consistent method to defend against adversarial attacks with a success rate of 73.72% averaged across all models and attack algorithms.

***Index Terms*—** adversarial attacks, deep learning, robust classifiers, sound event classification.

1. INTRODUCTION

Adversarial attacks are algorithms that add imperceptible perturbations to the input signal of a machine learning model in order to generate an incorrect output. The perturbed input signals are called adversarial examples. The existence of adversarial attacks presents a security threat to deep learning models that are used in tasks such as speech recognition and sound event classification, where fooling classifiers can be used to hide malicious content [1, 2]. Adversarial attacks call into question the robustness of machine learning models and whether we can improve them by addressing adversarial attacks.

There is extensive work that investigates the robustness of adversarial attacks against simple input transformations in the task of image recognition. Kurakin, Goodfellow and Bengio [3] apply transformations such as Gaussian noise, JPEG compression etc. to verify the robustness of adversarial attacks. They work towards physical adversarial examples where a photo can be taken of the adversarial example and fool the image recognition model. There is a lot of similar work in image recognition that focuses on different input transformations and their effect on adversarial attacks [4, 5, 6]. Ultimately, this led to 3-d printouts that were adversarial [7], adversarial stickers [8] etc.

In automatic speech recognition, a lot of adversarial attack algorithms have been developed keeping in mind audio specific concerns. Yakura and Samura [9] and Qin et al. [10] developed methods to simulate real world distortion while creating adversarial attacks to make them robust. Liu et al. [11] developed a system to make the process of generating adversarial attacks quicker and quieter. Du et al. [12] developed the Siren Attack that uses Particle Swarm Optimization to speed up generation of adversarial attacks. Besides the Siren Attack, none of these attacks have been tested on other audio tasks such as sound event classification or music classification.

Similarly, most of the work on defenses against adversarial attacks is focused on automatic speech recognition. Research has shown that mp3 compression, band pass filters, adding noise etc. [13, 14, 15, 16] are effective at eliminating adversarial examples in automatic speech recognition. To our knowledge, no work has been done on the effect of input transformations on adversarial attacks in sound event classification. Esmaeilpour et al. [17] developed a support vector machine (SVM) classifier that was more robust to adversarial attacks for sound event classification, but it was at the cost of model performance.

This research aims to establish a body of work that studies the effects of adversarial attacks and defenses in sound event classification. In this paper, we explore simple input transformations such as mp3 compression, resampling, white noise and live reverb as defenses against adversarial attacks across different models. We build off of work done in Subramanian et al. [18] where the performance of popular adversarial attacks was tested against the top submissions to the DCASE 2018 challenge on General purpose audio tagging[1]. We use the adversarial examples generated in that work and run experiments on how robust they are to input transformations.

Our contributions can be summarised as follows: 1. We evaluate the robustness of adversarial examples generated in [18] against simple input transformations. 2. We create a baseline system of defenses against adversarial attacks for sound event classification.

2. METHODOLOGY

2.1. Adversarial attacks

We use a subset of the adversarial attacks in Subramanian et al. [18], the attacks we ignore are the two weaker baseline attacks. All of the attacks used are white box attacks meaning that they have full information of the model they are attacking. The attacks fall into two categories, untargeted and targeted attack. A targeted attack is when the algorithm fools the output classifier to a specific predetermined class. An untargeted attack is when the algorithm reduces the confidence of the current class until the classifier is fooled. The adversarial attacks used in this work are as follows:

*This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No. 765068.

†supported by RAEng Research Fellowship RF/128.

[1]http://dcase.community/challenge2018/task-general-purpose-audio-tagging-results

https://doi.org/10.33682/sp9n-qk06

1. **L-BFGS:** Szegedy et al. [19] introduced one of the first methods for generating adversarial attacks, it is a targeted attack.

   Assume a classifier denoted as $f : \mathbf{R}^m \rightarrow \{1...k\}$ with a loss function $loss_f$. For a given input $x \in \mathbf{R}^m$ and target $t \in \{1....k\}$ we aim to identify the value of perturbation r as formulated below:

   $$\begin{aligned}&\text{Minimize } \|r\|_2 \text{ under the conditions:}\\ &f(x+r) = t\\ &x+r \in [0,1]^m\end{aligned}$$

   The box constraint on $x+r$ is to prevent clipping. The exact computation of this problem is difficult so it is approximated using the box constrained L-BFGS algorithm. So the new equation to minimize is:

   $$c|r| + loss_f(x+r, l) \quad \text{under the conditions } x+r \in [0,1]^m$$

2. **DeepFool**: DeepFool attack was introduced by Moozavi-dezfooli et al. [20]. It is an untargeted attack where the algorithm iteratively linearizes the deep learning model to generate perturbations to fool the classifier. We show how the Deepfool classifier works in the case of a binary classifier, in order to understand how it scales to the multi-class problem we recommend the readers look at the original paper.

   We use the same terminology as defined above for the L-BFGS algorithm. In this case we start by assuming we have a linear classifier f so the relationship between the input x and output can be written as:

   $$f(x) = \omega^T x + b$$

   Here, ω is the weight matrix and b is the bias added. In a binary linear classifier there is a hyper-plane $\mathcal{H}$ that separates the two classes. The hyper-plane is defined such that $x \in \mathcal{H} \rightarrow f(x) = 0$ So, to generate an adversarial attack you need to create a perturbation that pushes the input to the other side of the hyper-plane. This perturbation corresponds to the orthogonal projection of the input onto this hyper-plane. The perturbation for a particular input x_0 denoted by $r(x_0)$ can be computed using the formula:

   $$r(x_0) = -\frac{f(x_0)}{\|\omega\|_2^2}\omega$$

   In the case of a general differentiable binary classifier the model is linearized iteratively and the perturbation is calculated using the formula given above.

3. **Carlini and Wagner -** Carlini and Wagner introduce a strong set of attacks based on the L_0, L_2 and L_∞ distance [21]. This can be used as a targeted and untargeted attack. The problem for adversarial attacks is formulated the same way as Szegedy et al. [19]. The classifier is denoted as C with input x, c is constant:

   $$\begin{aligned}\text{Minimize } &D(x, x+\delta) + c.f(x+\delta)\\ &\text{such that } x+\delta \in [0,1]^n\end{aligned}$$

   Here D is a distance function that is either the L_0, L_2 or L_∞ norm and f is an objective that simplifies the problem such that:

   $$\begin{aligned}C(x+\delta) = t \text{ is true if } f(x+\delta) \leq 0\\ f(x') = (max(Z(x')_i) - Z(x')_t)^+ \; i \neq t\end{aligned}$$

   In the equation for f, Z denotes the penultimate layer of the classifier and t is the target class. This is just one example of the function f many other functions work and can be found in the paper [21]. In this work we use the L_2 version of the Carlini and Wagner attack.

2.2. Input Transformations

We pick input transformations that are likely to occur in the real world when playing an audio file and recording it on a smart phone. The input transformations are as follows:

Mp3 compression - Mp3 compression is a popular format for storing audio files. It is done using the libmp3lame encoding library inside ffmpeg [22]. In our experiments, we compress the adversarial audio examples at three constant bit-rates–48kbps, 128kbps and 320kbps. The lower the bitrate, the higher the information loss will be.

Re-sampling - We are interested in the effects of removing high frequency content on adversarial examples. In our work, re-sampling serves as a low pass filter and is performed using the resampy[2] python library. Resampy uses a band limited sinc interpolation method for re-sampling [23]. We use the "kaiser best" configuration which is the high quality version of resampy. The adversarial audio files in our experiments have a sampling rate of 32kHz. We resample the audio files to 8kHz, 16kHz, and 20kHz and resample it back to 32kHz.

White noise addition - White noise is a standard digital distortion. It is added to the adversarial examples at a signal-to-noise ratio of 20dB, 40dB and 60dB.

Live reverberation - Using the live recording setting of the audio degradation toolbox [24], we obtain the impulse response for the "Great Hall"–one of the live rooms with a very long reverb. We applied said impulse response to add reverb to our adversarial audio files using convolution. After convolution, we eliminate the tail of the audio file in order to preserve its original length.

2.3. Datasest

We use the FSDKaggle2018 dataset [25] introduced for the DCASE 2018 challenge on general-purpose audio tagging. This dataset consists of 41 classes, ranging from urban sounds such as buses, keys jangling and fireworks to musical instruments such as cello, snare drum and Glockenspiel. We use the adversarial audio examples generated on this dataset in Subramanian et al. [18]. For the untargeted attacks we use 6 audio files per class making a total of 246 audio files per model per attack. For the targeted attack we use a subset of 6 classes and for each class we generate 5 targeted adversarial attacks to each of the other 5 classes. This makes 180 audio files per model per attack. Since the adversarial attack algorithms are not 100% effective the actual number of adversarial examples are a bit lower than the number indicated above.

[2]https://github.com/bmcfee/resampy

Model	Training	Test
VGG13	0.9714	0.8093
CRNN	0.9768	0.8437
GCNN	0.9803	0.8437
dense_mel	0.9876	0.89875
dense_wav	0.9698	0.86125

Table 1: Model performance on training and test data.

2.4. Models

We use the DenseNet models described by Jeong and Lim [26]. The DenseNet model concatenates the input to the output for each module. We use two versions of the architecture, the first version uses log mel spectrogram as input (dense_mel) to the model and the second one uses raw audio (dense_wav) as the input to the model.

We use three models from Iqbal et al. [27], the VGG13, CRNN and GCNN networks. VGG13 is a convolutional neural network inspired by the VGG13 architecture. CRNN is a convolutional recurrent neural network that uses a bidirectional RNN after the convolutional layers. The GCNN is a gated convolutional neural network where the gated component is inspired from Long-Short Term Memory (LSTMs). The input to all three models is a log mel spectrogram.

Table 1 shows the training and test accuracy for each of the models on the FSDKaggle2018 dataset [25]. We pick these models because they were the top submissions for the DCASE 2018 challenge on "General purpose audio-tagging".

2.5. Experiment and metrics

The experimental setup has three sets of labels. First is the ground truth, which is the label associated with the audio file from the FSDKaggle2018 dataset [25]. The second is the adversarial label, generated by applying adversarial attacks on the audio file from the aforementioned dataset. The third is the transformed label, which is generated by running each audio file through each of the input transformations separately. Once these transformed inputs are run through the model, it generates the transformed label. Our task is to verify how effective a defense and input transformation is against an adversarial attack. We compare how many transformed labels are ground truth, adversarial, or different from both.

A good defense would convert a lot of the transformed labels to the ground truth; however, if an adversarial attack is robust, a lot of the transformed labels will remain the adversarial labels. We use signal-to-noise ratio and output confidence values generated from Subramanian et al. [18] to explain the results.

3. RESULTS AND DISCUSSION

Table 2 provides a reference for how the defenses affect the ground truth audio data before an adversarial attack is performed. Table 3 compares the effectiveness of mp3 compression, white noise addition, re-sampling and live reverb averaged across all of the models and adversarial attack algorithms. In general, the numbers look as we expect: the more distortion we add to the adversarial example, the more likely it will stop being an adversarial example. The best defense against the adversaries on average is adding noise at 20dB. As we raise the volume of noise to 40dB, the number of audio examples that are destroyed lowers; however, the number of adversarial examples that are classified as a different label is lowered as well.

Transform	GT	Diff
mp3 48k	93.98	6.02
mp3 128k	99.92	0.08
mp3 320k	100	0
noise 20dB	86.67	13.33
noise 40dB	97.89	2.11
noise 60dB	99.84	0.16
sr 8kHz	55.36	44.63
sr 16kHz	85.77	14.23
sr 20kHz	91.87	8.13
live reverb	82.85	17.15

Table 2: Summary of defenses on ground truth data.

Transform	Adv	GT	Diff
mp3 48k	45.02	50.23	4.75
mp3 128k	83.66	15.71	0.63
mp3 320k	91.55	8.26	0.17
noise 20dB	3.40	73.72	22.87
noise 40dB	31.68	63.69	4.61
noise 60dB	81.00	17.58	1.41
sr 8kHz	27.93	41.71	30.34
sr 16kHz	42.36	47.85	9.78
sr 20kHz	47.90	44.89	7.20
live reverb	4.09	67.07	28.83

Table 3: Summary of performance given as a percentage of adversarial examples that remain adversarial, that go back to ground truth and that change completely.

Live reverb is the second most successful defense against the adversaries. These simple input transformations can defend against adversaries, showing that there is a need to create more powerful attacks against sound event classification.

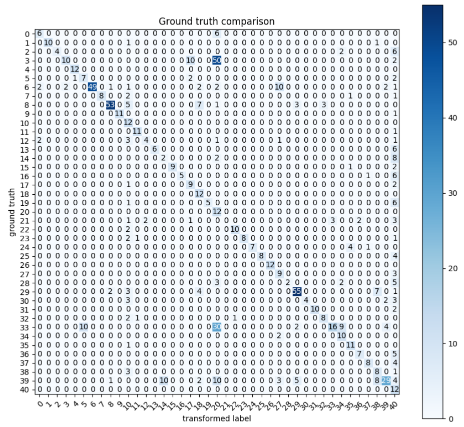
The next table, Table 4, shows how each model behaves in response to these input transformations on the adversarial attacks. We do not show all the data in the interest of space; instead, we show the best two defenses for each model. Across all the models, adding white noise is an effective defense against adversarial attacks. Besides adding white noise, reverb is one of the better defenses. In general, the defenses are more successful for the DenseNet models than for the other three models. One of the reasons for this could be because the adversarial attacks for the DenseNet model are optimized on the output probabilities; whereas the other three models are optimized on the output scores. This means that optimizing an adversarial attack for the DenseNet models only needs to maximize the relative score of the desired class.

Between the DenseNet models the raw audio configuration of the DenseNet behaves differently. Resampling at 20kHz successfully eliminates adversarial examples at 97.18%. For the mel spectrogram configuration of the DenseNet resampling at 20kHz is only successful for 72.36% of the cases. This strongly suggests that the model is sensitive to different changes between the mel spectrogram and raw audio inputs. We speculate that the perceptual weights introduced by the mel spectrogram make that version of the DenseNet give less importance to higher frequencies. This means that losing frequency above 20kHz would impact the mel spectrogram model less adversely than the raw audio model.

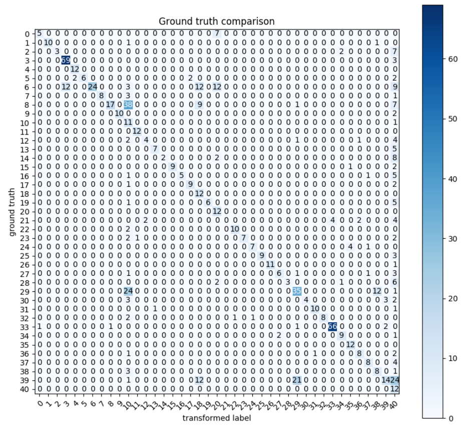
In the cases where the defenses are less effective, we want to know if the distribution of the adversarial examples on which the input transforms are effective or ineffective are similar. We pick the

Model	Best	Adv	GT	Diff
dense_mel	noise 20dB	1.52	97.86	0.61
	noise 40dB	6.25	92.97	0.76
dense_wav	sr 20kHz	1.46	97.55	0.97
	mp3 48k	1.58	97.18	1.22
VGG13	noise 20dB	4.34	58.74	38.89
	live	5.05	57.69	37.25
CRNN	noise 20dB	4.70	65.92	29.37
	noise 40dB	27.37	62.86	9.75
GCNN	noise 40dB	27.81	64.31	7.86
	live	5.98	57.74	36.26

Table 4: Top 2 defenses against the different model architectures averaged over all the adversarial attacks.



(a) VGG13 confusion matrix with live reverb input transform.



(b) VGG13 confusion matrix with noise 20dB input transform.

Figure 1: Comparison of two confusion matrices. The confusion matrix plots the ground truth against the transformed label.

VGG13 model to compare since the success of noise at 20dB and live reverb is very close. We plot the confusion matrices for the two scenarios in figure 1. Noise at 20dB is an effective defense for label 3 (Bass drum) and 33 (Snare drum), but live reverb is not a good defense, live reverb is successful against label 8 (Clarinet) but, noise at 20dB is not as effective etc. Interestingly label 21 (Gunshot or gunfire) has 0% success for both defenses. Evidently, audio files from different labels seem to have disparate properties; therefore, applying each distortion type will affect the differing labels uniquely This would mean that while developing an adversarial attack that works in the real world, we need to come up with a solution that is robust to different types of distortion.

Attack	SNR/Conf	Best	Adv	GT	Diff
deepfool	50/0.35	noise 20dB	4.14	77.23	18.61
		noise 40dB	16.09	76.17	7.72
		live	4.22	70.37	25.04
C&W untargeted	51.26/0.71	noise 20dB	5.88	74.83	19.28
		live	5.88	70.17	23.93
		noise 40dB	43.79	50.98	5.22
L-BFGS	56.25/0.98	noise 20dB	0.40	69.56	30.02
		noise 40dB	32.33	66.44	1.22
		live	1.63	62.77	35.59
C&W Targeted	50.49/0.97	noise 20dB	1.31	70.6	28.07
		noise 40dB	36.32	61.52	2.15
		live	3.46	60.93	35.60

Table 5: Table shows the top 3 defenses against input transforms for the different adversarial attacks averaged over all the models. The SNR in dB and label confidence (Conf) as probability of the adversarial examples are presented as averaged over all the models.

Table 5 shows the performance of the top 3 defenses for each adversarial attack algorithm. The targeted attacks seem more robust to these input transformations than the untargeted attacks but not by too much. We expect Carlini and Wagner to be more robust because it is a more powerful attack than Deepfool and L-BFGS as is shown in Subramanian et al. [18].

The fact that the SNR is very high for all of the attack algorithms is good from a real world perspective because that means that the noise added to make the audio files adversarial is less likely to be perceived. However, it is possible that the noise is being masked by the input transformations which makes the adversarial attacks not very robust. Given that the SNR is so high there is a lot of headroom to improve adversarial attack algorithms for sound event detection where we increase the amount of noise added without compromising too much on how perceivable the adversarial attacks are.

4. CONCLUSION

We show that simple input transformations such as mp3 compression, re-sampling, white noise addition and live reverb are effective defenses against popular adversarial attacks. White noise at 20dB is the most consistent method to defend against adversarial attacks with live reverb being a close second. The raw audio version of the DenseNet behaves differently with re-sampling at 20kHz, being the most effective defense. This suggests that different input representations affect the type of features that a deep learning model can learn by making the deep learning model focus on different frequency bands. Generally, we hope that these defenses give a sense of current weaknesses in research on adversarial attacks for audio.

Another area that we plan to explore is trying to explain the presence of adversarial attacks in sound event classification. We aim to combine research from interpretability and adversarial attacks in order to work towards explaining and interpreting deep learning.

5. REFERENCES

[1] X. Yuan, Y. Chen, Y. Zhao, Y. Long, X. Liu, K. Chen, S. Zhang, H. Huang, X. Wang, and C. A. Gunter, “Commandersong: A systematic approach for practical adversarial voice

recognition," in {*USENIX*} *Security Symposium*, USA, 2018, pp. 49–64.

[2] N. Carlini and D. Wagner, "Audio adversarial examples: Targeted attacks on speech-to-text," in *IEEE Security and Privacy Workshops (SPW)*, USA, May 2018, pp. 1–7.

[3] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *arXiv:1607.02533*, 2016.

[4] C. Guo, M. Rana, M. Cisse, and L. van der Maaten, "Countering adversarial images using input transformations," in *International Conference on Learning Representations (ICLR)*, Canada, April 2018.

[5] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille, "Mitigating adversarial effects through randomization," in *arXiv:1711.01991*, 2017.

[6] G. K. Dziugaite, Z. Ghahramani, and D. M. Roy, "A study of the effect of JPG compression on adversarial images," Aug. 2016, arXiv: 1608.00853.

[7] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *International Conference on Machine Learning (ICML)*, Sweden, Jul 2018, pp. 284–293.

[8] K. Eykholt, I. Evtimov, E. Fernandes, T. Kohno, B. Li, A. Prakash, A. Rahmati, and D. Song, "Robust physical-world attacks on machine learning models," in *arXiv:1707.08945*, 2017.

[9] H. Yakura and J. Sakuma, "Robust Audio Adversarial Example for a Physical Attack," *arXiv:1810.11793*, Oct. 2018.

[10] Y. Qin, N. Carlini, G. Cottrell, I. Goodfellow, and C. Raffel, "Imperceptible, robust, and targeted adversarial examples for automatic speech recognition," in *International Conference on Machine Learning (ICML)*, USA, Jun 2019, pp. 5231–5240.

[11] X. Liu, X. Zhang, K. Wan, Q. Zhu, and Y. Ding, "Towards Weighted-Sampling Audio Adversarial Example Attack," *arXiv:1901.10300*, Jan. 2019.

[12] T. Du, S. Ji, J. Li, Q. Gu, T. Wang, and R. Beyah, "SirenAttack: Generating Adversarial Audio for End-to-End Acoustic Systems," *arXiv:1901.07846*, Jan. 2019.

[13] K. Rajaratnam, K. Shah, and J. Kalita, "Isolated and Ensemble Audio Preprocessing Methods for Detecting Adversarial Examples against Automatic Speech Recognition," in *Conference on Computational Linguistics and Speech Processing (ROCLING)*, Taiwan, Oct. 2018, pp. 16–30.

[14] K. Rajaratnam and J. Kalita, "Noise flooding for detecting audio adversarial examples against automatic speech recognition," in *International Symposium on Signal Processing and Information Technology (ISSPIT)*. USA: IEEE, Dec 2018, pp. 197–201.

[15] Z. Yang, B. Li, P.-Y. Chen, and D. Song, "Characterizing audio adversarial examples using temporal dependency," in *International Conference on Learning Representations (ICLR)*, USA, 2019.

[16] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, S. Li, L. Chen, M. E. Kounavis, and D. H. Chau, "Shield: Fast, practical defense and vaccination for deep learning using jpeg compression," in *International Conference on Knowledge Discovery*, USA, 2018, pp. 196–204.

[17] M. Esmaeilpour, P. Cardinal, and A. L. Koerich, "A Robust Approach for Securing Audio Classification Against Adversarial Attacks," *arXiv:1904.10990*, Apr. 2019.

[18] V. Subramanian, E. Benetos, N. Xu, S. McDonald, and M. Sandler, "Adversarial attacks in sound event classification," in *arXiv:1907.02477*, 2019.

[19] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *International Conference on Learning Representations (ICLR)*, Canada, 2014.

[20] S. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: A simple and accurate method to fool deep neural networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, USA, June 2016, pp. 2574–2582.

[21] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *arXiv:1608.04644*, 2016.

[22] F. Developers, "ffmpeg tool (version beld324)," http://ffmpeg.org, 2016.

[23] J. O. Smith, *Digital Audio Resampling Home Page*. https://ccrma.stanford.edu/ jos/resample/, January 28, 2002.

[24] M. Mauch and S. Ewert, "The audio degradation toolbox and its application to robustness evaluation," in *International Society for Music Information Retrieval Conference (ISMIR)*, Brazil, 2013.

[25] E. Fonseca, M. Plakal, F. Font, D. P. W. Ellis, X. Favory, J. Pons, and X. Serra, "General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline," *CoRR*, vol. abs/1807.09902, 2018. [Online]. Available: http://arxiv.org/abs/1807.09902

[26] I.-Y. Jeong and H. Lim, "Audio tagging system for dcase 2018: focusing on label noise, data augmentation and its efficient learning," DCASE2018 Challenge, Tech. Rep., 2018.

[27] T. Iqbal, Q. Kong, M. D. Plumbley, and W. Wang, "General-purpose audio tagging from noisy labels using convolutional neural networks," in *Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, UK, 2018, pp. 212–216.

IMPROVEMENT OF DOA ESTIMATION BY USING QUATERNION OUTPUT IN SOUND EVENT LOCALIZATION AND DETECTION

Yui Sudo, Katsutoshi Itoyama, Kenji Nishida

Department of Systems and Control Engineering, School of Engineering, Tokyo Institute of Technology, 2-12-1 Ookayama, Meguro-ku, Tokyo 1528552, Japan
{sudo, itoyama, nishida}@ra.sc.e.titech.ac.jp

Kazuhiro Nakadai

Department of Systems and Control Engineering, School of Engineering, Tokyo Institute of Technology, 2-12-1 Ookayama, Meguro-ku, Tokyo 1528552, Japan
Honda Research Institute Japan Co., Ltd., 8-1 Honcho, Wako, Saitama 351-0188, Japan
nakadai@jp.honda-ri.com

ABSTRACT

This paper describes improvement of Direction of Arrival (DOA) estimation performance using quaternion output in the Detection and Classification of Acoustic Scenes and Events (DCASE) 2019 Task 3. DCASE 2019 Task3 focuses on the sound event localization and detection (SELD) which is a task that simultaneously estimates the sound source direction in addition to conventional sound event detection (SED). In the baseline method, the sound source direction angle is directly regressed. However, the angle is a periodic function and it has discontinuities which may make learning unstable. Specifically, even though -180 deg and 180 deg are in the same direction, a large loss is calculated. Estimating DOA angles with a classification approach instead of regression can solve such instability of discontinuities but this causes limitation of resolution. In this paper, we propose to introduce the quaternion which is a continuous function into the output layer of the neural network instead of directly estimating the sound source direction angle. This method can be easily implemented only by changing the output of the existing neural network, and thus does not significantly increase the number of parameters in the middle layers. Experimental results show that proposed method improves the DOA estimation without significantly increasing the number of parameters.

Index Terms— Sound event localization and detection, direction of arrival, inter-channel phase difference, quaternion, convolutional recurrent neural networks

1. INTRODUCTION

Sound event detection (SED) is a rapidly developing research area that aims to analyze and recognize a variety of sounds in urban and natural environments. Compared to audio tagging, event detection also involves estimating the time of occurrence of sounds. Automatic recognition of sound events would have a major impact in several applications. For example, SED has been drawing a surging amount of interest in recent years with applications including audio surveillance [1], healthcare monitoring [2], urban sound analysis [3], multimedia event detection [4] and bird call detection [5]. In an actual application, more convenient application can be realized by simultaneously performing sound source localization as well as detection of a sound event occurrence interval. For example, in the case of an audio surveillance, it is useful to detect the direction of the anomalous sound. Alternatively, individual sound events can be identified even when events of the same class are overlapped. Also, regarding the detection of overlapping sound events, more rational detection is possible by using spatial information.

Task 3 of the DCASE 2019 Challenge focuses on locating and detecting sound events (SELD) for overlapping sound sources [6]. A recently developed system called SELDnet was used as a baseline system. SELDnet uses magnitude spectrograms and phase spectrograms as input features to jointly train SED and DOA estimation purposes [7]. Regarding input features, it has been reported that simply using sinIPD and cosIPD (inter-channel phase difference) as input features for the neural network improves the performance in speech separation [8]. Meanwhile, DOA angle has been directly predicted in many research. During training, the difference between the correct angle and the estimated angle is calculated as a loss. Since the angle is a periodic function, it has discontinuities. Specifically, even though -180 deg and 180 deg are in the same direction, a large loss is calculated, which may make learning unstable. Regarding the discontinuity problem in rotation angle estimation, camera pose regression has been proposed that estimates camera position and orientation by using quaternion in computer vision [9-12].

This paper proposes a model that replaces input features of baseline system and DOA output with sinIPD, cosIPD and quaternion respectively. Since this method can be implemented without changing the middle layer of the network, it is easy to implement with almost no increase in the number of parameters of the existing model. Details of the proposed method are explained in the following sections.

2. METHOD

The entire network of Sound event localization and detection is shown in Fig. 1. The time-series sound source is input to the convolutional recurrent neural network (CRNN) [13] after feature extraction block. The CRNN is consists of three blocks, including three layers of convolutional neural network (CNN), two layers of bi-directional recurrent neural network (RNN) and two fully connected layers. There are two branches throughout the joint layer block. One is for SED, and the other is for DOA estimation.

https://doi.org/10.33682/jj50-hm12

The difference from the baseline is that the source direction angle is not directly estimated but through regressing quaternions. The estimated quaternions are converted to source direction angles by post-processing. Details of feature extraction, network, and post-processing is described in the following sections.

2.1. Feature extraction

The input to this method is multi-channel audio with a sampling rate of 48 kHz. At first, short time Fourier transformation (STFT) is applied using a 40 ms long Hanning window and M points (M=2048) from 20 ms hop length. Then, for each STFT obtained, select a reference microphone, p, and other non-reference microphones, q. As a spectral feature, an amplitude spectrogram of only the reference microphone is used. Meanwhile, we use the following equations to extract spatial features,

$$\cos IPD(t,f,p,q) = \cos(\theta_{t,f,p,q}), \tag{1}$$

$$\sin IPD(t,f,p,q) = \sin(\theta_{t,f,p,q}), \tag{2}$$

where $\theta_{t,f,p,q} = \angle x_{t,f,p} - \angle x_{t,f,q}$ is the phase difference between the STFT coefficients $x_{t,f,p}$ and $x_{t,f,q}$ at time t and frequency f of the signals at microphones p and q. In this paper, 6-channel sinIPD and cosIPD are used as spatial features for IPD of three combinations (1ch-2ch, 1ch-3ch, 1ch-4ch). That is, a total of 7-channel features consisting of one amplitude spectrogram of the reference microphone and the 6-channel spatial features are input to the neural network.

2.2. Network architecture

In order to verify the effect of quaternion estimation, basically the same network as the baseline system shown in Fig. 1 is used. A sequence of T spectrogram frames (T = 128), extracted in the feature extraction block, is fed to the three convolutional layers that extract shift-invariant features using P filters each (P=64). Batch normalization is used after each convolutional layers. Dimensionality reduction of the input spectrogram feature is performed using max pooling operation only along the frequency axis, which is called frequency pooling in [13]. The temporal axis is untouched to keep the resolution of the output unchanged from the input dimension. The temporal structure of the sound events is modeled using two bi-directional recurrent layers with Q gated recurrent units (GRU) each (Q=128). Finally, the output of the recurrent layer is shared between two fully connected layer (FC) branches each producing the SED as multiclass multilabel classification and DOA as multi-output regression; together producing the SELD output. The first FC layer contains R nodes each with linear activation. The SED output obtained is the class-wise probabilities for the C classes in the dataset at each of the T frames of input spectrogram sequence, resulting in a dimension of $T \times C$. The localization output estimates, for each time frame T, quaternions representing rotation in the azimuth direction and elevation direction for each of the C classes i.e., if multiple instances of the same sound class occur in a time frame the SELDnet localizes either one or oscillates between multiple instances. The overall dimension of localization output is $T \times 4C$, where $4C$ represents the class-wise $\sin(\theta_{azimth})$, $\cos(\theta_{azimuth})$ and $\sin(\theta_{elevation})$, $\cos(\theta_{elevation})$,

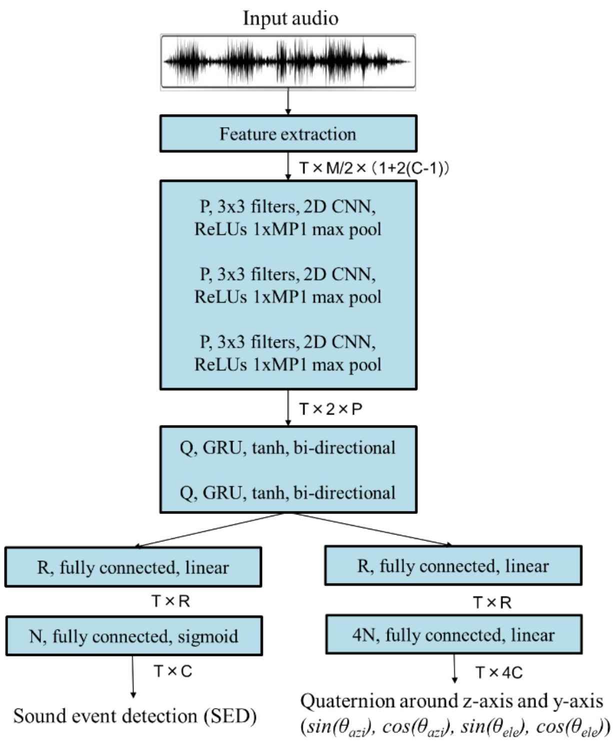

Input audio
Feature extraction
T×M/2×(1+2(C-1))
P, 3x3 filters, 2D CNN, ReLUs 1xMP1 max pool
P, 3x3 filters, 2D CNN, ReLUs 1xMP1 max pool
P, 3x3 filters, 2D CNN, ReLUs 1xMP1 max pool
T×2×P
Q, GRU, tanh, bi-directional
Q, GRU, tanh, bi-directional
R, fully connected, linear
R, fully connected, linear
T×R
T×R
N, fully connected, sigmoid
4N, fully connected, linear
T×C
T×4C
Sound event detection (SED)
Quaternion around z-axis and y-axis
(sin(θazi), cos(θazi), sin(θele), cos(θele))

Figure 1: Convolutional recurrent neural network for SELD.

Table 1: An example of the outputs and post-processing results.

Output of SED branch			Output of DOA branch				DOA angle (post processing)	
Sound event class	SED prediction	Sound activity	Ground truth of quaternion around z-axis		Ground truth of quaternion around y-axis		Azimuth	Elevation
			$\sin(\theta azi)$	$\cos(\theta azi)$	$\sin(\theta ele)$	$\cos(\theta ele)$		
SPEECH	0.8	active	1.0	0.0	0.5	0.9	90	30
CAR	0.1	inactive	0.1	0	-0.1	-0.1	inactive	inactive
…	0.2	inactive	0	0.1	-0.1	0	inactive	inactive
DOG	0.7	active	0.0	-1.0	0.0	1.0	-180	0
…	0.1	inactive	0.1	0	-0.1	0	inactive	inactive
TRAIN	0.1	inactive	0	-0.1	0.1	-0.1	inactive	inactive

which describes quaternion around z-axis and y-axis. Note that $\theta_{azimuth}$ and $\theta_{elevation}$ do not represent the phase of STFT but represent the sound source direction angles of azimuth and elevation. A sound event class is said to be active if its probability in SED output is greater than the threshold of 0.5, otherwise, the sound class is considered to be inactive. The presence of sound class in consecutive time frames gives the onset and offset times, and the corresponding DOA estimates from the localization output gives the spatial location with respect to time. A crossentropy loss is employed for detection output, while a mean square error loss on the quaternion distance between reference and estimated locations is employed for the localization output. The combined convolutional recurrent neural network architecture is trained using Adam optimizer and a weighted combination of the two output losses. Specifically, the localization output is weighted ×50 more than the detection output same as the baseline. The number of parameters is 615,799 while baseline has 613,537. This method does not significantly increase the number of parameters compared to baseline.

2.3. Output and post processing

In the baseline system, the sound source direction angle is directly estimated by regression, but since the angle is a periodic function, it has discontinuities. Specifically, -180 deg and 180 deg are in the same direction but are trained as a large loss as shown in Fig. 2. Therefore, in this paper, the source direction angle is estimated using quaternions defined by (4). Specifically, the output of the regression branch is changed to quaternion, and the post-processing converts it into the sound source direction angle.

$$\boldsymbol{a} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}, \tag{3}$$

$$\boldsymbol{q} = \begin{pmatrix} \cos(\theta/2) \\ a_x \sin(\theta/2) \\ a_y \sin(\theta/2) \\ a_z \sin(\theta/2) \end{pmatrix}, \tag{4}$$

where, $\boldsymbol{a}=[a_x, a_y, a_z]$ is a unit vector representing the rotation axis and $\boldsymbol{q}$ is the definition of quaternion. For the angle estimation in the azimuth direction, $\boldsymbol{a}=[0, 0, 1]$ is substituted and in the elevation direction angle, $\boldsymbol{a}=[0, 1, 0]$ is substituted. That is, the output of the network can be trained with $\sin(\theta/2)$ and $\cos(\theta/2)$ as ground truth. In order to simplify the calculation, θ is substituted instead of $\theta/2$. Since $\sin(\theta)$ and $\cos(\theta)$ are continuous function, it is possible to train efficiently. Additionally, $\sin(\theta)$ and $\cos(\theta)$ always have different values. If the sound source is inactive i.e., if the ground truth of the SED is 0, then $\sin(\theta) = 0$, $\cos(\theta) = 0$ are used as ground truth. During inference, post-processing is performed as in the following equation to calculate the sound source direction angle as described in (5).

$$\theta = \begin{cases} arctan\left(\frac{\sin(\theta)}{\cos(\theta)}\right) + \pi & (if \sin(\theta) \geq 0, \cos(\theta) < 0) \\ arctan\left(\frac{\sin(\theta)}{\cos(\theta)}\right) - \pi & (if \sin(\theta) < 0, \cos(\theta) \geq 0). \\ arctan\left(\frac{\sin(\theta)}{\cos(\theta)}\right) & (\text{otherwise}) \end{cases} \tag{5}$$

As shown in the Fig. 2, if both $\sin(\theta)$ and $\cos(\theta)$ values are known, the sound source direction angle can be uniquely calculated in the range of -180 to 180 degrees. However, if both $\sin(\theta)$ and $\cos(\theta)$ are within the range of -0.2 to 0.2, it is regarded as inactive, calculation of the sound source direction angle is not performed.

3. DEVELOPMENT RESULTS

Polyphonic sound event detection and localization are evaluated with individual metrics for SED and DOA estimation. For SED, segment-based error rate (ER) and F-score [14] are calculated in one-second lengths. A lower ER or a higher F-score indicates better performance. For DOA, DOA error and frame recall are used. A lower DOA error and a higher frame recall are better. Using the cross-validation split provided for this task, Tab. 1 shows the development set performance for the proposed method. As shown in Tab. 1, the DOA error decreased but the index related to SED did not improve. The reason is that the quaternion output model proposed in this paper backpropagates an error to the DOA branch but does not directly affect the SED branch. Moreover, since the SED results are not improved, it is not considered that the spatial information using sinIPD and cosIPD did not show a significant improvement. However, in terms of the number of parameters, the baseline uses features of a total of 8-channel features consisting of amplitude spectrum and phase spectrum, while the proposed method uses 7-channel features consisting of a reference spectrogram and sinIPD, cosIPD. Therefore, the proposed method is considered to have some dimensional reduction effect.

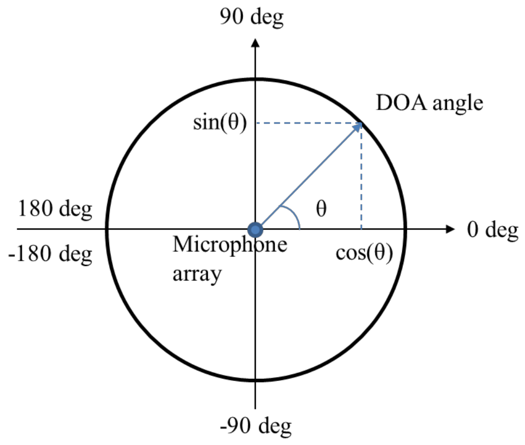


Figure 2: Unit circle with origin of microphone array position, and DOA.

Table 2: Cross validation results for the development set.

	Error rate	F score	DOA error	Frame recall
baseline-ambisonic	0.34	0.799	28.5	0.854
baseline-microphone array	0.35	0.80	30.8	0.840
proposed method	0.35	0.81	**11.5**	0.835

4. CONCLUSION

In this paper, as an approach applicable to the existing neural network model, we propose a method to replace the output and input with quaternion and sinIPD, cosIPD respectively. In the DCASE 2019 Task 3, verification was performed by replacing only the baseline input and output with the proposed method. From the experimental results, it was found that changing the output of the DOA branch to quaternion can improve the DOA estimation without changing the existing neural network model. However, because there were no changes in the SED branch, the performance associated with SED remained comparable. Regarding spatial features using sinIPD and cosIPD, although the performance was not improved, the dimension of feature was reduced. Since this method can be implemented with almost no change to existing network models, further improvement of DOA estimation is expected by using it in combination with other high-performance models.

5. ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant No.16H02884, 17K00365, and 19K12017.

6. REFERENCES

[1] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, “Reliable detection of audio events in highly noisy environments,” Pattern Recognition Letters, vol. 65, pp. 22–28, 2015.

[2] S. Goetze, J. Schroder, S. Gerlach, D. Hollosi, J.-E. Appell, and F. Wallhoff, “Acoustic monitoring and localization for social care,” Journal of Computing Science and Engineering, vol. 6, no. 1, pp. 40–50, 2012.

[3] J. Salamon and J. P. Bello, “Feature learning with deep scattering for urban sound analysis,” in 2015 23rd European Signal Processing Conference (EUSIPCO). IEEE, 2015, pp. 724–728.

[4] Y. Wang, L. Neves, and F. Metze, “Audio-based multimedia event detection using deep recurrent neural networks,” in 2016 IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2016, pp. 2742–2746.

[5] D. Stowell and D. Clayton, “Acoustic event detection for multiple overlapping similar sources,” in 2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), 2015, pp. 1–5

[6] http://dcase.community/challenge2019/ task-sound-event-localization-and-detection

[7] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, “Sound event localization and detection of overlapping sources using convolutional recurrent neural networks,” IEEE Journal of Selected Topics in Signal Processing, vol. 13, no. 1, pp. 34–48, 2019.

[8] Z.-Q. Wang, J. Le Roux, and J. R. Hershey, “Multi-Channel Deep Clustering: Discriminative Spectral and Spatial Embeddings for Speaker-Independent Speech Separation,” in IEEE International Conference on Acoustics, Speech and Signal Processing, 2018

[9] E. B. Dam, M. Koch, and M. Lillholm. Quaternions, interpolation and animation, volume 2. 1998.

[10] S. Altmann. Rotations, Quaternions, and Double Groups. Dover Publications, 2005.

[11] A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In Proceedings of IEEE International Conference on Computer Vision (ICCV), 2015.

[12] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[13] Cakir, E., Parascandolo, G., Herittola, T., Huttunen, H. and Virtanen, T., “Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection, IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 25, Issue 6, June, 2017, pp. 1291 - 1303

[14] A. Mesaros, T. Heittola, and T. Virtanen, “Metrics for polyphonic sound event detection,” in Applied Sciences, vol. 6, no. 6, 2016.

HIERARCHICAL SOUND EVENT CLASSIFICATION

Daniel Tompkins[1], *Eric Nichols*[1], *Jianyu Fan*[1,2]

[1] Microsoft, Dynamics 365 AI Research, Redmond, WA, 98052, USA
[2] Simon Fraser University, Metacreation Lab, Surrey, BC, V3T 0A3, Canada
{daniel.tompkins, eric.nichols, t-jiafan}@microsoft.com

ABSTRACT

Task 5 of the Detection and Classification of Acoustic Scenes and Events (DCASE) 2019 challenge is "urban sound tagging". Given a set of known sound categories and sub-categories, the goal is to build a multi-label audio classification model to predict whether each sound category is present or absent in an audio recording. We developed a model composed of a preprocessing layer that converts audio to a log-mel spectrogram, a VGG-inspired Convolutional Neural Network (CNN) that generates an embedding for the spectrogram, a pre-trained VGGish network that generates a separate audio embedding, and finally a series of fully-connected layers that converts these two embeddings (concatenated) into a multi-label classification. This model directly outputs both fine and coarse labels; it treats the task as a 37-way multi-label classification problem. One version of this network did better at the coarse labels `CNN+VGGish1`); another did better with fine labels on Micro AUPRC `CNN+VGGish2`). A separate family of CNN models was also trained to take into account the hierarchical nature of the labels (`Hierarchical1`, `Hierarchical2`, and `Hierarchical3`). The hierarchical models perform better on Micro AUPRC with fine-level classification.

Index Terms— Sound Event Classification, CNN, Hierarchy

1. INTRODUCTION

A soundscape recording is a "recording of sounds at a given location at a given time, obtained with one or more fixed or moving microphones" [1]. Automatic sound event classification has many applications, such as abnormal event detection [2], acoustic ecology [3] and urban noise pollution monitoring [4]. SONYC (Sounds of New York City) is a research project for mitigating urban noise pollution [4]. Because the SONYC sensor network collects millions of audio recordings, it is important to automatically detect and classify the collected audio recordings for noise pollution monitoring. Therefore, researchers designed the urban sound tagging task, which is to predict whether each of 23 sources of noise pollution is present or absent in the 10-second scene recorded. Researchers recruited individuals on Zooniverse, a web platform for citizen science, to provide weak labels for collected audio recordings based on a taxonomy involving both coarse- and fine-grained classes [4].

The relationship between coarse-grained and fine-grained tags is hierarchical. Therefore, we designed a hierarchical sound event classification model. Previous studies demonstrate that convolutional neural networks (CNNs) can achieve state-of-the-art performance in sound event classification tasks[5]. Thus, we adopt the CNNs structure for our model design. In our approach to audio event classification, we assessed two categories of methods: 1) creating and training a new model trained only on the DCASE Task 5 Challenge dataset, and 2) building a model that uses as input an embedding vector generated by an external model trained on a larger, different dataset. Both approaches have various advantages and disadvantages. Creating a new model results in a model trained for the specific sounds, environments, and sensors from the dataset, which can potentially offer better precision, yet the limited size of the dataset can reduce training success. Re-purposing a pre-trained model such as VGGish, trained on AudioSet [5], has the advantage of starting with a model that was trained on a large and diverse dataset, but the disadvantage of disregarding input features that might have been discarded by the VGGish model, reducing the ability to capture nuanced distinctions between specific classes in the DCASE Task 5 dataset.

Our approach combined the two approaches, in an attempt to benefit from both AudioSet's large dataset and the task-specific nature of a custom model trained on raw input data. We created several variants of the model in terms of the output classes predicted: a) all 37 labels; b) 29 "fine" labels from which we infer the 8 "coarse" labels; or c) 8 "coarse" labels. We trained two VGG-inspired CNN models (`CNN+VGGish1` and `CNN+VGGish2`) for the 37-way multi-label classification. We also created three hierarchical models (`Hierarchical1`, `Hierarchical2`, and `Hierarchical3`) to attempt to make use of the extra information encapsulated in the known hierarchical nature of the labels.

In addition to experimenting with model variants, we augmented the dataset by adding background noise, pitch shifting, and changing the volume. We also tried several approaches to learning rate decays and warm restarts.

2. RELATED WORK

The general problem of machine listening is discussed in [6]. Much existing work focuses on human speech, but this task focuses on primarily non-speech audio. A large weakly-labeled dataset called AudioSet [5] was created to facilitate research in this domain.

The authors of AudioSet also built an audio classification model called VGGish, based on log-mel spectrogram and CNNs [5]. Similarly, separate work used CNNs for classification of audio events, along with data augmentation to improve training. The work in [7] uses synthetic recordings involving multiple sound sources, where multiple recordings have been combined algorithmically and processed further via frequency band amplification or attenuation.

This task involves hierarchical category labels. The general problem of hierarchical classification is reviewed in [8].

https://doi.org/10.33682/v0ns-1352

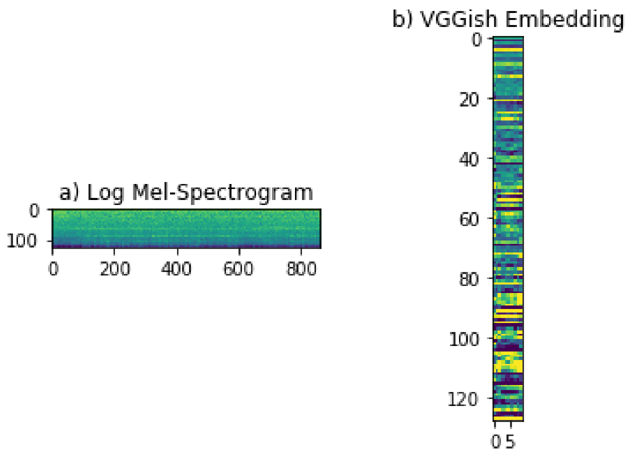


Figure 1: Input features for a sample input file. X-axis is time. a) Log-mel spectrogram: 128 mel bins, 862 time bins. b) VGGish embedding: 128 dimensions, 10 time bins.

3. FEATURE EXTRACTION

3.1. Data augmentation and spectrogram generation

To help the model generalize and to augment the dataset, each file was subjected to pitch shifting, volume changing, and an addition of background noise. After augmentation, each audio file was converted into a log-mel spectrogram with 128 mel bins. The original sample rate of 44.1 kHz was retained, resulting in each spectrogram having 862 time bins. The VGGish features (128x10) from the pre-trained AudioSet model were also generated for each input file. See Figure 1 for a visualization.

3.2. Label choice

To assign labels to each example, we tried several configurations that take into account the disagreement among human annotators. We tried several different thresholds of agreement from 25 percent to 75 percent agreement yielding a positive value. We also tried assigning labels as a float that represents the agreement among annotators. We achieved the best results when we restricted positive labels to only classes that had over 50 percent agreement from people who voted on that particular class.

4. MODELS

To build our model, we began by feeding log-mel spectrogram values into a VGGish architecture, and then modified the architecture parameters based on training results from the Task 5 dataset. The VGGish architecture failed to improve past the first epoch—possibly the model was overfitting due to the large number of layers and the relatively small size of the dataset. By removing some convolutional layers and maxpooling layers, the model would learn more gradually and continue to improve after the first epoch.

In addition to removing layers, we found that altering the kernel sizes improved training. Details of the convolution layers are given in Table 1 and Table 2, where each row describes a ”convolution block” of the following sequence of layers: convolution, maxpool, batch normalization, and dropout. In a given block, some of the layers may not be present, as specified in those tables. The first convolution block has a kernel size of 1x1, which was borrowed from ConvNet configurations, although the 1x1 layers occur in later layers rather than the first in [9]. For `CNN+VGGish1`, `CNN+VGGish2`, both fine- and coarse-level classification models used in `Hierarchical1`, and the coarse-level classification model used in `Hierarchical2` and `Hierarchical3`, the third convolution block features a large and rectangular (16x128) kernel size with a large stride and padding. We modified the configuration for the fine-level classification model used in `Hierarchical2` and `Hierarchical3` to reduce this to a smaller kernel – Table 2 gives the kernel size, stride, and padding used in those models.

Each convolution block contains batch normalization and dropout at a rate of 0.5. One maxpooling layer follows the third convolution block.

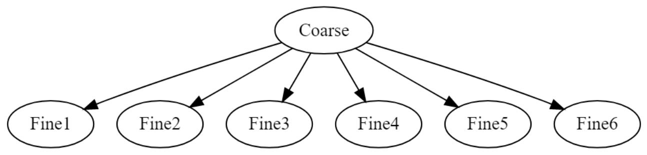


Figure 2: Hierarchical model.

4.1. CNN + VGGish

The results of our CNN model were unable to surpass the baseline results, so we decided to merge the AudioSet-based VGGish embeddings into our trained model at the fully-connected layer level (see Table 3). The output of our CNN model was 256 channels of 1 value (256x1) while the VGGish embedding output was 128x10. These outputs were flattened (to vectors of length 256 and 1280, respectively) and concatenated to yield a 1536-dimensional vector which was followed by three fully-connected layers that reduce the dimensionality to 512, 256, and finally the desired number of output classes. Batch normalization is applied to each fully-connected layer, as is a dropout rate of 0.2. Adding the VGGish embeddings improved our training results and allowed us to surpass the baseline results for some metrics.

4.2. Hierarchical

Because the class labels are given in terms of a known two-level hierarchy, we built an alternative model that takes the label hierarchy into account. Our model is similar to the ”Local classifier per parent node” approach in [8]. A top-level model M_C was built that would predict probabilities for each of the eight “coarse” labels. Two of the “coarse” labels (`non-machinery-impact` and `dog-barking-whining`) only had a single associated “fine” label, so a prediction from the top-level model of one of these two classes was hard-coded to generate the same probability of prediction for the associated fine-label class. To handle fine-label predictions for sound events in the other coarse categories, six individual low-level models $\{M_{F_i} \mid COARSE = i\}, 1 \leq i \leq 6$ were trained to classify the probability for each of the fine labels i, conditioned on knowledge of the coarse class label for a particular example. This resulted in a total of seven models; see Figure 2. Each of these models had essentially the same structure, with the exception of the number of nodes in the output layer.

Each fine-label classifier M_{F_i} was trained in the same way as the coarse classifier M_C (see Section 5). The dataset for each classifier was generated by simply extracting the subset of training data where the coarse label was that expected for the fine-label classifier. E.g., for the engine classifier, the data used for training consisted of solely those examples where the coarse label was identified in the ground truth as `engine`.

We constructed a working classification system from these models as follows. First an unknown input example would be given to the coarse-level classifier M_C. Then, the coarse category with the maximum output value would determine which model M_{F_i} to run to determine the fine label output values. Finally, if any other coarse categories were output with value > 0.5, the corresponding models M_{F_i} would be run as well to generate additional possible fine label classifications.

5. TRAINING TECHNIQUES

For `CNN+VGGish1`, `CNN+VGGish2`, `Hierarchical1`, and the coarse-level classification model of `Hierarchical2` and `Hierarchical3`, we used an Adam optimizer [10] with a learning rate of 0.001. Regarding the fine-level classification model for `Hierarchical2` and `Hierarchical3`, we found that using an RMSProp optimizer [11] with a learning rate of 0.01 performed better. For the loss function, we used binary cross entropy with logits, which combines the sigmoid function with binary cross entropy. The loss function is defined as:

$$l_{n,c} = -w_{n,c}[p_c y_{n,c} \cdot \log f(x_{n,c}) + (1-y_{n,c}) \cdot \log(1-f(x_{n,c}))] \tag{1}$$

where $f(x_{n,c}) \in [0,1]^c$ predicts the presence probabilities of sound categories. c is the class number, n is the index of the sample in the batch, and p is the weight of the positive answer for the class c.

We also experimented with modifying the loss function to give weight to classes based on their representation in the dataset. While fully weighting classes to offset the dataset imbalance decreased the micro AUPRC scores, smoothing the weights—such taking the tenth root of each value—helped under-represented classes perform better and made a slight overall improvement to the micro AUPRC scores (results of these experiments not shown here).

When training the coarse-level models, we implemented a modified form of warm restarts[12]. We monitored the micro AUPRC scores of coarse classes on the validation set, and when coarse micro AUPRC scores had not improved by a specified ”stagnation” threshold, the learning rate was reduced. This process was repeated until a minimum learning-rate threshold was reached. The model then would be reset to the original learning rate and made to cycle through again, with the rate of learning rate *reduction* set to be less severe. We saved a new best model at the end of any epoch that resulted in a new highest micro AUPRC score.

Training of the ”branch” models for fine-grained classification in the three `Hierarchical` models proceeded differently, without warm restarts. For each fine-grained classifier, we reduced the learning rate by multiplying it by a factor γ after a certain number of epochs p (for ”patience”) passed with no improvement to the loss. For the `Hierarchical1` model we set $\gamma = 0.1$ and $p = 5$, while for the `Hierarchical2` and `Hierarchical3` variants we set $\gamma = 0.2$ and $p = 6$. We saved a new best model at the end of any epoch that resulted in a new lowest validation loss.

6. RESULTS

Our results can be found in Table 4. Our method was able to surpass the Micro AUPRC and Macro AUPRC baseline scores in the coarse-level evaluation. However, our method was unable to beat the baseline in fine-level evaluation. Both `CNN+VGGish` models are checkpoints from different points of a single training session; the best fine-level score was achieved before the best coarse-level score.

We adopted the `CNN+VGGish1` model as the coarse-level classifier for the `Hierarchical1` and `Hierarchical2` models. For `Hierarchical3` the coarse-level classifier was the `CNN+VGGish2` model. For all the `Hierarchical` models, the results were worse than those of the best single model trained to jointly output fine and coarse labels (`CNN+VGGish1`), except for one metric: Micro F1 for the fine-level evaluation. Notice that we were able to increase the fine-level Micro F1 score from 0.490 (`Hierarchical1`) to 0.524 (`Hierarchical3`) via the modifications noted above to the optimization parameters. Notably, the latter score of 0.524 was better than the baseline system result of 0.502. However, this Micro F1 optimization also lead to a decrease in the Micro AUPRC and Macro AUPRC scores in the fine-level evaluation.

The inferior results of the three `Hierarchical` models for Micro and Macro AUPRC were a surprise, but they seem to indicate that the single model has more than enough parameters to do both fine and coarse tasks simultaneously. A possible explanation is that the fine-level models M_{F_i} were only trained on a strict subset of the dataset. An improvement might be to use the entire dataset, but to assign a new dummy output label in the ground truth for all examples where the coarse label $\neq i$, in order to provide more negative examples.

7. CONCLUSIONS

Our results show how fusing a custom CNN model with VGGish embeddings can impact scores. Furthermore, creating a hierarchical model has the potential to fine-tune subset classes of individual coarse classes. Further hyper-parameter tuning may yield better results, as may further experimentation with data augmentation. For more details please refer to our GitHub repository at `https://github.com/microsoft/dcase-2019`.

As future work, we plan to perform segmentation on the time-frequency domain to obtain background and foreground segments and adopt classification models on them to further improve the performance.

Conv Block	In Channels	Out Channels	Kernel Size	Stride	Padding	Batch Norm	Max Pooling	Dropout
1	1	8	(1,1)	(1,1)	(0,0)	True	False	.5
2	8	16	(3,3)	(1,1)	(1,1)	True	False	.5
3	16	32	(16,128)	(4,16)	(8,16)	True	(4,4)	.5
4	32	64	(5,5)	(2,2)	(1,1)	True	False	.5
5	64	128	(5,5)	(2,2)	(1,1)	True	False	.5
6	128	256	(3,3)	(2,2)	(1,1)	True	False	.5

Table 1: Convolution blocks of `CNN+VGGish1`, `CNN+VGGish2`, both fine- and coarse-level classification models in `Hierarchical1`, and the coarse-level classification model used in `Hierarchical2` and `Hierarchical3`.

Conv Block	In Channels	Out Channels	Kernel Size	Stride	Padding	Batch Norm	Max Pooling	Dropout
1	1	8	(1,1)	(2,2)	(0,0)	True	False	.5
2	8	16	(3,3)	(2,2)	(1,1)	True	False	.5
3	16	32	(3,3)	(4,4)	(1,1)	True	(4,4)	.5
4	32	64	(5,5)	(2,2)	(1,1)	True	False	.5
5	64	128	(3,3)	(3,3)	(1,1)	True	False	.5
6	128	256	(3,3)	(3,3)	(1,1)	True	False	.5

Table 2: Convolution blocks of the fine-level classification model used in `Hierarchical2` and `Hierarchical3`.

FC-Layer	In Channels	Out Channels	Batch Norm	Dropout
Bilinear	(256,1280)	512	True	.2
Linear	512	256	True	.2
Linear	256	number of classes	False	None

Table 3: Combining VGGish embeddings with spectrogram convolution output in fully-connected layers.

	Micro AUPRC	Micro F1	Macro AUPRC	Micro AUPRC	Micro F1	Macro AUPRC
System	Fine-level evaluation			Coarse-level evaluation		
`Baseline (Fine-level)`	**0.671**	0.502	**0.427**	0.742	0.507	0.530
`Baseline (Coarse-level)`	-	-	-	0.762	**0.674**	0.542
`CNN+VGGish1`	0.646	0.483	0.425	**0.787**	0.609	**0.579**
`CNN+VGGish2`	0.656	0.398	0.401	0.768	0.533	0.555
`Hierarchical1`	0.643	0.490	0.414	**0.787**	0.609	**0.579**
`Hierarchical2`	0.643	0.516	0.412	**0.787**	0.609	**0.579**
`Hierarchical3`	0.623	**0.524**	0.386	0.768	0.533	0.555

Table 4: Results: metrics computed on validation set. Best results for each metric indicated in **bold**.

8. REFERENCES

[1] M. Thorogood, J. Fan, and P. Pasquier, "Soundscape audio signal classification and segmentation using listeners perception of background and foreground sound," *Journal of the Audio Engineering Society*, vol. 64, no. 7/8, pp. 484–492, 2016.

[2] D. Conte, P. Foggia, G. Percannella, A. Saggese, and M. Vento, "An ensemble of rejecting classifiers for anomaly detection of audio events," pp. 76–81, 2012.

[3] A. Farina and P. Salutari, "Applying the ecoacoustic event detection and identification (EEDI) model to the analysis of acoustic complexity," vol. 14, pp. 13–42, 2016.

[4] P. J. Bello, C. Silva, O. Nov, R. L. Dubois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "Sonyc: A system for monitoring, analyzing, and mitigating urban noise pollution," *Communications of the ACM*, vol. 62, no. 2, pp. 68–77, 2019.

[5] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. [Online]. Available: https://arxiv.org/abs/1609.09430

[6] R. F. Lyon, *Human and Machine Hearing: Extracting Meaning from Sound*. Cambridge University Press, 2017.

[7] N. Takahashi, M. Gygli, B. Pfister, and L. V. Gool, "Deep convolutional neural networks and data augmentation for acoustic event detection," *CoRR*, vol. abs/1604.07160, 2016. [Online]. Available: http://arxiv.org/abs/1604.07160

[8] C. N. Silla and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1, pp. 31–72, Jan 2011. [Online]. Available: https://doi.org/10.1007/s10618-010-0175-9

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[10] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[11] T. Tieleman and G. Hinton, "Rmsprop, coursera: Neural networks for machine learning," *Technical report*, 2012.

[12] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

SOUND EVENT DETECTION IN DOMESTIC ENVIRONMENTS WITH WEAKLY LABELED DATA AND SOUNDSCAPE SYNTHESIS

Nicolas Turpault[1], *Romain Serizel*[1], *Ankit Shah*[2], *Justin Salamon*[3]

[1]Université de Lorraine, CNRS, Inria, Loria, F-54000 Nancy, France
[2]Language Technologies Institute, Carnegie Mellon University, Pittsburgh PA, United States
[3]Adobe Research, San Francisco CA, United States

ABSTRACT

This paper presents Task 4 of the Detection and Classification of Acoustic Scenes and Events (DCASE) 2019 challenge and provides a first analysis of the challenge results. The task is a follow-up to Task 4 of DCASE 2018, and involves training systems for large-scale detection of sound events using a combination of weakly labeled data, i.e. training labels without time boundaries, and strongly-labeled synthesized data. We introduce the Domestic Environment Sound Event Detection (DESED) dataset, mixing a part of last year's dataset and an additional synthetic, strongly labeled, dataset provided this year that we describe in more detail. We also report the performance of the submitted systems on the official evaluation (test) and development sets as well as several additional datasets. The best systems from this year outperform last year's winning system by about 10% points in terms of F-measure.

***Index Terms*—** Sound event detection, weakly labeled data, semi-supervised learning, synthetic data

1. INTRODUCTION

Sound conveys important information in our everyday lives – we depend on sounds to better understand changes in our physical environment and to perceive events occurring around us. We perceive the sound scene (the overall soundscape of e.g., an airport or inside a house) as well as individual sound events (e.g., car honks, footsteps, speech, etc.). Sound event detection within an audio recording refers to the task of detecting and classifying sound events, that is, temporally locating the occurrences of sound events in the recording and recognizing which object or category each sound belongs to. Sound event detection has potential applications in noise monitoring in smart cities [1, 2], surveillance [3], urban planning [1], multimedia information retrieval [4, 5]; and domestic applications such as smart homes, health monitoring systems and home security solutions [6, 7, 8] to name a few. In recent years the field has gained increasing interest from the broader machine learning and audio processing research communities.

Sound event detection (SED) systems trained using weak labels have seen significant interest [6, 9, 10, 11, 12] in the research community, as they address some of the challenges involved in developing models that require strongly labeled data for training. In particular, strongly labeled data is time-consuming and difficult to annotate as it requires annotating the temporal extent of event occurrences in addition to their presence or absence. Strong label annotations are also more likely to contain human errors/disagreement given the ambiguity in the perception of some sound event onsets and offsets. In the case of weakly labeled data, we only have information about whether an event is present in a recording or not. We have no information about how many times the event occurs nor the temporal locations of the occurrences within the audio clip. For real-world applications, it is critical to build systems that generalize over a large number of sound classes and a variety of sound event distributions. In such cases, it may be more feasible to collect large quantities of weakly labeled data as opposed to strongly labeled data which is significantly more costly in terms of both time and effort.

We propose to follow up on DCASE 2018 Task 4 [6] and investigate the scenario where large-scale SED systems can exploit the availability of a small set of weakly annotated data, a larger set of unlabeled data and an additional training set of synthetic soundscapes with strong labels. Given these data, the goal of this task is to train SED models that output event detections with time boundaries (i.e. strong predictions) in domestic environments. That is, a system has to detect the presence of a sound event as well as predict the onset and offset times of each occurrence of the event. We generate strongly annotated synthetic soundscapes using the Scaper library [13]. Given a set of user-specified background and foreground sound event recordings, Scaper automatically generates soundscapes containing random mixtures of the provided events sampled from user-defined distributions. These distributions are defined via a sound event specification including properties such as event duration, onset time, signal-to-noise ratio (SNR) with respect to the background and data augmentation (pitch shifting and time stretching). This allows us to generate multiple different soundscape instantiations from the same specification, which is chosen based on our general requirements for the soundscapes. Since generating such strongly labeled synthetic data is feasible on a large scale, we provide a strongly labeled synthetic dataset in order to explore whether it can help improve SED models. We believe insights learned from this task will be beneficial to the community as such an exploration is novel and will provide a pathway to developing scalable SED systems.

The remainder of this manuscript is organized as follows: Section 2 provides a brief overview of the task definition and how the development and evaluation datasets were created. Section 3 describes the baseline system and the evaluation procedure for DCASE 2019 Task 4. Section 4 gives an overview of the systems submitted to the challenge for this task. Finally, conclusions from the challenge are provided in section 5.

This work was made with the support of the French National Research Agency, in the framework of the project LEAUDS Learning to understand audio scenes (ANR-18-CE23-0020) and the French region Grand-Est. Experiments presented in this paper were carried out using the Grid5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000).

https://doi.org/10.33682/006b-jx26

Class	Unique events	Dev set Clips	Dev set Events
Alarm/bell/ringing	190	392	755
Blender	98	436	540
Cat	88	274	547
Dishes	109	444	814
Dog	136	319	516
Electric shaver/toothbrush	56	221	230
Frying	64	130	137
Running water	68	143	157
Speech	128	1272	2132
Vacuum cleaner	74	196	204
Total	1011	2045	6032

Table 1: Class-wise statistics for the synthetic development subset.

Class	Unique events	Synth set 1 Clips	Synth set 1 Events
Alarm/bell/ringing	63	101	184
Blender	27	84	95
Cat	26	113	197
Dishes	34	161	293
Dog	43	124	217
Electric shaver/toothbrush	17	113	117
Frying	17	52	52
Running water	20	67	73
Speech	47	471	803
Vacuum cleaner	20	92	93
Total	314	1378	2124

Table 2: Class-wise statistics for the synthetic evaluation subsets

2. TASK DESCRIPTION AND THE DESED DATASET

2.1. Task description

This task is the follow-up to DCASE 2018 Task 4 [6] and focuses on the same 10 classes of sound events. Systems are expected to produce strongly labeled output (i.e. detect sound events with a start time, end time, and sound class label), but are provided with weakly labeled data (i.e. sound recordings with only the presence/absence of a sound included in the labels without any timing information) for training. Multiple events can be present in each audio recording, including overlapping events. As in the previous iteration of this task, the challenge entails exploiting a large amount of unbalanced and unlabeled training data together with a small weakly annotated training set to improve system performance. However, unlike last year, in this iteration of the challenge we also provide an additional training set with strongly annotated synthetic soundscapes. This opens the door to exploring scientific questions around the informativeness of real (but weakly labeled) data versus strongly-labeled synthetic data, whether the two data sources are complementary or not, and how to best leverage these datasets to optimize system performance.

2.2. DESED development dataset

The DESED development dataset is composed of 10-sec audio clips recorded in a domestic environment or synthesized to simulate a domestic environment. The real recordings are taken from AudioSet [14]. The development dataset is divided in two subsets: (i) A training subset composed of real recordings similar to DCASE 2018 task 4 [10] and synthetic soundscapes generated using Scaper (see also Table 1). (ii) A validation subset composed of real recordings with strongly labeled data which is the combination of the validation and evaluation sets from DCASE 2018 Task 4.

2.2.1. Synthetic soundscape generation procedure

The subset of synthetic soundscapes is comprised of 10 second audio clips generated with Scaper [13], a python library for soundscape synthesis and augmentation. Scaper operates by taking a set of foreground sounds and a set of background sounds and automatically sequencing them into random soundscapes sampled from a user-specified distribution controlling the number and type of sound events, their duration, signal-to-noise ratio, and several other key characteristics. The foreground events are obtained from the Freesound Dataset (FSD) [15, 16]. Each sound event clip was verified by a human to ensure that the sound quality and the event-to-background ratio were sufficient to be used as an isolated sound event. We also controlled for whether the sound event onset and offset were present in the clip. Each selected clip was then segmented when needed to remove silences before and after the sound event and between sound events when the file contained multiple occurrences of the sound event class. The number of unique isolated sound events per class used to generate the subset of synthetic soundscapes is presented in Table 1. We also list the number of clips containing each sound class and the number of events per class.

The background textures are obtained from the SINS dataset (activity class “other”) [17]. This particular activity class was selected because it contains a low amount of sound events from our 10 target foreground sound event classes. However, there is no guarantee that these sound event classes are completely absent from the background clips. A total of 2060 unique background clips are used to generate the synthetic subset.

Scaper scripts are designed such that the distribution of sound events per class, the number of sound events per clip (depending on the class) and the sound event class co-occurrence are similar to that of the validation set which is composed of real recordings. The synthetic soundscapes are annotated with strong labels that are automatically generated by Scaper [13].

2.3. DESED evaluation dataset

The evaluation set is composed of two subsets: a subset with real recordings and a subset with synthetic soundscapes.

2.3.1. Real recordings

The first subset contains 1,013 audio clips and is used for ranking purposes. It is comprised of audio clips extracted from 692 YouTube and 321 Vimeo videos under creative common licenses. Each clip is annotated by a human and annotations are verified by a second annotator.

2.3.2. Synthetic soundscapes

The second subset is comprised of synthetic soundscapes generated with Scaper[1]. This subset is used for analysis purposes and its de-

[1]The JAMS [18] annotation files corresponding to these soundscapes can be accessed from DCASE website: `http://dcase.community/`.

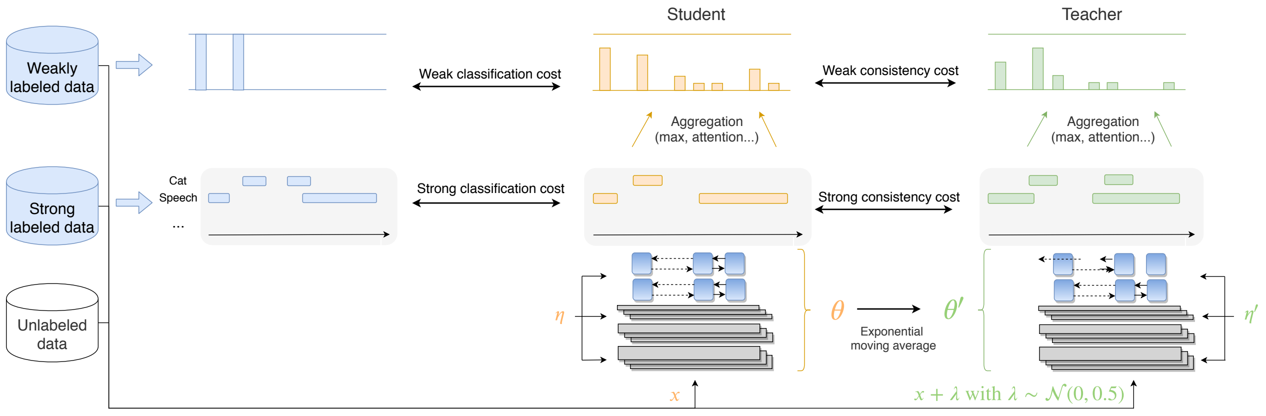


Figure 1: Mean-teacher model. η and η' represent noise applied to the different models (in this case dropout).

sign is motivated by the analysis of last year's results [10]. In particular, most submissions from last year performed poorly in terms of event segmentation. One of the goals of this subset is to facilitate studies on the extent to which including strongly labeled data in the training set helps improve and refine the segmentation output. The foreground events are obtained from the FSD [15, 16]. The selection process was the same as described for the development dataset. Background sounds are extracted from YouTube videos under a Creative Common license and from the Freesound subset of the MUSAN dataset [19]. The synthetic subset is further divided into several subsets (described below) for a total of 12,139 audio clips synthesized from 314 isolated events. The isolated sound event distribution per class is presented in Table 2.

Varying foreground-to-background SNR: A subset (denoted Synthetic set 1) of 754 soundscapes is generated with a sound event distribution similar to that of the training set. Four versions of this subset are generated varying the value of the foreground events' SNR with respect to the background: 0 dB, 6 dB, 15 dB and 30 dB.

Audio degradation: Six alternative versions of the previous subset (with SNR=0 dB) are generated introducing artificial degradation with the Audio Degradation Toolbox [20]. The following degradations are used (with default parameters) : "smartPhonePlayback", "smartPhoneRecording", "unit_applyClippingAlternative", "unit_applyLowpassFilter", "unit_applyHighpassFilter" and "unit_applyDynamicRangeCompression".

Varying onset time: A subset of 750 soundscapes is generated with uniform sound event onset distribution and only one event per soundscape. The SNR parameter is set to 0 dB. Three variants of this subset are generated with the same isolated events, only shifted in time. In the first version, all sound events have an onset located between 250 ms and 750 ms, in the second version the sound event onsets are located between 4.75 s and 5.25 s and in the last version the sound event onsets are located between 9.25 s and 9.75 s.

Long sound events vs. short sound events: A subset with 522 soundscapes is generated where the background is selected from one of the five long sound event classes (Blender, Electric shaver/toothbrush, Frying, Running water and Vacuum cleaner). The foreground sound events are selected from the five short sound event classes (Alarm/bell/ringing, Cat, Dishes, Dog and Speech). Three variants of this subset are generated with similar sound event scripts and varying values of the sound event SNR parameter (0 dB, 15 dB and 30 dB).

3. BASELINE

The baseline system[2] is inspired by the winning system from DCASE 2018 Task 4 by Lu [21]. It uses a mean-teacher model which is a combination of two models: a student model and a teacher model (both have the same architecture). Our implementation of the mean-teacher model is based on the work of Tarvainen and Valpola [22]. The student model is the final model used at inference time, while the teacher model is aimed at helping the student model during training and its weights are an exponential moving average of the student model's weights. A depiction of the baseline model is provided in Figure 1.

The models are a combination of a convolutional neural network (CNN) and a recurrent neural network (RNN) followed by an aggregation layer (in our case an attention layer). The output of the RNN gives strong predictions (the weights of this model are denoted θ_s) while the output of the aggregation layer gives the weak predictions (the weights of this model are denoted θ).

The student model is trained on the synthetic and weakly labeled data. The loss (binary cross-entropy) is computed at the frame level for the strongly labeled synthetic data and at the clip level for the weakly labeled data. The teacher model is not trained, rather, its weights are a moving average of the student model (at each epoch). During training, the teacher model receives the same input as the student model but with added Gaussian noise, and helps train the student model via a consistency loss (mean-squared error) for both strong (frame-level) and weak predictions. Every batch contains a combination of unlabeled, weakly and strongly labeled samples.

This results in four loss components: two for classification (weak and strong) and two for consistency (weak and strong), which are combined as follows:

$$\begin{aligned} L(\theta) =& L_{class_w}(\theta) + \sigma(\lambda) L_{cons_w}(\theta) \\ &+ L_{class_s}(\theta_s) + \sigma(\lambda) L_{cons_s}(\theta_s) \end{aligned} \tag{1}$$

4. SUBMISSION EVALUATION

DCASE 2019 Task 4 obtained 57 submissions from 18 different teams involving 60 researchers overall.

[2]Open source code available at: `https://github.com/turpaultn/DCASE2019_task4/tree/public/baseline`

Rank	System	Classifier	Real recordings					Synthetic
			Event-based				Segment-based	Event-based
			Eval	Youtube	Vimeo	Valid	Eval	Set 1
1	**Lin, ICT**	CNN	**42.7%**	47.7%	29.4%	45.3%	64.8%	47.6%
2	**Delphin-Poulat, OL**	CRNN	**42.1%**	45.8%	33.3%	43.6%	71.4%	59.8%
3	**Shi, FRDC**	CRNN	**42.0%**	46.1%	31.5%	42.5%	69.8%	53.2%
4	**Cances, IRIT**	CRNN	**39.7%**	43.0%	30.9%	39.9%	64.7%	50.8%
5	**Yan, USTC**	CRNN	**36.2%**	38.8%	28.7%	42.6%	65.2%	41.8%
6	**Lim, ETRI**	CRNN, Ensemble	**34.4%**	38.6%	23.7%	40.9%	66.4%	42.5%
7	**Kiyokawa, NEC**	ResNet, SENet	**32.4%**	36.2%	23.8%	36.1%	65.3%	42.3%
8	**Chan, NU**	NMF, CNN	**31.0%**	34.7%	21.6%	30.4%	58.2%	46.7%
9	**Zhang, UESTC**	CNN,ResNet,RNN	**30.8%**	34.5%	21.1%	35.6%	60.9%	49.2%
10	**Kothinti, JHU**	CRNN, RBM, CRBM, PCA	**30.7%**	33.2%	23.8%	34.6%	53.1%	35.6%
11	**Wang B., NWPU**	CNN, RNN, ensemble	**27.8%**	30.1%	21.7%	31.9%	61.6%	32.9%
12	**Lee, KNU**	CNN	**26.7%**	28.1%	22.9%	31.6%	50.2%	33.0%
	Baseline 2019	CRNN	**25.8%**	29.0%	18.1%	23.7%	53.7%	40.6%
13	**Agnone, PDL**	CRNN	**25.0%**	27.1%	20.0%	59.6%	60.4%	46.7%
14	**Rakowski, SRPOL**	CNN	**24.2%**	26.2%	19.2%	24.3%	63.4%	29.7%
15	**Kong, SURREY**	CNN	**22.3%**	24.1%	17.0%	21.3%	59.4%	23.6%
16	**Mishima, NEC**	ResNet	**19.8%**	21.8%	15.0%	24.7%	58.7%	33.0%
17	**Wang D., NUDT**	CRNN	**17.5%**	19.2%	13.3%	22.4%	63.0%	14.0%
18	**Yang, YSU**	CMRANN-MT	**6.7%**	7.6%	4.6%	19.4%	26.3%	7.5%

Table 3: F1-score performance on the evaluation sets

4.1. Evaluation metrics

Submissions were evaluated according to an event-based F1-score with a 200 ms collar on the onsets and a collar on the offsets that is the greater of 200 ms and 20% of the sound event's length. The overall F1-score is the unweighted average of the class-wise F1-scores (macro-average). In addition, we provide the segment-based F1-score on 1 s segments as a secondary measure. The metrics are computed using the sed_eval library [23].

4.2. System performance

The official team ranking (best system from each team) along with some characteristics of the submitted systems is presented in Table 3. Submissions are ranked according to the event-based F1-score computed over the real recordings in the evaluation set. For a more detailed comparison, we also provide the event-based F1-score on the YouTube and Vimeo subsets and the segment-based F1-score over all real recordings. The event-based F1-score on the validation set is reported for the sake of comparison with last year's results (75% of the 2019 validation set is comprised of the 2018 evaluation set). Performance on synthetic recordings is not taken into account in the ranking, but the event-based F1-score on Synthetic set 1 (0 dB) is presented here as well. The baseline for DCASE 2018 would obtain 22.2% F1-score on the evaluation set.

Twelve teams outperform the baseline with the best systems [24, 25, 26] outperforming the baseline by 16% points and the best system from 2018 by over 10 % points. While the ranking on the YouTube subset is similar to the official ranking, there rankings based on the Vimeo and synthetic subsets are notably different. Performance on the Vimeo set is in general considerably lower than on the YouTube set and Synthethic set 1. The fact that no data from Vimeo was used during training (unlike data from YouTube and synthetic data) suggests that the submitted systems struggle to generalize to an entirely unseen set of recording conditions.

All three top-performing teams used a semi-supervised mean-teacher model [22]. Lin et al. [24] focused on the importance of semi-supervised learning with a guided learning setup [27] and on how synthetic data can help when used together with a sufficient amount of real data. Delphin-Poulat et al. [25] focused on data augmentation and Shi [26] focused on a specific type of data augmentation where both audio files and their labels are mixed. Cances et al. [28] proposed a multi-task learning setup where audio tagging (producing weak predictions) and the sound event localization in time (strong predictions) are treated as two separate subtasks [29]. The latter was also the least complex of the top-performing systems.

Most of the top-performing systems also demonstrate the importance of employing class-dependent post-processing [24, 25, 28], which improves performance significantly compared to e.g. using a fixed median filtering approach. This highlights the benefits of applying dedicated segmentation post-processing [28, 30].

5. CONCLUSION

This paper presents DCASE 2019 Task 4 and the DESED dataset, which focus on SED in domestic environments. The goal of the task is to exploit a small dataset of weakly labeled sound clips together with a larger unlabeled dataset to perform SED. An additional training dataset composed of synthetic soundscapes with strong labels is provided to explore the gains achievable with simulated data. The best submissions from this year outperform last year's winning submission by over 10 % points, representing a notable advancement. Evaluation on the Vimeo subset, suggests there is still a significant challenge in generalizing to unseen recording conditions.

6. ACKNOWLEDGMENT

The authors would like to thank the Hamid Eghbal-Zadeh from Johannes Kepler University (Austria) who participated to the initial discussions about this task as well as all participants to the task.

7. REFERENCES

[1] J. P. Bello, C. Silva, O. Nov, R. L. DuBois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "SONYC: A system for the monitoring, analysis and mitigation of urban noise pollution," *Communications of the ACM*, In press, 2018.

[2] J. P. Bello, C. Mydlarz, and J. Salamon, "Sound analysis in smart cities," in *Computational Analysis of Sound Scenes and Events*. Springer, 2018, pp. 373–397.

[3] R. Radhakrishnan, A. Divakaran, and A. Smaragdis, "Audio analysis for surveillance applications," in *Proc. WASPAA*. IEEE, 2005, pp. 158–161.

[4] E. Wold, T. Blum, D. Keislar, and J. Wheaten, "Content-based classification, search, and retrieval of audio," *IEEE multimedia*, vol. 3, no. 3, pp. 27–36, 1996.

[5] Q. Jin, P. Schulam, S. Rawat, S. Burger, D. Ding, and F. Metze, "Event-based video retrieval using audio," in *Proc. Interspeech*, 2012.

[6] R. Serizel, N. Turpault, H. Eghbal-Zadeh, and A. Parag Shah, "Large-Scale Weakly Labeled Semi-Supervised Sound Event Detection in Domestic Environments," July 2018, proc. DCASE Workshop.

[7] C. Debes, A. Merentitis, S. Sukhanov, M. Niessen, N. Frangiadakis, and A. Bauer, "Monitoring activities of daily living in smart homes: Understanding human behavior," *IEEE Signal Processing Magazine*, vol. 33, no. 2, pp. 81–94, 2016.

[8] Y. Zigel, D. Litvak, and I. Gannot, "A method for automatic fall detection of elderly people using floor vibrations and soundproof of concept on human mimicking doll falls," *IEEE Transactions on Biomedical Engineering*, vol. 56, no. 12, pp. 2858–2867, 2009.

[9] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, "Dcase 2017 challenge setup: Tasks, datasets and baseline system," in *Proc. DCASE Workshop*, 2017.

[10] R. Serizel and N. Turpault, "Sound Event Detection from Partially Annotated Data: Trends and Challenges," in *Proc. IcETRAN conference*, June 2019.

[11] A. Shah, A. Kumar, A. G. Hauptmann, and B. Raj, "A closer look at weak label learning for audio events," *arXiv preprint arXiv:1804.09288*, 2018.

[12] B. McFee, J. Salamon, and J. P. Bello, "Adaptive pooling operators for weakly labeled sound event detection," *IEEE/ACM Trans. on Audio, Speech, and Language Processing*, vol. 26, no. 11, pp. 2180–2193, Nov. 2018.

[13] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *Proc. WASPAA*. IEEE, 2017, pp. 344–348.

[14] J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. ICASSP*, 2017.

[15] F. Font, G. Roma, and X. Serra, "Freesound technical demo," in *Proc. ACMM*. ACM, 2013, pp. 411–412.

[16] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound datasets: a platform for the creation of open audio datasets," in *Proc. ISMIR*, Suzhou, China, 2017, pp. 486–493.

[17] G. Dekkers, S. Lauwereins, B. Thoen, M. W. Adhana, H. Brouckxon, T. van Waterschoot, B. Vanrumste, M. Verhelst, and P. Karsmakers, "The SINS database for detection of daily activities in a home environment using an acoustic sensor network," in *Proc. DCASE Workshop*, November 2017, pp. 32–36.

[18] E. J. Humphrey, J. Salamon, O. Nieto, J. Forsyth, R. Bittner, and J. P. Bello, "JAMS: A JSON annotated music specification for reproducible MIR research," in *15th Int. Soc. for Music Info. Retrieval Conf.*, Taipei, Taiwan, Oct. 2014, pp. 591–596.

[19] D. Snyder, G. Chen, and D. Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015, arXiv:1510.08484v1.

[20] M. Mauch and S. Ewert, "The audio degradation toolbox and its application to robustness evaluation," in *Proc. ISMIR*, 2013, pp. 83–88.

[21] L. JiaKai, "Mean teacher convolution system for dcase 2018 task 4," DCASE2018 Challenge, Tech. Rep., September 2018.

[22] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proc. NeurIPS*, 2017, p. 10.

[23] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, May 2016.

[24] L. Lin and X. Wang, "Guided learning convolution system for dcase 2019 task 4," Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, Tech. Rep., June 2019.

[25] L. Delphin-Poulat and C. Plapous, "Mean teacher with data augmentation for dcase 2019 task 4," Orange Labs Lannion, France, Tech. Rep., June 2019.

[26] Z. Shi, "Hodgepodge: Sound event detection based on ensemble of semi-supervised learning methods," Fujitsu Research and Development Center, Beijing, China, Tech. Rep., June 2019.

[27] L. Lin, X. Wang, H. Liu, and Y. Qian, "What you need is a more professional teacher," *arXiv preprint arXiv:1906.02517*, 2019.

[28] L. Cances, T. Pellegrini, and P. Guyot, "Multi task learning and post processing optimization for sound event detection," IRIT, Universit de Toulouse, CNRS, Toulouse, France, Tech. Rep., June 2019.

[29] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[30] S. Kothinti, G. Sell, S. Watanabe, and M. Elhilali, "Integrated bottom-up and top-down inference for sound event detection," Department of Electrical and Computer Engineering, Johns Hopkins University, Baltimore, USA, Tech. Rep., June 2019.

OPEN-SET ACOUSTIC SCENE CLASSIFICATION WITH DEEP CONVOLUTIONAL AUTOENCODERS

Kevin Wilkinghoff, Frank Kurth

Fraunhofer Institute for Communication, Information Processing and Ergonomics FKIE
Fraunhoferstraße 20, 53343 Wachtberg, Germany
kevin.wilkinghoff@fkie.fraunhofer.de, frank.kurth@fkie.fraunhofer.de

ABSTRACT

Acoustic scene classification is the task of determining the environment in which a given audio file has been recorded. If it is a priori not known whether all possible environments that may be encountered during test time are also known when training the system, the task is referred to as open-set classification. This paper contains a description of an open-set acoustic scene classification system submitted to task 1C of the Detection and Classification of Acoustic Scenes and Events (DCASE) Challenge 2019. Our system consists of a combination of convolutional neural networks for closed-set identification and deep convolutional autoencoders for outlier detection. On the evaluation dataset of the challenge, our proposed system significantly outperforms the baseline system and improves the score from 0.476 to 0.621. Moreover, our submitted system ranked 3rd among all teams in task 1C.

***Index Terms*—** acoustic scene classification, deep convolutional autoencoder, open-set classification, outlier detection

1. INTRODUCTION

Acoustic scene classification is a subfield of machine listening, where systems need to determine the environment in which given audio files were recorded, and has always been an integral part of the DCASE challenge [1, 2]. Additionally, there is growing interest in open-set classification [3, 4] within the machine learning community since realistic scenarios and applications are almost always open-set problems. The reason is that one can only very rarely capture the entire space of classes when training a classification system. The only potential exception is a very artificial setup that ensures no encounters of data belonging to novel or unknown classes when running the system after training. But since change and evolution in general are inevitable this setup seems very unlikely, especially in real world applications. However, open-set classification is much more difficult than closed-set classification because one also needs to determine whether data belongs to one of the known classes or not (outlier detection [5]), which is an a priori assumption in closed-set classification. This difficulty is probably the reason why most research has been focused on closed-set classification.

To promote this research direction, in this year's edition of the DCASE challenge there is a subtask of the acoustic scene classification task entirely focusing on the open-set setting (task 1C) [6], which will also be the focus of this paper. The development dataset consists of 44 hours of 48kHz audio belonging to some unknown and ten known classes, namely airports, indoor shopping malls, metro stations, pedestrian streets, public squares, streets with medium level of traffic, traveling by a tram, traveling by a bus, traveling by an underground metro and urban parks. The evaluation dataset consists of 20 hours of audio. For all recordings the same recording device has been used (unlike to subtask 1B where four different devices have been used) and all have a length of 10 seconds. To evaluate the performance of the systems, the final score is computed as the weighted average accuracy of the known classes and unknown classes. For more information about the task, see [6].

To our best knowledge, previous work for open-set acoustic scene classification is extremely limited. Still, there are some papers entirely focusing on that task as for example [7] where the authors used one-class support vector machines for open-set classification. Another way to detect outliers and thus make open-set classification possible is to use deep convolutional autoencoders (DCAEs) [8, 9]. By training DCAEs with data belonging to the known classes, one can expect that the neural networks learn to reconstruct this data well but have difficulties when encountering data belonging to unknown classes. In turn, the reconstruction loss can be used as a heuristic to detect outliers.

The contributions of this work are the following. First and foremost, a system for open-set acoustic scene classification is presented[1]. More specifically, we propose to use CNNs for closed-set classification and DCAEs for rejecting unknown acoustic scenes via outlier detection. As a last contribution, an effective way to combine a closed-set classification system and outlier detection models into a single open-set system is presented. It is also worth mentioning, that we did not use any external data resources nor pretrained models for training our system. Although this makes the open-set classification task even more challenging, it also enables us to precisely compare the performance of our system with other submitted systems that did not use external data resources.

2. ACOUSTIC SCENE CLASSIFICATION SYSTEM

As already stated, this paper focuses on open-set acoustic scene classification. But in order to do open-set classification one also needs a well working closed-set classification chain. The reason is that the system needs to 1) determine whether given data belongs to one of the known classes (outlier detection) and if so, 2) predict the most likely of the known classes (closed-set classification). Mathematically, this corresponds to estimating

$$\begin{aligned} &P(Y = y_i, K = \text{true}|X = x) \\ =&P(Y = y_i|K = \text{true}, X = x)P(K = \text{true}|X = x) \end{aligned} \tag{1}$$

where X and Y are random variables denoting the data and one of the known class labels, respectively, and K is a binary random

[1]An open-source Python implementation of the presented system is available here: https://github.com/wilkinghoff/dcase2019

https://doi.org/10.33682/340j-wd27

variable indicating that the data belongs to one of the known classes (see [10]). Thus, open-set classification (left hand side) can indeed be decomposed into the subtasks closed-set classification and outlier detection (right hand side).

We will now present our feature extraction procedure followed by descriptions of the closed-set classification and outlier detection systems. This section is then concluded by a description of how to combine both systems into a single open-set acoustic scene classification system.

2.1. Feature extraction

Almost all recently proposed acoustic scene classification systems as well as the baseline system utilize log-mel spectrograms as input features (see e.g. [2, 11, 12]). As this is the state-of-the-art, we also used log-mel spectrograms and closely followed [13] for the parameter settings with a few changes. More precisely, we also used a Hanning window size of 1024, a hop size of 500 and 64 mel bins but used the cutoff frequencies 50Hz and 16000Hz. Additionally, we normalized the audio files with respect to the maximum norm before extracting the features. The resulting features are of dimension 64×442.

Furthermore, we utilized median filtering for Harmonic-Percussive Source Separation [14] via Librosa [15] as many participants have done in past editions of the DCASE challenge (see e.g. [11, 16]). All mel-spectrograms were separated into harmonic and percussive parts before applying the logarithm resulting in a total number of three features per audio file: The log-mel spectrograms themselves and their harmonic and percussive parts.

Before inserting the features into a neural network, we standardized them in two different ways. For closed-set classification, we subtracted the mean and divided by the standard deviation of all training data, which belongs to any of the ten known classes. When detecting outliers, all features were standardized in the same way but only data belonging to a single known class was used to compute the mean and standard deviation. As we will train individual DCAEs for each class, the data is standardized with respect to that specific class beforehand.

2.2. Closed-set classification

In the era of deep learning, CNNs are the method of choice to classify log-mel spectrograms. Classifying acoustic scenes is not an exception. The CNN proposed in [13] is reported to perform better than the baseline system of the challenge. Thus, we used this CNN as a starting point but changed a few details leading to an even better performance while using less parameters. All CNNs have been implemented using Keras [17] with Tensorflow [18] and their structure can be found in Table 1. For each of the three features, namely log-mel spectrograms and their harmonic and percussive parts, another CNN is trained for 6000 epochs with a batch size of 32 by minimizing the categorical crossentropy. Mixup [19] and Cutout [20] have been used to augment the training data, which are known to be effective in terms of improving classification accuracy (see [12]). Additionally, random shifts in time up to 60% of the entire duration and up to 3 mel bins were used when augmenting data. To acquire a single score per class, the geometric mean of the output distributions obtained with the three CNNs is taken. But since the classification accuracy obtained with the log-mel spectrograms were higher on the validation set, their corresponding scores have been multiplied with a factor of two to give them more weight than the scores resulting from the other two features. Using this heuristic enabled us to use the entire development set, i.e. training and validation split, for training the CNNs and led to better performance because more data results in more knowledge. Note that one can usually achieve better results when carefully tuning model-specific weights but this would have required additional labeled data to obtain meaningful scores for training these weights.

Table 1: CNN architecture for closed-set classification.

Layer	Output Shape	#Parameters
Input	(64, 442)	0
Convolution (kernel size: 3x3)	(64, 442, 64)	640
Batch Normalization	(64, 442, 64)	256
Non-linearity (ReLU)	(64, 442, 64)	0
Convolution (kernel size: 3x3)	(64, 442, 64)	36,928
Batch Normalization	(64, 442, 64)	256
Non-linearity (ReLU)	(64, 442, 64)	0
Average-Pooling (pool size: 2x3)	(32, 147, 64)	0
Convolution (kernel size: 3x3)	(32, 147, 128)	73,856
Batch Normalization	(32, 147, 128)	512
Non-linearity (ReLU)	(32, 147, 128)	0
Convolution (kernel size: 3x3)	(32, 147, 128)	147,584
Batch Normalization	(32, 147, 128)	512
Non-linearity (ReLU)	(32, 147, 128)	0
Average-Pooling (pool size: 2x3)	(16, 49, 128)	0
Convolution (kernel size: 3x3)	(16, 49, 196)	225,988
Batch Normalization	(16, 49, 196)	784
Non-linearity (ReLU)	(16, 49, 196)	0
Convolution (kernel size: 3x3)	(16, 49, 196)	345,940
Batch Normalization	(16, 49, 196)	784
Non-linearity (ReLU)	(16, 49, 196)	0
Average-Pooling (pool size: 2x3)	(8, 16, 196)	0
Convolution (kernel size: 3x3)	(8, 16, 256)	451,840
Batch Normalization	(8, 16, 256)	1,024
Non-linearity (ReLU)	(8, 16, 256)	0
Convolution (kernel size: 3x3)	(8, 16, 256)	590,080
Batch Normalization	(8, 16, 256)	1,024
Non-linearity (ReLU)	(8, 16, 256)	0
Global-Average-Pooling	256	0
Dense (Softmax)	10	2,570
		$\sum$1,880,578

2.3. Outlier detection

To detect outliers, we used one-class classification models, more concretely DCAEs. For each of the known classes, another DCAE was trained using only training data belonging to that particular class. By doing so, we avoided the direct usage of training data belonging to any unknown class. The reason for doing this is that the variability of the unknown class space cannot be captured sufficiently by using samples of unknown classes. However, in order for the outlier detection models to learn to distinguish between strong outliers and weak outliers, which are noisy samples still belonging to the known class a DCAE is trained for, samples belonging to unknown classes are still needed. A way to use these samples for training will be explained later in subsection 2.4.

The particular structure we have chosen for the DCAEs can be found in Table 2. The basic task is to reduce the feature space dimension from 64×442 to 16×110 and reconstruct the input features as accurately as possible. More precisely, we trained another DCAE for each of the ten known classes resulting in a total of ten models per feature type. Again, we implemented the DCAEs with Keras [17] and Tensorflow [18]. To train the DCAEs, we minimized the mean squared error for 1000 epochs using a batch size of 32. In contrast to the CNNs, no data augmentation techniques were applied while training, which is the reason why less epochs are sufficient. We still trained different models for all three features but this time only the training data split of the development set has

Table 2: DCAE architecture for outlier detection.

Layer	Output Shape	#Parameters
Input	(64, 442, 1)	0
Convolution (kernel size: 3x3)	(64, 442, 64)	640
Batch Normalization	(64, 442, 64)	256
Non-linearity (ReLU)	(64, 442, 64)	0
Convolution (kernel size: 3x3)	(64, 442, 64)	36,928
Batch Normalization	(64, 442, 64)	256
Non-linearity (ReLU)	(64, 442, 64)	0
Average-Pooling (pool size: 2x2)	(32, 221, 64)	0
Convolution (kernel size: 3x3)	(32, 221, 128)	73,856
Batch Normalization	(32, 221, 128)	512
Non-linearity (ReLU)	(32, 221, 128)	0
Convolution (kernel size: 3x3)	(32, 221, 128)	147,584
Batch Normalization	(32, 221, 128)	512
Non-linearity (ReLU)	(32, 221, 128)	0
Average-Pooling (pool size: 2x2)	(16, 110, 128)	0
Convolution (kernel size: 3x3)	(16, 110, 128)	147,584
Batch Normalization	(16, 110, 128)	512
Non-linearity (ReLU)	(16, 110, 128)	0
Convolution (kernel size: 3x3)	(16, 110, 128)	147,584
Batch Normalization	(16, 110, 128)	512
Non-linearity (ReLU)	(16, 110, 128)	0
Up-Sampling (size: 2x2)	(32, 220, 128)	0
Zero-Padding	(32, 221, 128)	0
Convolution (kernel size: 3x3)	(32, 221, 64)	73,792
Batch Normalization	(32, 221, 64)	256
Non-linearity (ReLU)	(32, 221, 64)	0
Convolution (kernel size: 3x3)	(32, 221, 64)	36,928
Batch Normalization	(32, 221, 64)	256
Non-linearity (ReLU)	(32, 221, 64)	0
Up-Sampling (size: 2x2)	(64, 442, 64)	0
Convolution (kernel size: 3x3)	(64, 442, 1)	577
Non-linearity (ReLU)	(64, 442, 1)	0
		$\sum$668,545

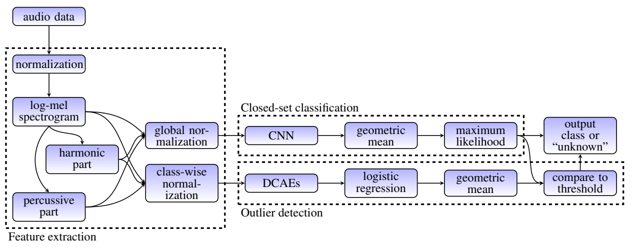


Figure 1: Structure of our proposed open-set acoustic scene classification system.

been used because the validation data set is needed in the next step. Note that using ReLU as an activation function in the last layer prevents the DCAE to perfectly reconstruct the data again since negative output values cannot be produced. But interesting events that are typical for an acoustic scene correspond to high energy in a mel-spectrogram and thus are still positive after normalization. An example are bird calls, which can only very rarely be heard in a metro but are one of the acoustic events one expects to hear in a park. Therefore, a DCAE trained on data belonging to the class "park" should be able to reconstruct bird calls but a DCAE associated with the class "park" should not, leading to a high reconstruction loss when encountering audio containing birds. In conclusion, the usage of ReLU can be seen as a form of regularization in this case and helped to improve the performance when detecting outliers.

2.4. Combined system

Since both subproblems, closed-set classification and outlier detection, have been tackled in some way, we can now determine the final output of the system. The only problem left is that, while the softmax output of the CNNs can be interpreted as a probability distribution, the loss of the DCAEs is just the mean squared error, which is not even bounded and scaled differently for each DCAE. Moreover, there is not only a single loss value per file but ten. Hence, it is highly non-trivial to find a suitable decision criterion when trying to detect outliers.

To solve this issue, we used logistic regression as implemented in Scikit-learn [21]. The idea is to treat the ten losses as ten dimensional features and train a binary classifier with them. For this purpose, we also made use of all audio files belonging to unknown classes. Although it is not a good idea to use these files or their spectral features directly for training a binary classifier, their losses should look much more close to each other (equally bad) than the outliers themselves. Hence, it may be a valid assumption to use them as valuable training data. In addition to that, the logistic regression model is very simple compared to all neural networks involved before. Thus, there is less room for the model to learn more than differentiating between losses corresponding to known classes and the "strange looking ones" belonging to unknown classes. In order to obtain meaningful positive examples of loss values belonging to known classes, we used the validation split of the development set. This is the only reason why the data files have not been used for training the DCAEs before.

To decide whether given data should be treated as an outlier, we used a threshold of 0.5 for all probabilities resulting from the logistic regression model. This means that for each encountered audio file, the class belonging to the maximum likelihood is chosen but if the score is smaller than 0.5, it is labeled as "unknown" instead. Choosing this particular threshold makes sense because the logistic regression model has been trained with balanced class weights to compensate for the different number of known and unknown training samples. In addition to that, we also labeled all audio files that had a maximum likelihood score less than 0.5 in the closed-set classification evaluation as "unknown". The underlying assumption is that most resulting scores are very high anyway and thus very small scores indicate that the model has difficulties in deciding which class the encountered data belongs to. This may indicate data belonging to unknown classes. See Figure 1 for an abstract overview of the entire system.

3. EXPERIMENTAL RESULTS

3.1. Closed-set classification

Closed-set classification is not the focus of this paper. Still, it is a vital part of any open-set classification system (see Equation 1). Therefore, we compared the performance of our closed-set classification system to those obtained with other systems. For this purpose, we used the datasets provided for subtask A of task 1. Using the datasets of subtask C for this purpose is impossible because the score also includes the system's outlier detection performance, which also affects the closed-set classification accuracy. The results can be found in Figure 2.

It can be seen that our closed-set classification accuracies are significantly higher than the ones obtained with the baseline system and with the system provided in [13]. Furthermore, our ensemble, which utilizes all three features types, performs significantly better than all models based on a single feature type. This justifies the final design of our closed-set classification system. We also included the winning system [22] to give an example of how much performance can be gained when improving the system.

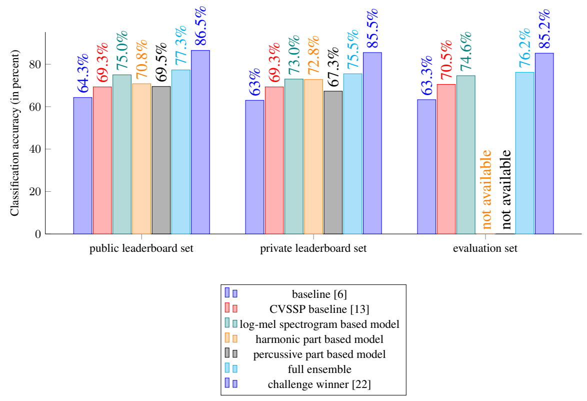


Figure 2: Comparison of closed-set classification accuracies obtained in task 1A.

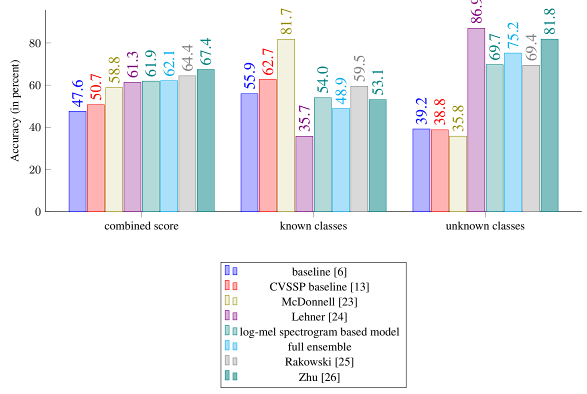


Figure 3: Comparison of all submitted systems' open-set classification scores obtained on the evaluation dataset of task 1C (final challenge results).

3.2. Open-set classification

The open-set classification performances on the evaluation dataset of task 1C obtained with all submitted systems including ours can be found in Figure 3. One can see that all systems outperform the baseline system as well as the system presented in [13] on the evaluation dataset. More concretely, the relative performance gain of our proposed system with respect to the score is 30.5% when comparing to the challenge's baseline system. Since these improvements are larger in this open-set setting than in task 1A, much of the success can be credited to using DCAEs for outlier detection. In contrast to the results obtained in the closed-set classification task, the accuracy of our ensemble significantly degraded for the known classes, which looks a bit strange at first sight. But since the accuracy significantly improved on the unknown classes, more test samples were predicted as outliers by the ensemble also resulting in more false rejections and a lower accuracy on the known classes.

When comparing all submitted systems, one can distinguish the three leftmost systems (baseline [6], CVSSP baseline [13] and McDonnell [23]), which have a relatively low accuracy on the unknown classes and thus are detecting only a few outliers, from the other systems. Because this also results in fewer false rejections, these three systems have a comparatively high accuracy on the known classes. Compared to them, the other systems have a much higher accuracy on unknown classes and thus are detecting more outliers. However, since this also results in more false rejections, this degrades the performance on the known classes. It is worth pointing out that Lehner [24] has the highest accuracy on the unknown classes but the lowest accuracy on the known classes. In conclusion, this system detects too many outliers. The fact that our system ranks 3rd among all submitted systems shows that the overall structure of our open-set acoustic scene classification system is suitable for the task.

4. CONCLUSIONS AND FUTURE WORK

In this paper, we presented an open-set acoustic scene classification system that has been submitted to task 1C of the DCASE challenge 2019. It has been shown that a combination of CNNs for closed-set classification and DCAEs for outlier detection yields significant improvements over the baseline system. In fact, our system outperformed the baseline system by 30.5% without using any external data resources, increasing the score from 0.476 to 0.621 on the evaluation dataset. Using the presented system, our team ranked 3rd overall in task 1C of the challenge.

For future work, we plan to improve the structure of the DCAE. In addition, using the mean squared error of DCAEs for outlier detection is just a heuristic since the loss function to be optimized does not directly aim at rejecting unknown examples. Instead of using DCAEs, one may also train a neural network with another loss function that is specifically targeted at one-class classification (e.g. [27]). The results can also be compared to those obtained with an OpenMax layer [28], which can be understood as the open-set version of a softmax layer. Another path to be investigated is to make use of embeddings as for example the L3-Net embedding [29] or OpenL3 [30]. These embeddings could be used in the same way as i-vectors [31] or x-vectors [32] in open-set speaker identification (see e.g. [10]). Note that both, i-vector and x-vector, have been successfully applied for closed-set acoustic scene classification [33, 34] in past editions of the DCASE challenge. Thus, utilizing embeddings seems promising. Lastly, using external data for training the models or improving our relatively simple closed-set classification model with more sophisticated techniques as for example an attention mechanism [35] also improves the open-set performance.

5. REFERENCES

[1] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge," *Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 2, pp. 379–393, Feb 2018.

[2] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification: An overview of DCASE 2017 challenge entries," in *16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, September 2018, pp. 411–415.

[3] W. J. Scheirer, A. Rocha, A. Sapkota, and T. E. Boult, "Towards open set recognition," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, July 2013.

[4] W. J. Scheirer, L. P. Jain, and T. E. Boult, "Probability models for open set recognition," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2317–2324, 2014.

[5] C. C. Aggarwal, *Outlier Analysis*, 2nd ed. Springer, 2017.

[6] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification in DCASE 2019 challenge: closed and open set classification and data mismatch setups," in *Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, 2019.

[7] D. Battaglino, L. Lepauloux, and N. Evans, "The open-set problem in acoustic scene classification," in *International Workshop on Acoustic Signal Enhancement (IWAENC)*. IEEE, 2016.

[8] S. Hawkins, H. He, G. Williams, and R. Baxter, "Outlier detection using replicator neural networks," in *International Conference on Data Warehousing and Knowledge Discovery*. Springer, 2002, pp. 170–180.

[9] J. Chen, S. Sathe, C. Aggarwal, and D. Turaga, "Outlier detection with autoencoder ensembles," in *International Conference on Data Mining*. SIAM, 2017, pp. 90–98.

[10] K. Wilkinghoff, "On open-set speaker recognition with i-vectors," *Preprint (submitted)*, 2019.

[11] Y. Han, J. Park, and K. Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, pp. 56–50, 2017.

[12] S. Gharib, H. Derrar, D. Niizumi, T. Senttula, J. Tommola, T. Heittola, T. Virtanen, and H. Huttunen, "Acoustic scene classification: A competition review," in *28th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2018, pp. 235–240.

[13] Q. Kong, Y. Cao, T. Iqbal, Y. Xu, W. Wang, and M. D. Plumbley, "Cross-task learning for audio tagging, sound event detection and spatial localization: DCASE 2019 baseline systems," *arXiv preprint arXiv:1904.03476*, 2019.

[14] D. Fitzgerald, "Harmonic/percussive separation using median filtering," in *13th International Conference on Digital Audio Effects (DAFX)*, 2010.

[15] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *14th Python in Science Conference*, 2015, pp. 18–25.

[16] Y. Sakashita and M. Aono, "Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divisions," *Detection and Classification of Acoustic Scenes and Events (DCASE) Challenge Report*, 2018.

[17] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[18] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2016, pp. 265–283.

[19] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *International Conference on Learning Representations (ICLR)*, 2018.

[20] T. DeVries and G. W. Taylor, "Improved regularization of convolutional neural networks with cutout," *arXiv preprint arXiv:1708.04552*, 2017.

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[22] H. Chen, Z. Liu, Z. Liu, P. Zhang, and Y. Yan, "Integrating the data augmentation scheme with various classifiers for acoustic scene modeling," DCASE2019 Challenge, Tech. Rep., June 2019.

[23] W. Gao and M. McDonnell, "Acoustic scene classification using deep residual networks with late fusion of separated high and low frequency paths," DCASE2019 Challenge, Tech. Rep., June 2019.

[24] B. Lehner and K. Koutini, "Acoustic scene classification with reject option based on resnets," DCASE2019 Challenge, Tech. Rep., June 2019.

[25] A. Rakowski and M. Kośmider, "Frequency-aware CNN for open set acoustic scene classification," DCASE2019 Challenge, Tech. Rep., June 2019.

[26] H. Zhu, C. Ren, J. Wang, S. Li, L. Wang, and L. Yang, "DCASE 2019 challenge task1 technical report," DCASE2019 Challenge, Tech. Rep., June 2019.

[27] L. Ruff, N. Görnitz, L. Deecke, S. A. Siddiqui, R. Vandermeulen, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *International Conference on Machine Learning (ICML)*, 2018, pp. 4390–4399.

[28] A. Bendale and T. E. Boult, "Towards open set deep networks," in *International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016, pp. 1563–1572.

[29] R. Arandjelovic and A. Zisserman, "Look, listen and learn," in *International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 609–617.

[30] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3852–3856.

[31] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2010.

[32] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 165–170.

[33] H. Eghbal-Zadeh, B. Lehner, M. Dorfer, and G. Widmer, "CP-JKU submissions for DCASE-2016: a hybrid approach using binaural i-vectors and deep convolutional neural networks," in *AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE)*. IEEE, 2016.

[34] H. Zeinali, L. Burget, and J. Cernocky, "Convolutional neural networks and x-vector embedding for DCASE2018 acoustic scene classification challenge," pp. 202–206, 2018.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 5998–6008.

MAVD: A DATASET FOR SOUND EVENT DETECTION IN URBAN ENVIRONMENTS

Pablo Zinemanas, Pablo Cancela, Martín Rocamora

Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay
{pzinemanas, pcancela, rocamora}@fing.edu.uy

ABSTRACT

We describe the public release of a dataset for sound event detection in urban environments, namely MAVD, which is the first of a series of datasets planned within an ongoing research project for urban noise monitoring in Montevideo city, Uruguay. This release focuses on traffic noise, MAVD-traffic, as it is usually the predominant noise source in urban environments. An ontology for traffic sounds is proposed, which is the combination of a set of two taxonomies: vehicle types (e.g. car, bus) and vehicle components (e.g. engine, brakes), and a set of actions related to them (e.g. idling, accelerating). Thus, the proposed ontology allows for a flexible and detailed description of traffic sounds. We also provide a baseline of the performance of state–of–the–art sound event detection systems applied to the dataset.

***Index Terms*—** SED database, traffic noise, urban sound

1. INTRODUCTION

Recent years have witnessed the upsurge of the Smart City concept, i.e. networks of Internet of Things (IoT) sensors used to collect data in order to monitor and manage city services and resources. Noise levels in cities are often annoying or even harmful to health, being consequently among the most frequent complaints of urban residents [1]. This fuelled the development of technologies for monitoring urban sound environments, mainly oriented towards the mitigation of noise pollution [2, 3]. The application of signal processing and machine learning has lead to the automatic generation of high–level descriptors of the sound environment. This encompasses the problem of sound event detection (SED), as an attempt at describing the acoustic environment through the sounds encountered in it. It is defined as the task of finding individual sound events, by indicating the onset time, the duration and a text label describing the type of sound [4, 5].

The SED problem is usually approached within a supervised learning framework, using a set of predefined sound event classes and annotated audio examples of them [5, 6]. One of the most challenging aspects of the problem is that it involves the detection of overlapping sound events. In addition, given the intrinsic variability of sound sources of the same type (e.g. cars) and the influence of the acoustic environment (e.g. reverberation, distance) for different locations and situations, the acoustic features of each class can exhibit great diversity. The solutions proposed typically use a mel–spectral representation of the audio signal as the input features, and apply different classification methods, including Random Forest [7], GMM [8], and more recently convolutional neural networks [9, 10] and recurrent neural networks [11, 12, 13].

1.1. Related work

Publicly available datasets for SED are of crucial importance to foster the development of the field as they encourage reproducible research and fair comparison of algorithms. In this respect, the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge, held for the first time in 2013 and repeated every year since 2016, has established a benchmark for sound event detection using open data [6, 14].

Two of the datasets used in the DCASE challenge for SED in urban environments are part of the TUT database (TUT Sound Events 2016 and 2017), which was collected in residential areas in Finland by Tampere University of Technology (TUT) and contain overlapping sound events manually annotated [8]. The classes are defined during the labeling process. In a first step, the participants are asked to mark all the sound events freely, and later the labels are grouped into more general concepts. In addition, the tags must be composed of a noun and a verb, such as ENGINE ACCELERATING [8].

Manual annotation of audio recordings for SED is a very time consuming task, primarily due to multiple overlapping sounds, which has limited the amount of annotated audio available. A way to alleviate the work involved in manual annotation is to use weak labels, as in DCASE 2017 task 4 [14], which indicate the presence of a source without giving time boundaries. Another approach is to create synthetic audio mixtures using isolated sound events. This is the approach adopted in the URBAN-SED dataset [9], that contains synthesized soundscapes with sound event annotations generated using Scaper [9] (a software library for soundscape synthesis). The original sound events are extracted from the UrbanSound8K dataset [15], where a taxonomic categorization of urban sounds is proposed. At the top level, four groups are defined: HUMAN, NATURAL, MECHANICAL and MUSICAL, which have been used in previous works. To define the lower levels, the most frequent noise complaints in New York city from 2010 to 2014 were used [15].

Table 1 summarizes the characteristics of the available datasets for SED in urban environments. While the TUT datasets are limited to only one and two hours, the URBAN-SED dataset comprises 30 hours of audio but contains synthetic audio mixtures instead of real recordings. Other resources for research on urban sound environments are available, such as the SONYC Urban Sound Tagging (SONYC–UST) dataset [2], though they are not specifically devoted to the SED problem. If traffic sounds are to be considered, the DCASE 2017 task 4 training dataset has only weak labels, the TUT database has only a moderate amount of traffic activity since it was recorded in a calm residential area, and only three out of the ten classes in URBAN-SED are related to traffic (i.e. CAR_HORN, ENGINE_IDLING and SIREN). Therefore, there is plenty of room for expanding the existing resources, in particular, for specific applications' scenarios such traffic noise monitoring.

https://doi.org/10.33682/kfmf-zv94

dataset	classes	hours	type	label
TUT-SE 2016 [8]	7	1	recording	strong
TUT-SE 2017 [8]	6	2	recording	strong
URBAN-SED [9]	10	30	synthetic	strong
DCASE2017 #4 [14]	17	141	10-s clips	weak
MAVD-traffic	21	4	recording	strong

Table 1: Available datasets for SED in urban environments, along with the released dataset.

1.2. Our contributions

We describe the first public release of a dataset for SED in urban environments, called MAVD, for Montevideo Audio and Video Dataset. This release focuses on traffic sounds, namely MAVD-traffic, which corresponds to the most prevalent noise source in urban environments. The records were generated in various locations in Montevideo city and include both audio and video files, along with annotations of the sound events. The video files, apart from being useful for manual annotation, open up new research possibilities for SED using audio and video. The annotations follow a new ontology for traffic sounds that is proposed in this work. It arises from the combination of a set of two taxonomies: vehicle types (e.g. car, bus) and vehicle components (e.g. engine, brakes), and a set of actions related to them (e.g. idling, accelerating). Thus, the proposed ontology allows for a flexible and detailed description of traffic sounds. In addition, we provide a baseline of the performance of state–of–the–art SED systems applied to the MAVD-traffic dataset. Finally, we discuss possible directions for further research and some efforts we undertaken to improve and extend current dataset.

2. ONTOLOGY

The proposed ontology focuses on traffic noise. Consequently, vehicles (such as cars, buses, motorcycles and trucks) are the main sources of noise and define the classes of interest. However, vehicles generate different types of sounds, for example those related to the braking system, the rolling of the wheels or the engine, calling for a classification that is more specific than just the type of vehicle. One way to approach it, is by classifying sound events with different correlated attributes, such as the sound source (object), the action, and the context [16]. These attributes can be defined by one or several taxonomies, implying that the same event can be classified by several schemes simultaneously [16]. In this case, the context is defined by urban environments where traffic noise is predominant. Then, sound sources and actions can be described by several taxonomies, for instance, one that defines the type of vehicle and other that defines the internal components that generate the sound.

We define an ontology based on a graph like the one shown in Figure 1, which consists of two taxonomies that blend in the middle: the top one describes the categories of vehicles; and the bottom one describes the categories of components. The categories of components are further combined with a set of actions to form an object-action pair (e.g. ENGINE IDLING, ENGINE ACCELERATING).[1]

The categories indicated in bold are those that are called *basic level* (CAR, BUS, etc. for vehicles and ENGINE, WHEEL, etc. for components). These two taxonomies of the ontology are merged into what is called the *subordinate level* (depicted in italics), which are combinations of elements of the categories of vehicles and components, with the aim of providing a more detailed description of the noise source (e.g. CAR/ENGINE IDLING, BUS/COMPRESSOR). Note that the diagram of Figure 1 does not show all the class labels.

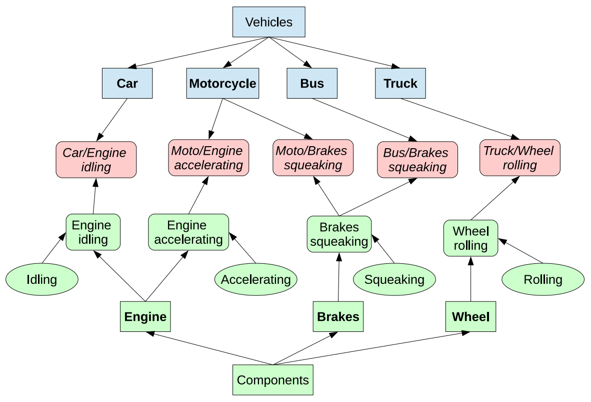


Figure 1: Graph representing the ontology. The top taxonomy refers to the vehicle categories and the bottom one to the components. The basic levels are indicated in bold and the subordinate level is marked in italics. The rectangle nodes denote objects; the ellipses denote actions; and the rounded rectangles indicate objects–actions pairs.

3. DATASET

3.1. Recordings

The recordings were produced in Montevideo, the capital city of Uruguay, which has population of 1.4 million people. Four different locations were included in this release of the dataset, corresponding to different levels of traffic activity and social use characteristics:

L1. Residential area, with several shops and many buses.

L2. Park area. No housing or shops. Some light traffic nearby.

L3. Park/residential area. Similar to location L2, but next to a residential area, with more traffic noise and less nature sounds.

L4. Residential area, with a few shops and some buses.

The sound was captured with a SONY PCM-D50 recorder at a sampling rate of 48 kHz and a resolution of 24 bits. The video was recorded with a GoPro Hero 3 camera at a rate of 30 frames per second and a resolution of 1920×1080 pixels. Audio and video files of about 15–minutes long were recorded at different times of the day in the different locations.

Some basic processing was done to generate the files of the dataset from the raw recordings. This included the synchronization of audio and video, the removal of windy sections and the segmentation into excerpts of approximately five minutes to facilitate their manipulation. The train and validation sets are composed of 24 and 7 files from the location L1 respectively, while the test set consists of 16 files from the L2, L3 and L4 locations[2]. The dataset totals 233 minutes (almost 4 hours, as shown in Table 1), of which 117 minutes correspond to the train set, 33 minutes to the validation set and 83 minutes to the test set.

[1]This could also be done in the top taxonomy for the vehicles, for example BUS PASSING BY, CAR STOPPING, etc., but was considered redundant.

[2]In train/validation we favoured the location with more events (L1) but other fold schemes could be implemented using the metadata information.

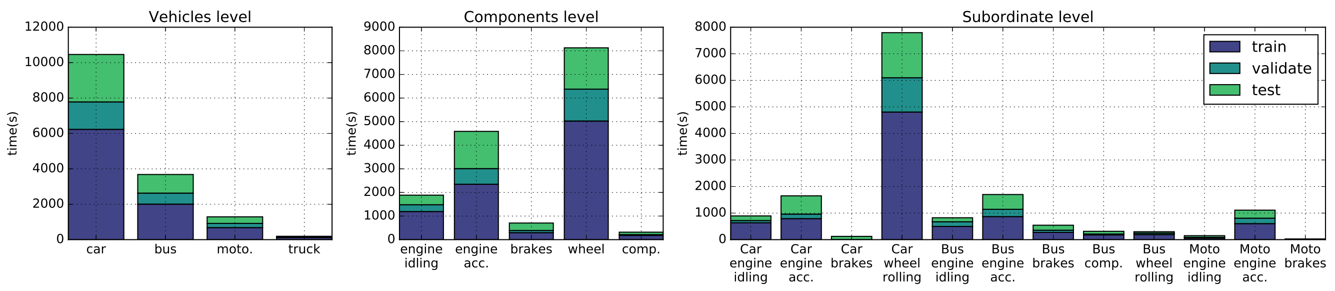


Figure 2: Total time for each class in the dataset. The first two graphs correspond to the basic levels, and the third one to the subordinate level.

3.2. Annotation

The ELAN [17] software was used to manually annotate the recordings of the dataset. The software allows the user to simultaneously inspect several audio and/or video recordings and produce annotations time–aligned to the media. During the annotation process the software session displayed the audio waveform, the video record and the spectrogram of the audio signal. For the latter, an auxiliary video file was generated for each recording, showing the spectrogram of the audio signal and a vertical line indicating current time instant (as shown in Figure 3). The annotations can be created on multiple layers, which can be hierarchically interconnected. This feature is a perfect fit for the taxonomies' approach defined above.

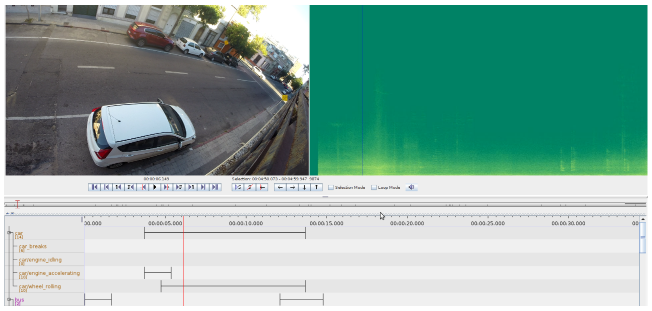


Figure 3: Screenshot of the ELAN software showing MAVD–traffic data: at the top the video and the spectrogram (with a marker indicating the instant being labeled) and at the bottom the annotations.

The annotation process was carried out in two steps. First, the vehicle categories were labeled (e.g. CAR, BUS). Then, for each of the marked segments, the labels of the component categories (e.g. ENGINE IDLING) were annotated to form the subordinate level. Figure 2 shows the total duration of the events for the three category types. Note that the dataset is highly unbalanced, especially the subordinate level, being CAR/WHEEL ROLLING the predominant class.

4. EXPERIMENTS AND RESULTS

4.1. Experiments

We devised two different experiments to provide a baseline of the SED performance on the MAVD–traffic dataset. For the first experiment, we used a Random Forest classifier with the acoustic features defined as follows. We extracted 20 mel–frequency cepstral coefficients (MFCC) using the energy in 40 mel bands. The MFCCs were calculated in frames of 40 ms overlapped 50% and using a Hamming analysis window. Besides, first and second derivatives were calculated (ΔMFCC, Δ^2MFCC), to describe the temporal variations of the coefficients. The features were computed with *librosa* (version 0.6.1) [18] and the Random Forest models were implemented with *scikit-learn* (version 0.17) [19].

For the second experiment, we used the convolutional neural network for SED proposed by Salamon et. al in [9] (S–CNN). The input of this network is a one–second length mel–spectrogram and has three convolutional layers followed by three fully–connected layers. The final layer is a sigmoid that performs the classification task (the number of units is equal to the number of classes). First we trained the S–CNN model with the URBAN–SED dataset using the same strategy used in [9]. Then, we used a fine–tuning strategy in order to specialize the network to the MAVD–traffic dataset. We replace the last sigmoid layer of the network to accomplish the classification task of the MAVD-traffic dataset. The parameters of the other layers of the network were kept unchanged during the fine–tuning training process. The S–CNN model was implemented in *keras* (version 2.2.0) [20] using *tensorflow* (version 1.5.0) [21].

4.2. Metrics

The performance measures typically used for the SED problem are: F–score ($F1$) and Error Rate (ER), on a fixed time grid [22]. The detected sound events are compared with the *ground–truth* in one–second length segments. Based on the number of false positives (FP) and false negatives (FN), the values of the *precision* (P) and *recall* (R) are computed. Then, the F–score ($F1$) is calculated as:

$$F1 = \frac{2PR}{P+R} = \frac{2TP}{2TP+FN+FP}. \tag{1}$$

The error rate (ER) is calculated in terms of insertions $I(k)$, deletions $D(k)$ and substitutions $S(k)$ in each segment k. A substitution is defined as the case in which the system detects an event in a segment but with the wrong label. This corresponds to a simultaneous FP and FN for the segment. The remaining FP not included in the substitutions are considered insertions and the remaining FN as deletions. Finally, the ER is calculated considering all errors as:

$$ER = \frac{\sum_{k=1}^{K} S(k) + \sum_{k=1}^{K} D(k) + \sum_{k=1}^{K} I(k)}{\sum_{k=1}^{K} N(k)}, \tag{2}$$

where K is the total number of segments and $N(k)$ is the number of active classes in the *ground-truth* at segment k [8, 22].

The values of $F1$ and ER are usually calculated globally over the full set of segments and classes simultaneously. They can also

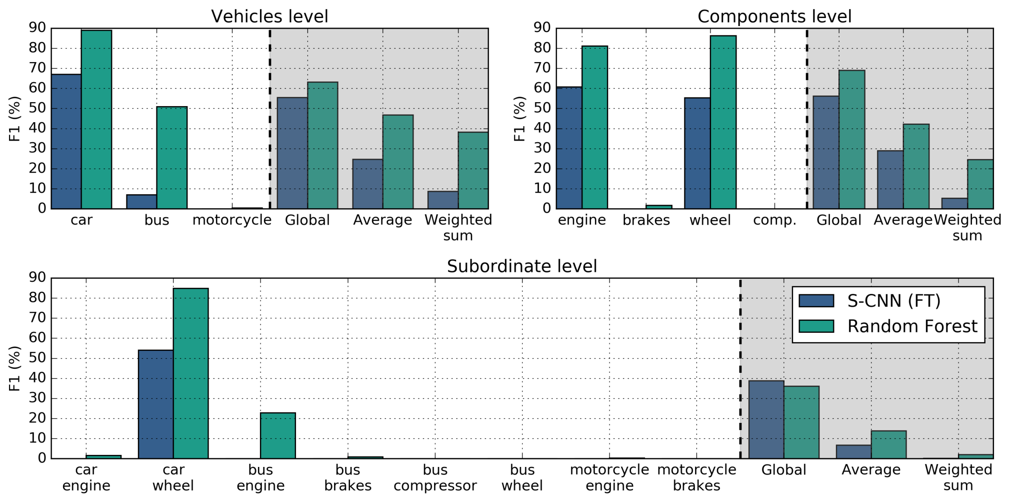


Figure 4: Comparison of the SED results for both S-CNN and Random Forest systems applied to the MAVD–traffic dataset. The performance is shown at the basic and subordinate levels for the different classes and for the three discussed metrics: Global, Average and Weighted sum.

be calculated restricted to each class and then averaged, which are denoted as $\bar{F}1$ and $\bar{ER}$ respectively. This average is calculated as:

$$\bar{M} = \frac{1}{C}\sum_{c=1}^{C} M_c, \tag{3}$$

where C is the number of classes and M_c is the metric for class c.

These global metrics can bias the SED algorithms to detect only the majority class. This is illustrated by the results of DCASE challenges 2016 and 2017, in which the algorithms that obtained better global results actually detect only the majority class [6, 14].

We aim to improve these evaluation metrics (ER and $F1$) in the case of multi–class SED systems trained with very unbalanced data, by increasing the importance of detecting the minority classes. To do so, we propose a weighted sum of the metric values as,

$$\hat{M} = \sum_{c=1}^{C} w_c M_c, \qquad w_c = \frac{1/N_c}{\sum_{j=1}^{C} 1/N_j} \tag{4}$$

where N_c is the number of active segments for class c, and w_c is the weight for each class, which is designed to give more importance to the minority classes. Note that w_c increases when N_c decreases, as expected. The sum in the denominator ensures that $\sum_c w_c = 1$.

4.3. Results

We trained the Random Forest and the S–CNN models for the three class levels (vehicles, components and subordinate) and obtained the results shown in Figure 4 and in Table 2. Note that the S–CNN models tend to classify only the majority class while yielding quite good results for the global ER and $F1$ metrics, as discussed in Section 4.2. On the other hand, the weighted sum metrics, $\hat{ER}$ and $\hat{F}1$, clearly penalize the detection of only the majority class. The Random Forest models perform better in detecting the minority classes (see the BUS class), reaching higher values of the weighted sum metrics. The source code for training the models and reproducing these results on the MAV-traffic dataset is publicly available.[3]

[3]`https://github.com/pzinemanas/MAVD-traffic`

		Global		Weighted sum	
Level	Model	ER	$F1(\%)$	$\hat{ER}$	$\hat{F}1(\%)$
Vehicles	RF	0.54	63.1	0.71	38.2
	S–CNN	0.51	55.5	0.97	8.70
Components	RF	0.49	69.0	0.80	24.6
	S–CNN	1.17	56.2	1.03	5.35
Subordinate	RF	0.78	36.1	0.96	1.98
	S–CNN	0.70	38.9	1.00	0.17

Table 2: Results for Random Forest (RF) and S-CNN using the original (Global) and the proposed (Weighted sum) metrics.

5. CONCLUSION

In this work a new dataset for SED in urban environments is described and publicly released.[4] The dataset focuses on traffic noise and was generated from real recordings in Montevideo city. Apart from audio recordings it, also includes synchronized video files.[5] The dataset was manually annotated using an ontology proposed in this work, which combines two taxonomies (vehicles and component–action pairs) for a detailed description of traffic noise sounds. Since the taxonomies follow a hierarchy they can be used with different levels of detail. The performance of two SED system is reported as a baseline for the dataset. Some considerations are given regarding the evaluation metrics for class–unbalanced datasets. In future work, we will increase the size of the dataset, by including other locations with different levels of traffic activity. We also plan to address urban soundscapes in which other noise sources are predominant, such as those related to social, construction or industrial activities. In addition, image processing techniques will be applied to the video files to develop a multi–modal SED system.

[4]Available from Zenodo, DOI 10.5281/zenodo.3338727

[5]In this release, the video files are available in low resolution as we are anonymizing them, after which they will be available in high resolution.

6. REFERENCES

[1] H. Ising and B. Kruppa, "Health effects caused by noise: Evidence in the literature from the past 25 years," *Noise & health*, vol. 6, pp. 5–13, 11 2004.

[2] J. P. Bello, C. Silva, O. Nov, R. L. Dubois, A. Arora, J. Salamon, C. Mydlarz, and H. Doraiswamy, "SONYC: A system for monitoring, analyzing, and mitigating urban noise pollution," *Communications of the ACM*, vol. 62, no. 2, pp. 68–77, Feb 2019.

[3] D. K. Daniel Steele and C. Guastavino, "The sensor city initiative: cognitive sensors for soundscape transformations," in *Geoinformatics for City Transformations*. Technical University of Ostrava, January 2013, pp. 243–253.

[4] J. P. Bello, C. Mydlarz, and J. Salamon, *Computational Analyis of Sound Scenes and Events*. Springer, 2017, ch. 13 Sound Analysis in Smart Cities.

[5] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational Analyis of Sound Scenes and Events*. Springer, 2017, ch. 1 Introduction to Sound Scene and Event Analysis.

[6] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley, "Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 2, pp. 379–393, Feb 2018.

[7] B. Elizalde, A. Kumar, A. Shah, R. Badlani, E. Vincent, B. Raj, and I. Lane, "Experimentation on the dcase challenge 2016: Task 1 - acoustic scene classification and task 3 - sound event detection in real life audio," in *Detection and Classification of Acoustic Scenes and Events 2016*, 2016.

[8] A. Mesaros, T. Heittola, and T. Virtanen, "TUT database for acoustic scene classification and sound event detection," in *In 24rd European Signal Processing Conference 2016 (EUSIPCO 2016), Budapest, Hungary*, 2016.

[9] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello., "Scaper: A library for soundscape synthesis and augmentation," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, october 2017.

[10] I.-Y. Jeong, S. Lee, Y. Han, and K. Lee, "Audio event detection using multiple-input convolutional neural network," DCASE2017 Challenge, Tech. Rep., September 2017.

[11] S. Adavanne, G. Parascandolo, P. Pertila, T. Heittola, and T. Virtanen, "Sound event detection in multichannel audio using spatial and harmonic features," in *Detection and Classification of Acoustic Scenes and Events 2016*, 2016.

[12] R. Lu and Z. Duan, "Bidirectional GRU for sound event detection," DCASE2017 Challenge, Tech. Rep., September 2017.

[13] E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *Transactions on Audio, Speech and Language Processing: Special issue on Sound Scene and Event Analysis*, vol. 25, no. 6, pp. 1291–1303, June 2017.

[14] A. Mesaros, A. Diment, B. Elizalde, T. Heittola, E. Vincent, B. Raj, and T. Virtanen, "Sound event detection in the DCASE 2017 Challenge," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 6, pp. 992–1006, June 2019.

[15] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *22st ACM International Conference on Multimedia (ACM-MM'14)*, Orlando, FL, USA, Nov. 2014.

[16] C. Guastavino, *Computational Analyis of Sound Scenes and Events*. Springer, 2017, ch. 7 Everyday Sound Categorization.

[17] Max Planck Institute for Psycholinguistics, "ELAN - the lenguage active." [Online]. Available: tla.mpi.nl/tools/tla-tools/elan/

[18] B. McFee, M. McVicar, C. Raffel, D. Liang, O. Nieto, E. Battenberg, J. Moore, D. Ellis, R. Yamamoto, R. Bittner, D. Repetto, P. Viktorin, J. F. Santos, and A. Holovaty, "librosa: 0.4.1," Oct. 2015. [Online]. Available: https://doi.org/10.5281/zenodo.32193

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[20] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[21] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[22] A. Mesaros, T. Heittola, and T. Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, p. 162, 2016.