A Simple Relational Classifier

Sofus A. Macskassy and Foster Provost

NYU Stern School of Business
44 W. 4th Street
New York, NY 10012, U.S.A.
{smacskas,fprovost}@stern.nyu.edu

Abstract. We analyze a Relational Neighbor (RN) classifier, a simple relational predictive model that predicts only based on class labels of related neighbors, using no learning and no inherent attributes. We show that it performs surprisingly well by comparing it to more complex models such as Probabilistic Relational Models and Relational Probability Trees on three data sets from published work. We argue that a simple model such as this should be used as a baseline to assess the performance of relational learners.

1 Motivation

In recent years, we have seen remarkable advances in algorithms for relational learning, especially statistically based algorithms. These algorithms have been developed in a wide variety of different research fields and problem settings. Relational data differ from traditional data in that they violate the instance-independence assumption. Instances can be related, or linked, in various ways. The label of an instance might depend on the instances it is related to either directly or through arbitrarily long chains of relations. This relational structure further complicates matters as it makes it harder, if not impossible, to separate the data cleanly into test and train sets without losing much relational information. Recent work has begun to investigate foundational issues within relational learning, such as the dimensions across which learners can be compared [11, 14, 25] as well as issues of link dependencies [13]. We broaden these investigations by describing a baseline method to which relational learners should be compared when assessing how well they have extracted a useful model from the given relational structure—beyond what can be achieved by looking only at known class labels of related neighbors.

Recent probabilistic relational learning algorithms—e.g., Probabilistic Relational Models (PRMs) [16, 10, 27], Relational Probability Trees (RPTs) [22] and Relational Bayesian Classifiers (RBCs) [23]—search the relational space for useful attributes and relational structure of neighbors (possibly more than one link away). While there are other relational learning algorithms available [7, 9, 6], we focus in this paper on the three named algorithms.

We know from classical machine learning that even very simple statistical methods such as naive Bayes can perform remarkably well even when compared to more complex methods. However, a question that has yet to receive much attention is how much of the performance of relational learners is due to their complexity and how much can be attributed to the relational structure of the data. In the latter case, even a simple model using no learning may perform quite well.

While results reported on the relational classifiers (PRMs, RPTs and RBCs) have been compared to non-relational baseline learners (e.g., the naive Bayes classifier [5, 15, 19] or C4.5 [26]), a simple relational classifier is an equally important, and perhaps a more appropriate, point of comparison. We analyze here the Relational Neighbor (RN) [25] classifier as such a simple classifier which uses only class labels of known related instances and does no learning. We show that it performs competitively to these related learning algorithms when compared against their published results on three different relational data sets. Although we believe that the complex methods can add value, we argue that in order to assess how well they have learned from relations, they should be compared against baseline methods such as RN.

The rest of the paper is outlined as follows. We first describe a simple relational neighbor (RN) classifier, followed by three studies comparing to reported results using the PRM, RPT and RBC relational learners. We then propose a probabilistic version of the RN and show that unexpectedly it does not add value in the cases outlined in this paper, though it can do so in other domains. We describe and report results on one such domain, and conclude with final remarks.

2 A Relational Neighbor Classifier

The Relational Neighbor (RN) classifier estimates class probabilities solely based on entities of the same type whose class labels are known.[1] The classifier works by making two strong, yet often reasonable, assumptions: (1) some entities' class labels are known within the same linked structure (see [25] for more discussion), and (2) the entities exhibit homophily—entities related to each other are similar and likely belong to the same class along one or more dimensions [2, 18]. The classifier may not perform well if entities are isolated or if no labels are known.

Definition. The relational-neighbor classifier estimates $P(c|e)$, the class-membership probability of an entity e belonging to class c, as the (weighted) proportion of entities in D_e that belong to class c. We define D_e as the set of entities that are linked to e. Thus,

$$P(c|e) = \frac{1}{Z} \sum_{\{e_j \in D_e | \text{label}(e_j) = c\}} w(e, e_j), \tag{1}$$

where $Z = \sum_{e_i \in D_e} w(e, e_i)$, and $w(e, e_i)$ is the weight of the link[2] between entities e and e_i. Entities in D_e that are not of the same type as e are ignored. If D_e is empty or has no entities with known class labels, then the RN will estimate e based on the class prior (of the known labels).

For example, consider a graph of linked web pages, each belong to a class (e.g., homepage vs. non–homepage). If we consider a link to be undirected (e.g., two pages

[1] We have based RN on the Relational Vector Space Model (RVSM) using the $s_{\text{wend}}(e, i)$ RVS scoring function [1].

[2] Note that the notion of "linked to" is domain dependent and, as we will show, different definitions can lead to (very) different performance.

are related if one links to the other, regardless of link directionality), then a RN classifier would classify a candidate page, p, as a homepage if the majority of pages related to p—either through being linked to p or linking to p—were known to be homepages.

However, we have a potential problem if all (or many) entities in D_e are unknown—we do not truly take the nature of relational data into account. For example, known information should propagate through the network to related instances. This idea of propagation has been used successfully in other work—e.g., iterative classification [21], relaxation labeling [3] and belief propagation [24], which is used in PRMs among others.

Definition. The iterative relational-neighbor classifier (RN*) iteratively classifies entities using the RN classifier in its inner loop. We define RN^i as the model at iteration i, where RN^0 defines what is initially known and RN^1 is equivalent to RN. At iteration i, RN^i uses the labels given by RN$^{(i-1)}$ to predict class-membership of currently unknown instances. In the case where the class-membership probability of a neighboring entity, $e_j \in D_e$, is a prediction from RN* and was not initially known, the class with the highest probability score is used. Thus, an entity e will be classified as `unknown` if the (weighted) majority is `unknown`.[3] For this paper, RN* stops when no unknown entities are left or when no new entities can be labeled (as could be the case when there are isolated components with no known labels).

3 Case Studies

We compare, in this section, the relational neighbor classifier to three different published results on relational learning. The question we ask in each of these case studies is whether we perform well enough to warrant the use of such a simple model as a baseline comparison for relational learning. To have a fair comparative study, we tried to replicate the original test environments closely.

3.1 CoRA

The first case study is based on the CoRA data set [17], which is a data set of academic papers within Computer Science. This corpus includes the full citation graph as well as labels for the topic of each paper (and potentially sub- and sub-sub-topics). We focused on 4240 papers within the machine learning topic with the classification task of predicting which of seven sub-topics a given paper belongs to [27]. We used all 4007 unique authors that we could identify in this subset. Thus, our graph differed from the original study [27] in which they report using 4187 papers and 1454 authors.

Papers can be linked in one of two ways, using a common author, or a citation. Thus, classification can be based on using only one of the given link types, or using both links between the two papers. We chose the latter, where we assign the 'weight' of a relation as the sum of the number of authors two papers have in common and the number of citations that link them to each other. This latter weight would ordinarily only be zero or one unless the two papers cite each other.

[3] In cases where too little propagation takes place, because of too much weight from unknown labels, the need for a majority of weight from a known class can be weakened. This was not necessary for the cases presented in this paper.

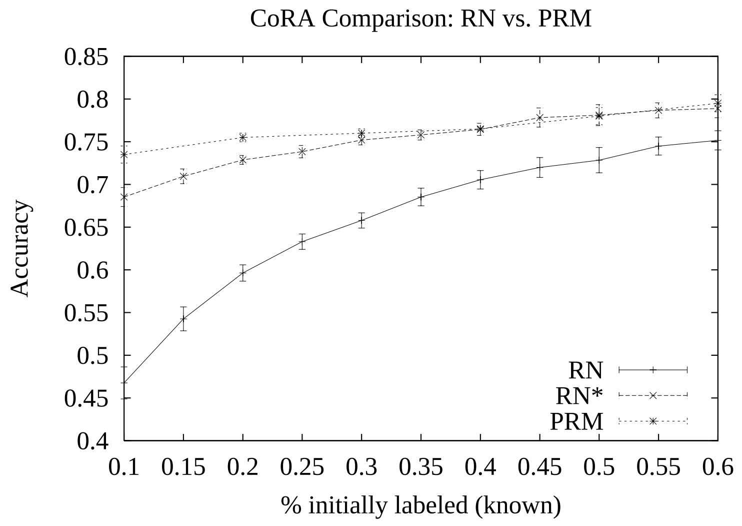

Fig. 1. Comparison of RN, RN* and PRM on the CoRA data set.

Using Probabilistic Relational Models (PRMs), the prior work reported accuracies ranging from 0.68 to 0.79, as the ratio of initially known labels increased from 10% to 60%. The best performance reported was a PRM that used both author and citation links as well as the words in the paper.

Using the same methodology as reported in the PRM study, we varied the proportion of papers for which the class is initially known from 10% to 60%. We varied in 5% increments; we performed a 10–fold cross-validation at each setting (the previous study performed a 5–fold cross-validation, varying the training set in 10% increments). Figure 1 compares the classification accuracy of RN with the reported results from the PRM, where the PRM used both citation links and author links.[4] Although RN only has an accuracy of 46% initially, it is able to reach relatively high performance once 50% of the papers are labeled, though it is still not as accurate as the PRM. Also shown in Figure 1, RN* estimates the classes of the unknown papers by repeatedly updating the class-probabilities of unknown nodes (using RN) until all nodes have been labeled. This took 3–5 iterations, decreasing as we increased the labels that were known initially. We see a marked improvement throughout the whole graph. RN* is competitive with the PRM, matching its performance once 35% of the labels are known initially.

We will only report on the RN* classifier in the next two studies due to its clear superiority.

[4] The PRM values were approximated from the graphs in the original paper.

3.2 IMDb

The second data set case study is based on the Internet Movie Database (IMDb)[5] data set, where we build models to predict movie box-office receipts [12]. We focus on movies released in the United States between 1996 and 2001 with the goal of predicting whether a movie "will be" a blockbuster (the opening weekend box-office receipts exceed $2 million) [22]. We used the database from the authors of the original study to extract "blockbuster" classifications. However, it was non–trivial to recreate the complete graph as described in the original work, which used 1364 movies (45% of those being blockbusters) [22]. We thus used a data set from the IMDb web-site. We identified 1441 movies released between 1996 and 2001 that we were able to link up with a "blockbuster" classification in the original database. 615 of the 1441 movies (42.6%) were classified as "blockbuster."

The IMDb data consists of movies, which have relations such as 'actor', 'director', 'producer', etc. Movies can be linked through these intermediate objects. RN can be based on a particular link type ($\mathrm{RN}_{<\mathrm{link\_type}>}$). Links between movies are through various other entities (actor, studios, production companies, etc.), and we consider the links to be typed by the entity through which they pass (e.g., $\mathrm{RN}_{\mathrm{producer}}$ means: how often does the producer produce blockbusters). Based on a suggestion from David Jensen, we consider four types of links: {`actor, director, producer, production company`[6]}.

Using Relational Probability Trees (RPTs) and Relational Bayesian Classifiers (RBCs), the prior work reported areas under the ROC curve (AUCs) of 0.82 and 0.85 for RPTs and RBCs, respectively, using a set of eight attributes on related entities, such as the most prevalent genre of the movie's studio.

We used a 10-fold cross-validation to generate predictions for all training examples. It is then straightforward to generate an ROC curve—using the class-membership probabilities produced by RN^*—and its AUC by pooling these predictions and sorting the prediction scores for the primary class ("blockbuster", in our case) [8]. In order to account for variance, we ran the 10-fold cross-validation 10 times, each time generating new folds. Table 1 shows the mean AUCs and their variances for the RN^* classifier on each of the four link types.

RN^* link-type	**AUC**	(variance)
actor	0.766	(0.003)
director	0.658	(0.007)
producer	0.850	(0.005)
prodco	0.862	(0.003)

Table 1. AUCs of RN^* using only 1 link type.

As is clear from Table 1, RN^* is very competitive, outperforming the RBC AUC score of 0.85. However, it may be possible to perform even better by considering more

[5] `http://www.imdb.com`

[6] We shorten 'production company' to 'prodco' when describing our results below.

than one link type as each link type obviously contribute some evidence of class-membership. Thus, $\mathrm{RN}^*_{\mathrm{actor+director}}$ would consider edges both along the `actor` as well as along the `director` links. To test this, we ran a simple forward feature-selection search: For each remaining (unused) link-type/feature, add it to the current best performer—starting with prodco, the best performer from Table 1—and keep the combination that reported the best performance. Keep adding one feature at a time until it stops improving the performance.[7] Using this methodology, we end up with the AUCs presented in Table 2.[8]

RN* link-type(s)	**AUC**	(variance)
prod+prodco	0.884	(0.003)
dir+prod+prodco	0.885	(0.003)

Table 2. AUCs of RN* using a forward-selection feature-based search to combine multiple link types.

We see that even with a relatively naive feature-based search, we were able to increase performance over that of using only one link type.

3.3 WebKB

The last case study we present is based on the data set collected by the WebKB Project [4].[9] It consists of a set of web pages from four computer science departments, with each page manually labeled into the categories: course, department, faculty, person, project, staff, student or other. This data set includes clearly defined `link-to` relations between pages. The classification task is to predict whether a page belongs to a student [22]. As with the prior study, we extracted the pages that have at least one incoming and one outgoing link, and kept remaining pages that either link to a page or are linked to by a page in this subset of pages. This resulted in a data set of 920 pages and 3036 background pages, giving us a total of 3956 pages. This differs from the prior work which had 910 extracted pages and a total of 3877 pages, including the background pages [20].

We create an edge between two pages if one page links to the other. We weight these edges by summing the number of links from one page to the other and vice versa. Thus, we do not take directionality into account, but treat these as generic links.

We performed a preliminary investigation, using the same 10-fold cross-validation methodology as described in the previous study—we used the 920 pages identified earlier to create the training folds, but allowed paths to any background page. This resulted

[7] The relational structure of the data complicates things, making it unclear what it means to use only the training set to perform the feature-selection. We circumvented this problem by using all the data in this study. These feature-selection results therefore might be optimistic.

[8] We also performed a brute-force analysis of all possible combinations of link types. For this study, the AUCs reported in Table 2 were the best two results among all possible combinations.

[9] We use the WebKB-ILP-98 data set.

in a low AUC score of 0.310 with a variance of (0.045)—worse even than random. However, when considering the relational structure of the domain, this is not so surprising. How often does a student link to another student? An observation in earlier work in this domain states that it is more likely that a student will link to her advisor or a group/project page rather than to her peers [4]. This would indicate that it is more likely that student pages would have intermediaries in common (e.g., they are likely to both link to their advisor, department or project page)—and that student pages are really 2 edges apart. Thus, we define "neighbors" for this domain as pages linked through some other page. The way we weigh these type of paths is to multiply the edge weights along this path (e.g., if a student page has 2 links to a group page, and a fellow student has 3 links to the same group page, then the weight along that path between those 2 students would be 6). This weight represents how many possible ways two pages could reach each other. Running the RN* classifier using the same cross-validation methodology as before resulted in a mean AUC of 0.948 with a variance of (0.003), a dramatic improvement in performance.

Using RPTs and RBCs, the prior work shows AUCs ranging from 0.716 to 1.0 for RPTs and 0.432 to 0.493 for RBCs [20]. The study used a set of ten attributes on related entities, such as the URL path and host, as well as structural attributes such as the number of in-links and out-links of each page. However, doing a direct comparison to this study is not possible. The results reported were based on a 4-fold cross-validation methodology in which one university is used as a holdout set while the remaining three are used for training. This methodology is obviously not appropriate for our simple classifier, as it needs to have direct links to known classes. Instead, the question we ask is how many instance labels we do need to know in order to perform comparably. Considering each university as a separate data set, we perform a similar study to that of the CoRA data set: we randomly pick $x\%$ of the pages and label them. The remaining labels are unknown. Running RN*, we can calculate the resulting AUC. We do this 10 times, each time randomly picking $x\%$ pages to label, giving us an average and standard deviation for the expected AUC for a given x. Doing this for $x \in \{\frac{1}{2}, 1, 5, 10, 15, 20, 25, 30, 40, \ldots, 90\}$, we graph the resulting AUC scores as x increases and compare these to the AUCs reported in the earlier study. Figure 2 shows, for each university, the resulting graphs.

Three immediate observations can be noted: in all cases the RN* was able to get close to its best performance even when given only 5% of the data. Second, in all cases, though less so on the Cornell data set, the RN* was competitive with RPT even having seen only 5% of the data. In fact, it was able to outperform RPT on the Washington data set, having seen only 5% of the data, and having seen 30% of the Wisconsin data made it perform on par with RPT. Third, in all cases, even when knowing the label of only 1 page (e.g., $x = \frac{1}{2}$), RN* was able to outperform RBC.

One important point that this comparison brings out is that even when a direct comparison is not possible, it is still possible to quantify the effectiveness of an algorithm by seeing how much data RN needs in order to perform comparably. In this case, having seen only 5% of the data yielded very close performance on 3 out of 4 data sets.

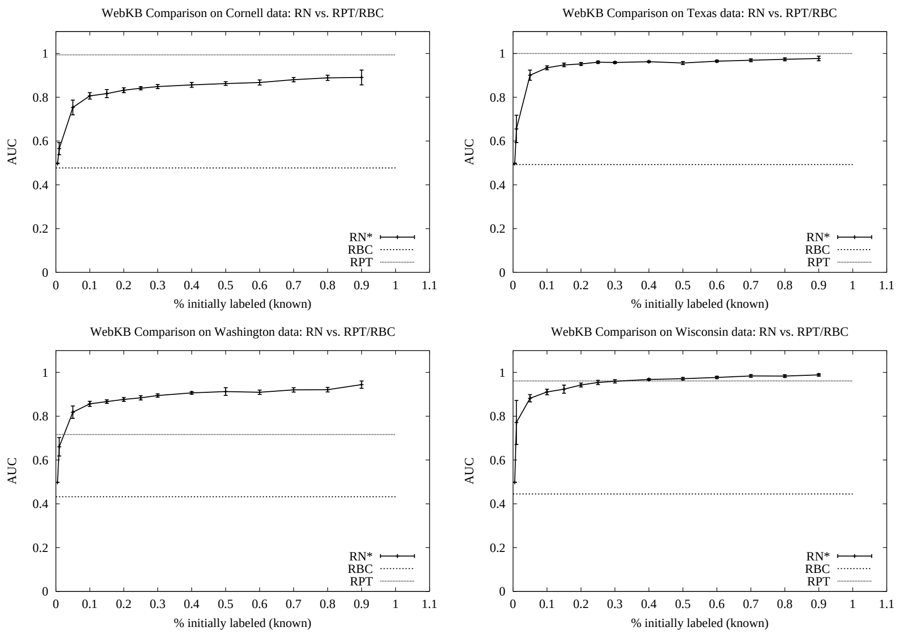

Fig. 2. Comparison of RN*, RPT and RBC on the four universities in the WebKB data set.

4 The Probabilistic Relational Neighbor Classifier

In Section 2 we described how RN generates class-membership probabilities based on known background entities in the set of neighbors, D_e. Intuitively it seems that even background entities whose labels are unknown should offer some evidence of class-membership. What if we could make some initial estimate of their class? Even the class prior might be useful, if no better initial estimation is available.

Definition. The probabilistic relational-neighbor classifier (pRN) estimates $P(c|e)$ as the (weighted) mean of the class-membership probabilities of the entities in D_e.

$$P(c|e) = \frac{1}{Z} \sum_{e_j \in D_e} w(e, e_j) * P(c|e_j), \tag{2}$$

where D_e, Z and $w(e, e_j)$ are defined as before. Background entities whose class labels are not known will be assigned the class priors.

What about an iterative version of pRN? As with RN, RN* seems sub-optimal—if we have properly estimated probabilities of class-membership, why not make use of them? Further, if an instance is unknown, why not give it some prior—such as the class priors from the known instances—and use that as the base to propagate? Might such a more evenly distributed scoring not perform better than the hard labeling of RN*?

Definition. The iterative probabilistic relational-neighbor classifier (pRN*) is similar to RN*, except it uses pRN in its inner loop and all initially unknown instances have their probabilities continuously updated. Unknown instances initially are assigned the class priors based on the initially known labels. Unlike RN*, pRN* updates class-probabilities of *all* initially unknown entities at every iteration. Because of the loopy nature of the propagation, there is no guarantee of convergence, though in all our test cases it the probabilities seems to be converging.[10] However, we need to set a maximum number of iterations as well as a convergence stopping criterion (e.g., based on how much the probabilities change from one iteration to the next).[11]

This probabilistic propagation is more satisfying intuitively as it takes into account probabilities, and can further be bootstrapped by assigning priors to unknown examples. These priors can simply be class priors, or could be estimated by other learning algorithms [21]. We ran pRN* on the three case studies, using the same methodology as reported above for RN*, to see how well this probabilistic version would stand up to the more simple class-propagating classifier.

Surprisingly, pRN* generally performed worse or—at best—only comparably to RN*. In virtually all tests, when only a small fraction ($\leq 30\%$)) of data was initially labeled, RN* performed better than pRN*, though they often had similar performance when we label $> 75\%$ of the data. In only two instances did pRN* outperform RN*. In the CoRA domain, while pRN* is initially worse, it does end up outperforming both RN* as well as the PRM. Figure 3 shows their comparative performances.

The other case where pRN* outperformed RN* was in the WebKB study using only directly neighboring pages, where it was able to achieve an AUC of 0.468 (variance of 0.018) whereas RN* only got an AUC of 0.310—though both are worse than random. However, when we went to a path length of 2, the two classifiers had equivalent performance (same mean AUC and variance).

5 pRN on Synthetic Data

Although pRN* did not perform as well as RN* on the previous data sets, there are cases where it should perform better. In particular, it seems likely that if there is a large class skew or if very few instances are known, then pRN would benefit from being able to propagate probabilities rather than class labels.

We test this hypothesis on a data set where there is a large class skew, where we varied the amount of initially labeled instances. We had access to a synthetic data set—created by others, and not with this study in mind—of people and their interactions, where people were categorized as good or bad depending on whether they belonged to any bad group. The data set consisted of 306 people (each belonging to one or more of 12 groups), 8 of whom belonged to the one bad group. We had $63,602$ interaction links between these people ($23,350$ links from $19,251$ 2-way communication events and $40,252$ from 3842 n-way communication events).

[10] For a simple case where there is no convergence, consider a two-class problems with two instances linked to each other, one having a prior of 1 for class A and the other having a prior of 1 for class B. In this case, they would just keep alternating their beliefs between A and B.

[11] In all our test cases, probabilities were still changing slowly even after 100 iterations, which was the maximum number of iterations we set. However, even a few iterations are generally enough to get estimates that seem to be converging.

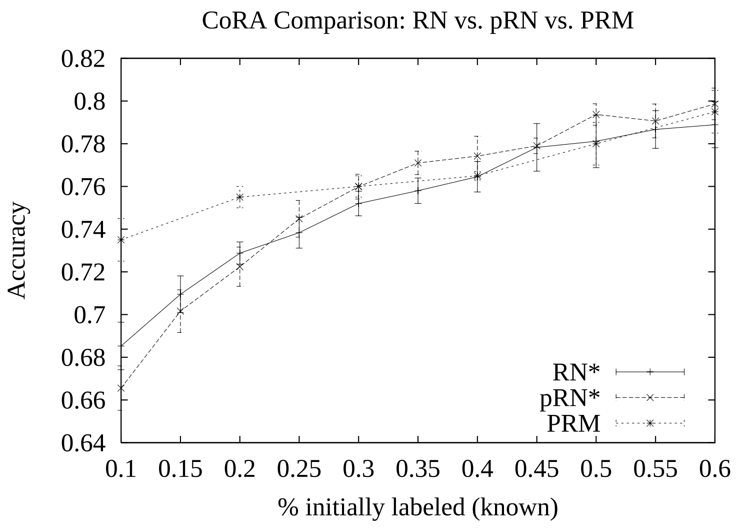

Fig. 3. CoRA Comparison

We have 2 dimensions along which we can test this data set—how many good people are initially known as well as how many bad people are known. We tested our classifiers labeling 0.1%, 1% and 5% of the good people, and labeling $0, .., 7$ of the bad people (in one extreme we do not know *any* bad person, and in the other extreme we know all but one of the bad people)—giving us a total of 24 possible test scenarios. Each test scenario was run 10 times, where we randomly sampled the appropriate number of good and bad people—we pooled the predictions of all runs for a given scenario and created an overall AUC score for that scenario.

Known	0.1% **good**		1% **good**		5% **good**	
# bad	**RN***	**pRN***	**RN***	**pRN***	**RN***	**pRN***
0	0.500	0.836	0.613	0.842	0.506	0.937
1	0.822	0.999	0.873	0.999	0.939	0.999
2	0.746	1.000	0.911	0.995	0.988	1.000
3	0.714	1.000	0.924	0.999	0.952	0.999
4	0.654	0.999	0.885	0.999	0.999	0.999
5	0.660	0.995	0.899	0.998	0.987	0.999
6	0.672	0.990	0.941	0.999	0.987	1.000
7	0.787	0.994	0.947	0.998	0.999	0.998

Table 3. AUCs of RN* and pRN* on a synthetic data set.

Table 3 shows the resulting AUCs for the RN and pRN classifier, where the columns are paired by how many good people are known, and the rows represent the number of bad people initially known. It is clear from these results that pRN* is able to use very little information (e.g., one bad guy and only one good guy—0.1% good) to virtually perfectly label the remaining bad guys. Even when no bad guys are known, pRN* is still able to perform very well. RN* has obvious problems if it doesn't have at least one labeled instance of each class. Further, it needs to have 5% of the good guys labeled before it can perform comparably to pRN*.

6 Final Remarks

We started by observing that although there recently has been much work in the area of relational learning, very little work had been done on creating baseline studies with relational structure in mind. In fact, published results generally compare against non-relational learners as the comparative baseline. We put forth a simple relational neighbor classifier as one potential baseline relational model, which uses no learning and considers nothing but known class labels of related entities.

We performed three comparative studies on reported results for three probabilistic relational learners on three relational data sets (CoRA, IMDb, and WebKB). In all three studies we were able to perform comparably to the relational learners using no learning—though we did use feature selection to improve on upon an already comparable performance for the IMDb data set. If we assume that the experimental designs were comparable, these results provide strong evidence that simple models should receive more attention. Specifically, we feel this makes a strong case for how information can be simply extracted and effectively used from class labels of instances in the immediate neighborhood of an unknown instance. It also makes a strong case for using such simple models for baseline comparisons to assess how well the more complex learners are able to learn from the relational data. One important point was raised in the WebKB study—what to do if the study uses a holdout set that is completely separate from the test set. In this case, we were still able to quantify the performance of the learners by identifying how much of the test set RN needed in order to perform comparably—in the WebKB data set, this turned out to be on the order of 5%.

We also proposed a probabilistic version of RN, though replicating the three case studies showed that it at best performed comparably to the non-probabilistic version, often performing quite worse. While this seemed to indicate that the probabilistic version was not as powerful, a test case on a synthetic data set showed that a probabilistic version does work better under certain conditions, such as a large class skew or having very few labels known initially. Further, a probabilistic version has the added benefit that it can easily take into account estimated probabilities—provided either by the class priors or other learning methods—of unknown instances. The use of learning methods to generate priors warrants further investigation.

Acknowledgments

Thanks to Claudia Perlich for many helpful discussions. David Jensen made many helpful suggestions, including pointing us to the WebKB data set. We thank Ben Taskar and

Andrew McCallum for providing us with versions of the Cora data set. Jennifer Neville was instrumental in our recreating the test environments and setup for the IMDb and WebKB studies. This work is sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-01-2-585. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA), the Air Force Research Laboratory, or the U.S. Government.

References

1. A. Bernstein, S. Clearwater, and F. Provost. The Relational Vector-space Model and Industry Classification. Working paper CDeR IS-03-02, Stern School of Business, New York University, 2003.
2. P. M. Blau. *Inequality and Heterogeneity: A Primitive Theory of Social Structure.* New York: Free Press, 1977.
3. S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *SIGMOD*, 1998.
4. M. Craven, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and C. Y. Quek. Learning to Extract Symbolic Knowledge from the World Wide Web. In *15th Conference of the American Association for Artificial Intelligence*, 1998.
5. P. Domingos and M. Pazzani. Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. In *Proceedings of the 13th International Conference on Machine Learning*, pages 105–112, 1996.
6. S. Dzeroski and N. Lavrac. *Relational data mining.* Berlin; New York: Springer, 2001.
7. W. Emde and D. Wettschereck. Relational Instance-Based Learning. In L. Saitta, editor, *Proceedings 13th International Conference on Machine Learning*, pages 122–130. Morgan Kaufmann, 1996.
8. T. Fawcett. ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. Technical Report HPL-2003-4, HP Labs, 2003.
9. P. A. Flach and N. Lachiche. 1BC: A First-Order Bayesian Classifier. In S. Dzeroski and P. A. Flach, editors, *Ninth International Workshop on Inductive Logic Programming (ILP'99)*, volume 1634, pages 92–103. Springer-Verlag, June 1999.
10. N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning Probabilistic Relational Models. In *Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, 1999.
11. D. Jensen. Quantitative criteria to characterize kd-ml research for c-xnt. 1998.
12. D. Jensen and J. Neville. Data Mining in Social Networks. In *National Academy of Sciences workshop on Dynamic Social Network Modeling and Analysis*, 2002.
13. D. Jensen and J. Neville. Linkage and Autocorrelation Cause Feature Selection Bias in Relational Learning. In *Nineteenth International Conference on Machine Learning (ICML2002)*, 2002.
14. D. Jensen and J. Neville. Schemas and Models. In *Proceedings of the Workshop on Multi-Relational Data Mining (MRDM-2002)*, pages 56–70, 2002.
15. T. Joachims. A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In D. H. Fisher, editor, *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 143–151, Nashville, US, 1997. Morgan Kaufmann.

16. D. Koller and A. Pfeffer. Probabilistic Frame-Based Systems. In *AAAI/IAAI*, pages 580–587, 1998.
17. A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval*, 3(2):127–163, 2000.
18. M. McPherson, L. Smith-Lovin, and J. M. Cook. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology*, 27:415–444, 2001.
19. T. Mitchell. *Machine Learning*. McGraw Hill, 1997.
20. J. Neville. Personal communication, June 2003.
21. J. Neville and D. Jensen. Iterative Classification in Relational Data. In *AAAI Workshop on Learning Statistical Models from Relational Data*, pages 13–20, 2000.
22. J. Neville, D. Jensen, L. Friedland, and M. Hay. Learning Relational Probability Trees. Technical Report 02-55, Department of Computer Science, University of Massachusetts Amherst, 2002. Revised version February 2003.
23. J. Neville, D. Jensen, B. Gallagher, and R. Fairgrieve. Simple Estimators for Relational Bayesian Classifiers. Technical Report 03-04, Department of Computer Science, University of Massachusetts Amherst, 2003.
24. J. Pearl. *Probabilistic Reasoning in Intelligent Systems.* Morgan Kaufmann, 1998.
25. F. J. Provost, C. Perlich, and S. A. Macskassy. Relational Learning Problems and Simple Models. In *Proceedings of the Relational Learning Workshop at IJCAI-2003*, 2003.
26. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.
27. B. Taskar, E. Segal, and D. Koller. Probabilistic Classification and Clustering in Relational Data. In *17th International Joint Conference on Artificial Intelligence*, pages 870–878, 2001.