LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

UCRL-TR-231993

Leveraging Network Structure to Infer Missing Values in Relational Data

Brian Gallagher, Tina Eliassi-Rad

June 20, 2007

Disclaimer

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

Leveraging Network Structure to Infer Missing Values in Relational Data

Brian Gallagher and Tina Eliassi-Rad
Center for Applied Scientific Computing
Lawrence Livermore National Laboratory
Box 808, L-560, Livermore, CA 94551
{bgallagher, eliassi}@llnl.gov

Abstract

Inference techniques for relational data improve classification performance by exploiting dependencies between attributes of related instances. In particular, a great deal of recent attention has been paid to collective inference procedures, which make simultaneous inferences over attributes of related instances. Collective inference has been shown to be particularly effective for overcoming substantial amounts of missing attribute information. We propose a novel approach for inference in relational data, which leverages information about the relational network structure. We show that when structural characteristics are informative, our approach leads to consistent, and sometimes dramatic, improvement in classification performance regardless of the amount of attribute information available. We demonstrate the utility of our method on several real-world classification tasks. Interestingly, for many of these tasks, collective inference does not perform well, apparently due to low amounts of relational autocorrelation. Understanding data characteristics that influence collective inference is a largely unexplored area for further study.

1. Introduction

Traditional inference techniques utilize dependencies between attributes of a single data instance to make predictions. For example, we might use the text of book *A* to infer *A*'s political orientation (see Figure 1). Inference techniques for relational data improve classification performance by taking advantage of the dependencies between attributes of related instances. For instance, if we know the political orientation of other books purchased by owners of book *A*, this gives us further information about *A*'s orientation. In this paper, we explore another potential source of information present in relational data: namely, the structure of the network formed by the relationships between individual data instances.

Figure 1: Co-purchase network of books about U.S. politics. Nodes represent books purchased from an Internet bookseller. Links indicate that two books were purchased by the same consumer. Each book in this data set is labeled either 'liberal', 'conservative', or 'neutral' [9]. Liberal books appear along the left periphery of the graph, conservative books along the right periphery, and neutral books in the center.

The basic problem we address is as follows. We are given a network, consisting of attributed nodes and links (e.g., book nodes and co-purchase links). For some attribute of interest, certain nodes have a known value and for other nodes, the value is unknown (e.g., a book's political orientation is missing). The goal is to infer the unknown values for the attribute of interest. We treat this as a supervised learning problem, where we use the labeled nodes to train a classifier and then apply this classifier to predict the values of the missing

attribute values. Missing labels are common in real-world applications (e.g., due to imperfect text extractors or simply unknown information).

Relational learning approaches generally take advantage of dependencies between the class labels (or other attributes) of related instances (i.e., relational autocorrelation [7]). However, there is another potentially rich source of information in the network that has been largely overlooked – namely, the dependence between attribute values and structural characteristics of the graph. It is apparent from Figure 1 that there are important differences between the structural characteristics of the 'neutral' political books (in the center of the graph) and those of the 'conservative' and 'liberal' books (around the periphery of the graph). Among these differences: neutral books have fewer neighbors, non-neutral books have more tightly clustered neighborhoods, and neutral books are more central to the network (i.e., without them there is no connection between the large liberal cluster and the large conservative cluster). We aim to take advantage of these differences by explicitly modeling structural features of networks in addition to commonly used attribute-based features.

The main contributions of this paper are as follows:

- We introduce the relational random forest model, an extension to random forests that models (1) dependencies between attributes of related instances and (2) dependencies between attributes and graph structure.
- We demonstrate that modeling relational dependencies based on network structure can dramatically improve classification performance when (1) network structure provides information not captured by attributes and/or (2) predictive attribute values are missing.
- We observe that collective inference procedures do not perform well on a number of our chosen classification tasks due to low levels of relational autocorrelation. We demonstrate how features of these classification tasks account for this low autocorrelation and suggest that a large number of interesting classification tasks on relational data may suffer from low autocorrelation due to a combination of class skew and link density.

The remainder of this paper is organized as follows. Section 2 describes our approach. Sections 3, 4, and 5, respectively, present related work, our experimental study, and detailed discussion. We conclude the paper in Section 6.

2. The Relational Random Forest

In order to demonstrate the utility of structural graph characteristics, we need a model that can incorporate both attribute-based and structural information. For this purpose, we introduce the relational random forest (RRF) model. The RRF consists of two components: (1) a relational feature constructor and (2) a standard random forest model [1] for learning and inference. The purpose of feature construction is to produce a set of features that capture the various dependencies preset in relational data. In particular, we want to be able to capture dependencies of the class label on local and relational attributes and on network structure. Once we construct relational features from our data set, we pass these feature vectors to a standard random forest model for learning and inference. We chose random forests as a way to combine information from attribute-based and structural features because of their ability to make sense of large numbers of features, irrelevant features, multiple correlated features, and so on.

The following sections describe our approach to relational feature construction and the random forest model in more detail. We also describe the approach to collective inference we use for the RRF model.

2.1. Relational Feature Construction

The relational random forest model constructs a variety of features based on both the attribute and structural information available in the network. We define a feature to be a function of the network observables (i.e., known attribute values and network structure). Figure 2 presents a simple taxonomy of relational features. At the top level, we separate features into attribute-based and structural.

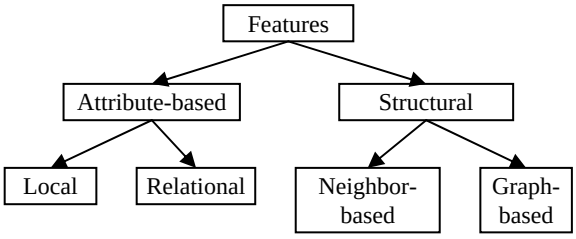


Figure 2: Taxonomy of features in relational data

Attribute-based features are further divided into: (1) local features, which are intrinsic attributes of a node or a link (e.g., Person.name) and (2) relational features, which are calculated by applying aggregation functions to the set of attribute values of neighboring nodes or links (e.g., mode(NeighborPersons.title)). We use

mode, *count*, and *proportion* to aggregate values of categorical attributes and *mean*, *min*, and *max* to aggregate values of numerical attributes. Aggregations over node attributes are applied both to unique nodes as well as weighted by the number of links to each node. In addition, each aggregation is applied to incoming links only, outgoing links only, and all links.

Structural features are divided into: (1) neighbor-based features that provide information about the structure of the immediate neighborhood (e.g., neighborCount(Person)), and (2) graph-based features, which leverage information on the structure of a more extended neighborhood, which may even include the entire network (e.g., betweenness(Person)).

For neighbor-based structural features, we use the number of neighboring nodes and number of incident links. Note that in multigraphs, these two values are different. For graph-based structural features, we use betweenness centrality (which identifies nodes/links that occur along many paths) and clustering coefficient (which measures neighborhood strength in terms of how connected nodes in a neighborhood are to one another). We formally define betweenness centrality and clustering coefficient next. For more details, we encourage the reader to see [16].

Betweenness centrality can be defined for nodes or links. For node betweenness, we compute the following function:

$$bet_i = \frac{1}{\frac{1}{2}N(N-1)} \sum_{s \neq i \neq t \in V} \frac{g_i(s,t)}{N_{st}}$$

where $g_i(s; t)$ is the number of shortest paths from node s to node t that pass through node i. N_{st} is the total number of geodesic paths from s to t. V is the set of nodes in the network and N is the total number of nodes (i.e., $N = |V|$). A node with high betweenness has great influence over what information flows in the network.

Clustering coefficient for a node i is defined as

$$C(i) = \frac{E_i}{k_i(k_i - 1)}$$

where k_i is the number of neighbors of node i and E_i is the number of edges between the k_i nodes. Within social networks, the clustering coefficient captures the common belief that a friend of a friend is also a friend.

2.2. Random Forest Models

The random forest model was introduced by Breiman in 2001 [1]. A random forest consists of a collection of decision trees, each of which is trained on a random subset of the training examples using a randomly selected subset of the available features. Inference is performed independently by each tree and the votes are combined to obtain a final probability of each class.

Random forests have been shown to perform well in practice on a variety of learning tasks. They can sort through a large number of features extremely well. They are also computationally efficient and easily parallelizable.

2.3. Collective Inference

Collective inference (a.k.a. collective classification) works by simultaneously inferring the values of a set of related labels (e.g., the political orientation of a set of books linked by co-purchases). The inference process can be viewed as a message passing algorithm, where each round consists of a set of messages being passed between a node and its neighbors. There are different procedures for performing collective inference. The most popular include iterative classification, mean-field relaxation labeling, loopy belief propagation, and Gibbs sampling. Sen and Getoor [20] and Macskassy and Provost [12] both provide empirical studies of these methods.

Like other conditional models for attribute prediction in relational data, the RRF can use collective inference to exploit long range dependencies between attributes of related instances. For this study, the RRF implements collective inference using the iterative classification algorithm. See section 4.5 for the specifics of the algorithm.

3. Related work

In recent years, there has been a great deal of work on models for learning and inference in relational data [6, 10, 11, 14, 15]. Many use some sort of feature construction to incorporate attribute-based relational information. However, to our knowledge, no previous approach uses structural information from the extended neighborhood for attribute prediction. Table 1 summarizes the common attribute-based features used by existing models.

Note that most of these models use only a single aggregation function at a time. Relational Probability Trees (RPTs) [14] use several features concurrently. However, they construct only binary features. RPTs also use neighbor-based structural features (specifically, a node's degree), but they do not use graph-based structural features such as betweenness or clustering coefficient. The relational random forest allows for the simultaneous use of multiple types of features: attribute-based, structural, temporal, etc.

Table 1: A list of functions used in existing models to aggregate attribute-based relational features. The abbreviations refer to the following. PRM: Probabilistic Relational Models [6]; RPT: Relational Probability Trees [14]; RBC: Relational Bayesian Classifiers [15]; LB: Link-Based Classifiers [10]; RN: Relational Neighbor Classifiers [11].

	Existing Models				
	PRM	**RPT**	**RBC**	**LB**	**RN**
Mode	✓	✓	✓	✓	
Mean	✓	✓	✓		
Min		✓			
Max		✓			
Count		✓		✓	
Prop†		✓			✓
Rand†	✓	✓	✓		

Whereas we use structural network characteristics directly as features for classification, Rattigan et al. [18] use network structure to decide which nodes to label in an active learning setting. Their use of network structure is complimentary to ours. It may be possible to gain additional benefit from a combination of the two approaches. In addition, we may be able to take advantage of their faster, approximate calculations of network measures like betweenness.

Singh et al. [21] use descriptive attributes and structural properties to prune a network down to its 'most informative' affiliations and relationships for the task of attribute prediction.

Perlich and Provost [17] provide a nice hierarchy for aggregation of values of attributes of related instances. However, they do not consider structural features.

There are many recent papers on collective inference [2, 8, 12, 13, 19, 20]. In this group, Sen and Getoor [20] provide a nice empirical study of the various procedures for collective inference. Macskassy & Provost [12] provide a nice case-study of previous work in learning attributes of networked data.

4. Experimental Evaluation

This section describes our experiments on a variety of tasks. These include a comparison between RRF and other models for relational classification, an assessment of how much attribute-based features, structural features, and their combination contribute to the overall classification performance, and finally a study of the effects of collective inference on our tasks.

4.1 Data Sets

We present results on three real-world data sets: political book purchases [9], Enron emails [3], and Reality Mining cell phone calls [4].

The political books data set consists of 105 books labeled as liberal, conservative, or neutral. Links between books indicate that both books were purchased by the same customer. There are 441 co-purchase links in this data set. Our task is to identify the neutral books (Pr(*neutral*) ≈ 0.12).

From the Enron data set, we use a subset containing all data collected during a 32 day period, from 6/8/2001 to 7/10/2001. This subset consists of approximately 9K people nodes and 57K email links. We explore two prediction tasks in this data set. The first is to identify executives among Enron employees (Pr(*exec*) ≈ 0.015). For this task, we use the subset of nodes for which we have ground truth, which yields 1.6K nodes and 6.5K links.

The second task is to identify Enron employees. For this task, we use a continuous subgraph of the 32-day temporal email network described in the previous paragraph, consisting of 1K nodes and 14K links (Pr(*enron*) ≈ 0.76). This subgraph was obtained by starting from a random node in the graph and expanding out in a breadth-first fashion until 1000 nodes had been touched. The final subgraph includes all nodes and links touched during the breadth-first search. Note that people who do not work at Enron get pulled into the graph by sending email to or receiving email from an Enron employee.

For the Reality Mining data set, we also use a continuous subgraph, again obtained via breadth-first sampling. This subgraph consists of approximately 1K people nodes and 32K phone call links. Our task in the Reality Mining data is to identify which of the people in the phone call network are study participants (Pr(*study*) ≈ 0.084).

Table 2 shows a number of statistics on our various prediction tasks: relational autocorrelation [7] (which is a measure of correlation between the class labels of neighboring instances) and Pearson's correlation coefficient between class label and (a) proportion of neighbors of positive class, (b) betweenness, (c) clustering coefficient, (d) number of incident links, and (e) number of neighboring nodes.

† Prop is short for proportion. Rand denotes stochastic mode.

Table 2: Statistics on prediction tasks

	Political Books	Enron Execs	Enron Empl.	Reality Study
Relational autocorrelation	0.166	0.222	0.023	-0.856
Correlation with class label				
Prop. neighbors w/ positive class	-0.414	-0.029	0.037	-0.878
Betweenness	0.093	-0.105	0.130	0.216
Clustering Coefficient	-0.002	0.013	0.137	0.025
# incident links	-0.176	-0.065	0.150	0.414
# neighbor nodes	-0.176	-0.079	0.251	0.447

4.2 Methodology

For all results presented here, the basic experimental setup is the same. In all cases, classifiers have access to the entire data graph during both training and testing. However, not all nodes in the graph are labeled. We vary the proportion of labeled nodes from 10% to 90%. Classifiers are trained on all labeled nodes and evaluated on all unlabeled nodes.

Our methodology is essentially the same as the one used by Macskassy and Provost [12] for their study of within-network classification, except that we ensure that each instance in the data set is given equal weight in the overall evaluation. For each proportion labeled, we run 20 trials. For each trial and proportion labeled, we choose a class-stratified random sample containing (1.0 – proportion labeled)% of the total instances as the test set and the remaining instances become the training set. Note that for proportion labeled less than 0.9 (or greater than 10 trials), this means that a single instance will necessarily appear in multiple test sets. As Macskassy and Provost note, the test sets cannot be made to be independent because of this overlap. However, we carefully choose the test sets so as to ensure that each instance in our data set occurs in the same number of test sets over the course of 10 trials. This ensures that each instance carries the same weight in the overall evaluation regardless of the proportion labeled. Labels are kept on the training instances and removed from the test instances. We use identical train/test splits for each classifier.

4.3 Comparison with Existing Models for Relational Classification

We compare the RRF model with a number of existing models for relational classification: Weighted-Vote Relational Neighbor classifier (wvRN) [11, 12], Nework-Only Link-Based classifier (nLB) [10, 12], Relational Bayesian Classifier (RBC) [15], and Relational Probability Tree (RPT) [14]. We use the Proximity implementation[1] of the RBC and RPT and our own implementation of the wvRN and nLB classifiers based their descriptions. For the relational random forest (RRF) model, we use the R implementation[2] of the random forest model. We use the default parameters for the R random forest (including a forest of 500 trees). We do not use collective inference in this set of experiments (see Section 4.3 for collective inference results).

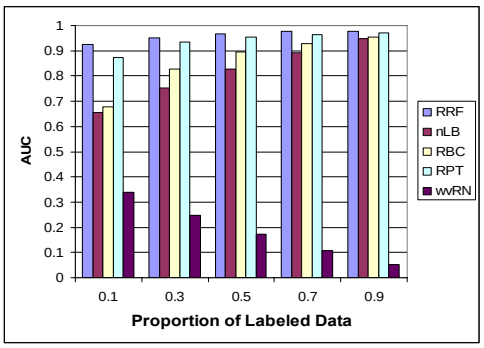
The results of this experiment are summarized in figure 3. We use the area under the Receiver Operating Characteristic (ROC) curve (AUC) to compare the results. We chose AUC because most of our tasks have a class-skew problem and, therefore, all methods achieve close to default accuracy. Figure 6 shows the p-values from paired t-tests.

Figure 3 shows that the RRF model performs as well as, and often much better than, the other models. This effect is more pronounced as the proportion of known labels decreases. The one exception is that wvRN outperforms RRF with only 10% of data labeled on the political book task. This likely results from a lack of training examples due to the small size of this data set (train set size ≈ 10 at 10% labeled). Note that wvRN is not affected by the amount of training data, since it is not a learning method. See section 5 for more on this.

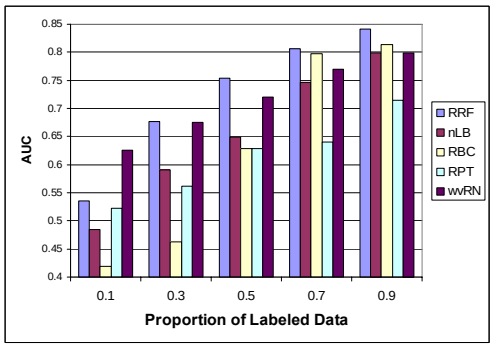
4.4 Modeling Attributes and Structure

Figure 4 shows the classification performance of RRFs with (1) attribute-based features only, (2) structural features only, and (3) the combination of both. The goal here is to tease out the contributions of each feature type. Again we do not use collective inference in this set of experiments. Figure 6 shows the p-values from paired t-tests.

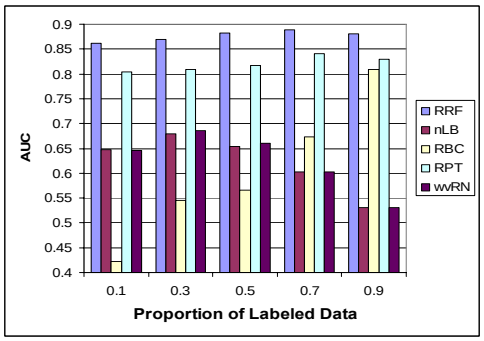
There are several things to note regarding figure 4. First, we see that, at high levels of labeled data, the relative performance of attribute-based and structural features on their own varies across tasks, but the two are generally comparable in their predictive power. However, as the proportion of unknown labels increases, the predictive power of the structural features remains relatively consistent, while the power of the attribute-based features declines dramatically. Over all levels of labeling, the combination of attribute-based and structural features generally results in increases in performance over either on its own.



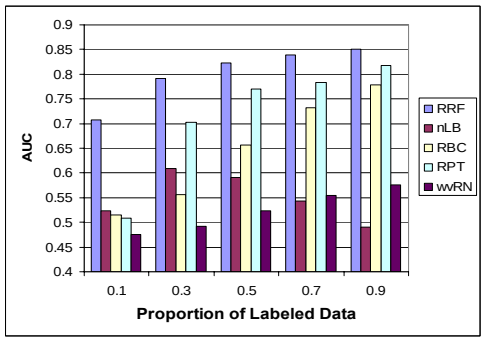
(a) Is person *X* in Reality Mining study?



(b) Is book *X* a neural political book?

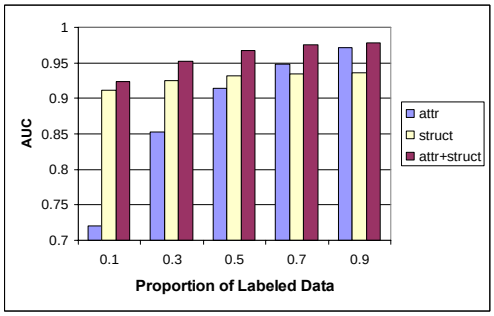


(c) Is person *X* an Enron employee?

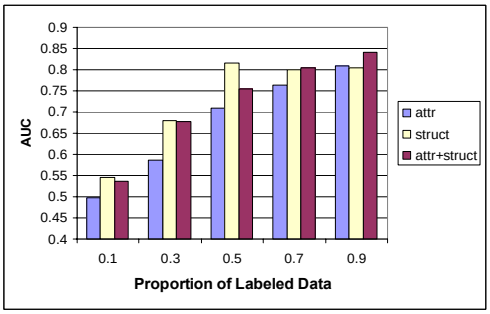


(d) Is person *X* an Enron executive?

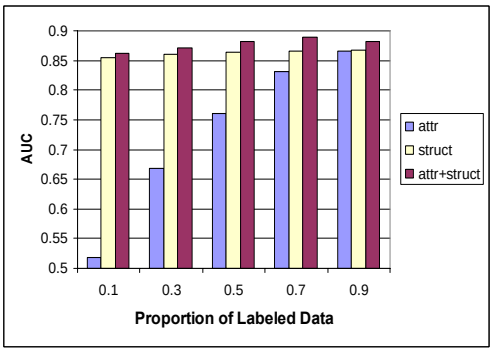
Figure 3: Baseline comparison of RRF to other models for relational classification



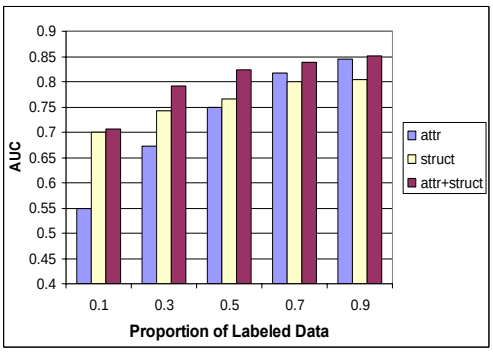
(a) Is person *X* in Reality Mining study?



(b) Is book *X* a neural political book?

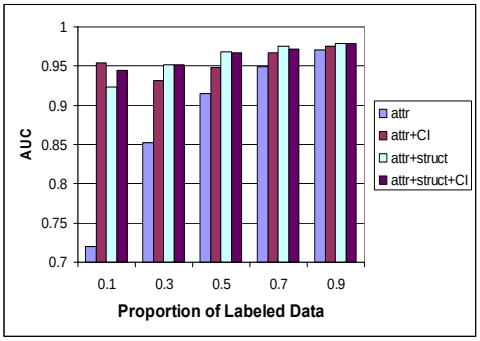


(c) Is person *X* an Enron employee?

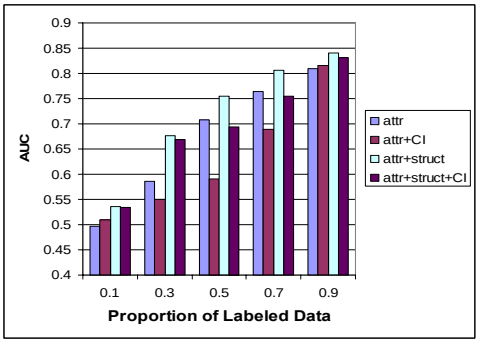


(d) Is person *X* an Enron executive?

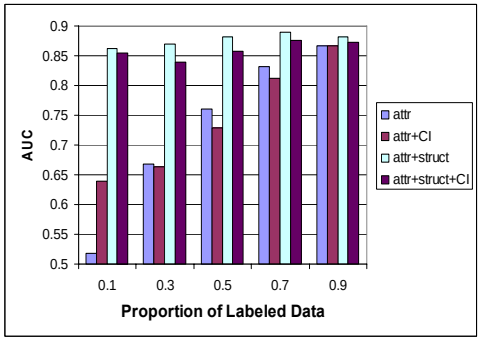
Figure 4: Contributions of attributes, structure, and their combination on classification performance



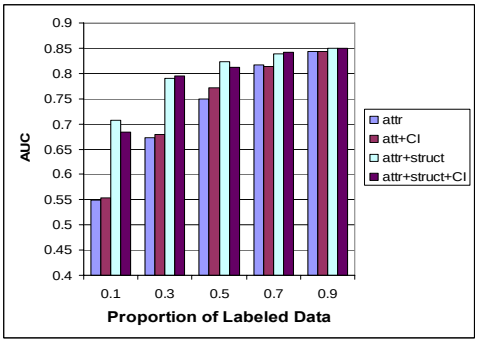
(a) Is person *X* in Reality Mining study?



(b) Is book *X* a neural political book?



(c) Is person *X* an Enron employee?



(d) Is person *X* an Enron executive?

Figure 5: Effects of collective inference

	Proportion of Data Labeled				
Political books task	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**
attr+struct > nLB	**0.036**	**0.011**	**0.002**	0.085	0.123
attr+struct > wvRN		0.478	0.156	0.103	0.123
wvRN > attr+struct	**0.001**				
attr+struct > RBC	**0**	**0**	**0.012**	0.381	0.233
attr+struct > RPT	0.291	**0**	**0**	**0**	**0.001**
struct > attr	**0.04**	**0.013**	**0**	**0.039**	0.528
attr+struct > attr	**0.011**	**0**	**0.014**	**0.008**	0.107
attr+struct > attrStruct	0.439	0.707	0.997	0.72	0.567
struct > attr+struct	0.283	0.462	**0.007**	0.587	0.775
attr+CI > attr	0.336	0.882	0.997	0.996	0.348
attr+struct+CI > attr+struct	0.549	0.654	0.985	0.996	0.661
attr > attr+CI	0.664	0.118	**0.003**	**0.004**	0.652
attr+struct > attr+struct+CI	0.451	0.346	**0.015**	**0.004**	0.339
attr+struct > attr+CI	0.122	**0**	**0**	**0**	0.13
attr+CI > attr+struct	0.878	1	1	1	0.87

	Proportion of Data Labeled				
Enron executive task	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**
attr+struct > nLB	**0**	**0**	**0**	**0**	**0**
attr+struct > wvRN	**0**	**0**	**0**	**0**	**0**
attr+struct > RBC	**0**	**0**	**0**	**0**	**0.041**
attr+struct > RPT	**0**	**0.001**	**0.031**	**0.024**	0.083
attr > struct	1	0.987	0.694	0.183	**0.049**
struct > attr	**0**	**0.013**	0.306	0.817	0.951
attr+struct > attr	**0**	**0**	**0.001**	**0.038**	0.323
attr+struct > struct	0.186	**0**	**0.002**	**0.001**	**0.007**
attr+CI > attr	0.392	0.295	0.061	0.655	0.673
attr+struct+CI > attr+struct	0.946	0.37	0.905	0.371	0.433
attr > attr+CI	0.608	0.705	0.939	0.345	0.327
attr+struct > attr+struct+CI	0.054	0.63	0.095	0.629	0.567
attr+struct > attr+CI	**0**	**0**	**0.003**	**0.03**	0.319
attr+CI > attr+struct	1	1	0.997	0.97	0.681

	Proportion of Data Labeled				
Reality Mining task	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**
attr+struct > nLB	**0**	**0**	**0**	**0**	**0.007**
attr+struct > wvRN	**0**	**0**	**0**	**0**	**0**
attr+struct > RBC	**0**	**0**	**0**	**0.005**	0.098
attr+struct > RPT	**0**	**0.004**	**0.004**	0.056	**0.046**
attr > struct	1	0.998	0.817	0.17	**0.005**
struct > attr	**0**	**0.002**	0.183	0.83	0.995
attr+struct > attr	**0**	**0**	**0.001**	**0.008**	0.136
attr+struct > struct	**0.018**	**0**	**0**	**0**	**0**
attr+CI > attr	**0**	**0**	**0.001**	**0.039**	0.127
attr+struct+CI > attr+struct	**0.001**	0.606	0.731	0.961	0.108
attr > attr+CI	1	1	0.999	0.961	0.873
attr+struct > attr+struct+CI	0.999	0.394	0.269	**0.039**	0.892
attr+struct > attr+CI	0.992	**0.003**	**0.008**	0.072	0.147
attr+CI > attr+struct	**0.008**	0.997	0.992	0.928	0.853

	Proportion of Data Labeled				
Enron employee task	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**
attr+struct > nLB	**0**	**0**	**0**	**0**	**0**
attr+struct > wvRN	**0**	**0**	**0**	**0**	**0**
attr+struct > RBC	**0**	**0**	**0**	**0**	**0**
attr+struct > RPT	**0**	**0**	**0**	**0**	**0**
struct > attr	**0**	**0**	**0**	**0.002**	0.435
attr+struct > attr	**0**	**0**	**0**	**0**	**0.001**
attr+struct > struct	**0.009**	**0.026**	**0.001**	**0**	0.052
attr+CI > attr	**0.016**	0.528	0.777	0.816	0.464
attr+struct+CI > attr+struct	0.888	0.946	0.969	0.999	0.981
attr > attr+CI	0.984	0.472	0.223	0.184	0.536
attr+struct > attr+struct+CI	0.112	0.054	**0.031**	**0.001**	**0.019**
attr+struct > attr+CI	**0**	**0**	**0**	**0.003**	0.057
attr+CI > attr+struct	1	1	1	0.997	0.943

Figure 6: p-values from paired t-tests. Statistically significant differences (p-values ≤ 0.05) appear in bold.

4.5 Effects of Collective Inference

To perform collective inference, we use the iterative classification algorithm described by Macskassy and Provost [12] and cap the number iterations at 10. The algorithm converges and terminates in fewer than 10 iterations for about 93% to 100% of trials, depending on the experiment. The situations where the algorithm has not converged after 10 iterations typically involve a small number of labels (≤5) changing on each of the last few iterations. Macskassy and Provost [12] and Sen and Getoor [20] both report similar observations regarding the convergence speed of iterative classification. We also tried Gibbs sampling [5] with up to 2000 iterations, which yielded comparable results. We ultimately chose iterative classification because (1) it is simple, (2) it has been shown to have consistently good performance on a variety of collective classification tasks, and (3) it converges more quickly than other approaches.

Figure 5 depicts the effects of collective inference on the classification performance (1) with attribute-based features, (2) with attribute-based features and collective inference, (3) with attribute-based and structural features, and (4) with attribute-based and structural features plus collective inference. As before, Figure 6 shows the p-values from paired t-tests.

Figure 5 shows that the benefit of incorporating structural information on these tasks is generally much greater than the benefit of using collective inference. Collective inference provides a significant benefit for all levels of known labels < 90% on the Reality Mining task and at 10% labeled on the Enron employee task. However, in all other cases, the use of collective inference provides no significant benefit and, in some cases, significantly hurts performance. The use of structural features provides a significant benefit on all tasks for all levels of known labels < 90%. On the Enron employee identification task, structural features significantly improve performance across the range of labeled data proportions.

5. Discussion

On the Reality Mining task, RPT's performance is close to RRF (although RRF performs significantly better, except at 70% labeled). This is likely due to the high correlation of the class with the neighbor-based features (see Table 2). Recall that the RPT uses the number of neighbors as a feature. Here, the graph-based structural features appear to be less important.

The performance of wvRN on the Reality Mining task is extremely poor because of the negative autocorrelation is this data set (Table 2). Note that wvRN is not a learning technique. It assumes that there is positive relational autocorrelation and it cannot take advantage of other patterns of dependence between attributes of related instances. In particular, the wvRN model just looks at the proportion of neighbors that have each class and uses these proportions as the probabilities it returns. So, if 9/10 of the neighbors are in the Reality Mining study and 1/10 are not in the study, wvRN returns Pr(*study*) = 0.9 and Pr(~*study*) = 0.1. However, for this particular task, since there is actually negative autocorrelation between the class labels, having a large number of in-study neighbors actually indicates that you are NOT in the study. The nLB model also uses the proportion of neighbors of each class as features. However, since it learns from the training data (using a logistic regression model) and does not just assume that the autocorrelation is positive, it is able to figure out the correct (in this case, inverse) relationship between neighboring class labels.

It is also interesting to note the cause of the negative autocorrelation in the Reality Mining task. The network here is of phone calls made and received by participants in a study who are essentially all co-workers (faculty, staff, and students at MIT). It makes sense that co-workers do not generally call each other on their cell phones since they have many other means of communication available (e.g., email, work phone, face-to-face meetings). Therefore, people in the study generally communicate with people outside of the study. In addition, the calls between people outside of the study are not recorded. Hence, there is very strong negative autocorrelation. This situation is in contrast with the Enron email data set, where Enron employees use email to communicate with people both inside and outside of the company. However, Enron employees will tend to communicate more with other Enron employees than with non-employees. So, in this case, we have positive relational autocorrelation (see Table 2), although the correlation is very weak because non-employees communicate exclusively with employees due to the way the data is collected.

With the exception of the political book task, we see very little performance degradation as the proportion of labeled nodes decreases in the classifiers that incorporate structural information. Note that varying the proportion of labeled data has two effects: (1) it determines the number of instances available during training and (2) it determines the amount of information available from neighboring instances during inference. The quality of structural information

is not affected by (2), but if the amount of training data available is very small (as in the political book data), we will not have enough examples to learn the correct structural dependencies.

The generally mediocre performance of collective inference in this study is likely due to low amounts of relational autocorrelation in all but the Reality Mining task (see Table 2). Our observation that the performance of collective inference tails off as autocorrelation decreases is consistent with the findings of other studies on collective inference (e.g., [8]). This effect is probably exacerbated in the political book data due to the small amount of labeled data available for both learning and inference.

Note that the tasks we consider all have a large imbalance in the relative frequencies of their classes. There are many examples of important real-world prediction tasks like this, where one class is extremely rare (e.g., detecting rare events, fraud, etc). We hypothesize that the class skew present in our tasks may account for the low levels of autocorrelation we observe. The intuition here is that if in a data set 95% of the instances belong to class A and 5% to class B, the B's are going to have a lot harder time finding other B's to link to.

To investigate our hypothesis, we ran a set of experiments where we vary the class skew and observe the effects on autocorrelation levels. For simplicity, we focus here on positive autocorrelation. We used both a uniform 2-D lattice of 1600 (40x40) nodes and the network from the Enron employee identification task and varied the percent of positive instances as follows. We initially labeled all nodes with the negative class. Then we randomly chose a starting node, labeled it positive, and spread out from there in a breadth-first fashion, labeling each node we encountered as positive until we had labeled the specified proportion of positive nodes. For each network and proportion of positive labels, the results were averaged over 10 trials, each with a different, randomly chosen starting node. The results are shown in Figure 7.

We see that for both the 2-D lattice and the Enron graph, the amount of autocorrelation tails off sharply as the class skew approaches the extremes. The general downward trend in autocorrelation as the number of positive instances increases in the Enron graph is due to the set of negative instances being split from a single connected subgraph into many smaller subgraphs, as the positive subgraph increases in size. In the 2-D lattice, the negative instances remain part of a single connected subgraph because of the lattice's regular structure.

Since the prediction of rare events is a common and important application, these results suggest that low autocorrelation in classification problems may be more common that is generally assumed. Understanding the data characteristics that influence relational autocorrelation is an important and largely unexplored area of study. An in-depth investigation into this area is beyond the scope of this work. However, we note that class skew on its own does not explain the amount of observed autocorrelation in the previous experiments and offer another explanation here.

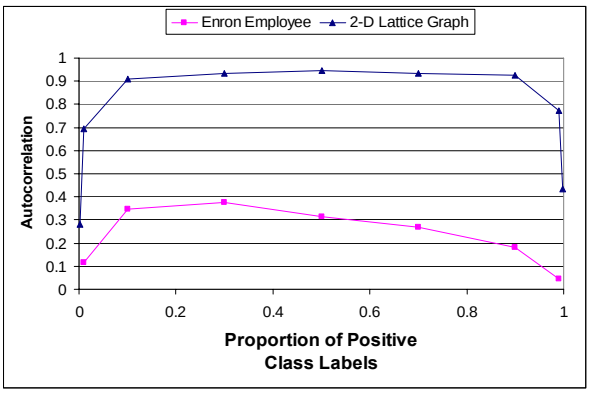


Figure 7: Relationship between autocorrelation and class skew

Judging by the differences in autocorrelation between the 2-D lattice and Enron graph across the range of positive class proportions, differences in graph structure apparently contribute to the amount of autocorrelation observed. We hypothesize that one source of variation is the link density of the networks, although the pattern of linkage likely plays a role as well. One way to measure link density is the average number of neighbors per node (3.9 for the 2-D lattice network and 5.0 for the Enron network). All else being equal, more links per node means more chances for the less prevalent class to link to the more prevalent class. Therefore, we hypothesize that denser linkage will lead to lower autocorrelation. To investigate the relationship between link density and autocorrelation we ran the following experiment. We again generated a network with 1600 nodes, but this time connected in a straight line instead of a lattice. Then we added additional links at random in varying numbers to create varying link densities. We then initialized all nodes to the negative class and then labeled 50% of the nodes as positive using breadth-first search as in our previous experiment. Figure 8 shows how autocorrelation varies with link density, here measured using average node degree. Again, each data point is an average over 10 trials.

We see from these results that, at least in a network with random link structure, there is a strong correlation between link density and autocorrelation. In addition,

we see that the autocorrelation level may be affected dramatically by small changes in link density. For example, in our experiments, an average degree of ~2 corresponds to near perfect autocorrelation (i.e., 0.99), whereas an average degree of ~5 corresponds to an autocorrelation level around 0.3.

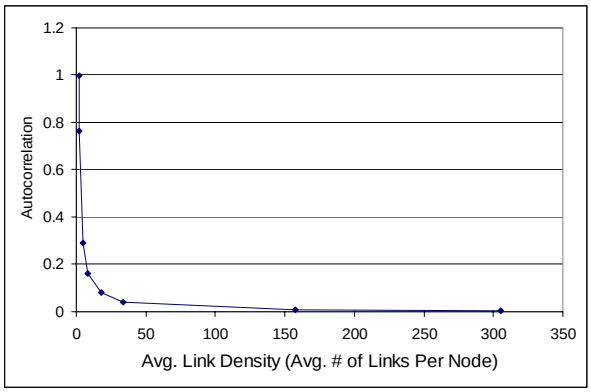


Figure 8: Relationship between autocorrelation and link density

For this study, we have identified several interesting tasks for which structural graph characteristics are predictive of the class label of interest. It remains an open question how often in practice structural features are correlated with attributes in the ways we see here.

6. Conclusions

Attribute dependence is an important source of information for classification in relational data, but relational autocorrelation is not always present in real-world tasks. Network structure can provide a great deal of information whether or not autocorrelation is present. The combination of attribute-based and structural features can provide more information than either on its own. Relational random forests are an effective way to combine attribute-based and structural information for the task of attribute prediction. Structural modeling is a complimentary approach to collective inference in its strengths since each method relies on a different kind of dependence.

Important problems where autocorrelation is low may be more common than is generally assumed, due in part to high levels of class skew. Future work includes the exploration of the effects of network characteristics on collective inference in real-world relational data.

Acknowledgments

This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-ENG-48. UCRL-TR-XXXXXX.

Notes

1. Proximity is available at http://kdl.cs.umass.edu/software/proximity.html.
2. The R implementation of random forest is available at http://cran.r-project.org/src/contrib/Descriptions/randomForest.html.

References

[1] L. Breiman, "Random forests," *Machine Learning*, 45(1), 2001, pp. 5-32.

[2] S. Chakrabarti, B. Dom, and P. Indyk, "Enhanced hypertext categorization using hyperlinks," In *Proc. of ACM SIGMOD Int'l Conf. on Management of Data*, 1998, pp. 307-318.

[3] W.W. Cohen, "Enron email data set," http://www.cs.cmu.edu/~enron/.

[4] N. Eagle and A. Pentland, "Reality mining: sensing complex social systems," *Journal of Personal and Ubiquitous Computing*, 10(4), 2006, pp. 255-268.

[5] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence* (*PAMI*), 6, 1984. pp. 721-741.

[6] L. Getoor, N. Friedman, D. Koller, and B. Taskar, "Learning probabilistic models of link structure," *Journal of Machine Learning Research*, 3, 2002, pp. 679-707.

[7] D. Jensen and J. Neville, "Linkage and autocorrelation cause feature selection bias in relational learning," In *Proc. of the 19th Int'l Conf. on Machine Learning* (*ICML*), 2002, pp. 259-266.

[8] D. Jensen, J. Neville, and B. Gallagher, "Why collective inference improves relational classification," In *Proc. of the 10th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining* (*KDD*), 2004, pp. 593-598.

[9] V. Krebs, "Books about U.S. Politics," http://www.orgnet.com/, 2004.

[10] Q. Lu and L. Getoor, "Link-based classification," In *Proc. of the 20th Int'l Conf. on Machine Learning* (*ICML*), 2003, pp. 496-503.

[11] S. Macskassy and F. Provost, "A simple relational classifier," In *Notes of the 2nd Workshop on Multi-relational Data Mining at KDD*, 2003.

[12] S. Macskassy and F. Provost, "Classification in networked data: a toolkit and a univariate case study," *Journal of Machine Learning Research*, 2007 (to appear).

[13] J. Neville and D. Jensen, “Dependency networks for relational data,” In *Proc. of the 4th IEEE Int’l Conf. on Data Mining* (*ICDM*), 2004, pp. 170–177.

[14] J. Neville, D. Jensen, L. Friedland, and M. Hay, “Learning relational probability trees,” In *Proc. of the 9th ACM SIGKDD Int’l Conf. on Knowledge Discovery and Data Mining* (*KDD*), (2003), pp. 625-630.

[15] J. Neville, D. Jensen, and B. Gallagher, “Simple estimators for relational Bayesian classifiers,” In *Proc. of the 3rd IEEE Int’l Conf. on Data Mining* (*ICDM*), 2003, pp. 609-612.

[16] M.E.J. Newman, “The structure and function of complex networks,” *SIAM Review*, 45, 2003, pp. 167-256.

[17] C. Perlich and F. Provost, "Aggregation-based feature invention and relational concept classes." In *Proc. of the 9th ACM SIGKDD Int’l Conf. on Knowledge Discovery and Data Mining* (*KDD*), 2003, pp. 167 – 176.

[18] M. Rattigan, M. Maier and D. Jensen, “Exploiting network structure for active inference in collective classification,” Technical Report 07-22, University of Massachusetts, Amherst, MA, 2007.

[19] B. Taskar, P. Abbeel, and D. Koller, “Discriminative probabilistic models for relational data,” In *Proc. of the 18th Conf. on Uncertainty in AI* (*UAI*), 2002, pp. 485-492.

[20] P. Sen and L. Getoor, “Link-based classification,” Technical Report CS-TR-4858, University of Maryland, College Park, MD, February 2007.

[21] L. Singh, L. Getoor, and L. Licamele, “Pruning Social networks using structural properties and descriptive attributes,” In *Proc. of the 5th IEEE Int’l Conf. on Data Mining* (*ICDM*), 2005, pp. 773-776 .