Improving Learning in Networked Data by Combining Explicit and Mined Links

Sofus A. Macskassy
Fetch Technologies, 2041 Rosecrans Ave, El Segundo, CA 90245
sofmac@fetch.com

Abstract

This paper is about using multiple types of information for classification of networked data in a semi-supervised setting: given a fully described network (nodes and edges) with known labels for some of the nodes, predict the labels of the remaining nodes. One method recently developed for doing such inference is a guilt-by-association model. This method has been independently developed in two different settings–relational learning and semi-supervised learning. In relational learning, the setting assumes that the networked data has explicit links such as hyperlinks between web-pages or citations between research papers. The semi-supervised setting assumes a corpus of non-relational data and creates links based on similarity measures between the instances. Both use only the known labels in the network to predict the remaining labels but use very different information sources. The thesis of this paper is that if we combine these two types of links, the resulting network will carry more information than either type of link by itself. We test this thesis on six benchmark data sets, using a within-network learning algorithm, where we show that we gain significant improvements in predictive performance by combining the links. We describe a principled way of combining multiple types of edges with different edge-weights and semantics using an objective graph measure called node-based assortativity. We investigate the use of this measure to combine text-mined links with explicit links and show that using our approach significantly improves performance of our classifier over naively combining these two types of links.

Motivation

Recent years have seen a lot of attention on classification with networked data in various domains and settings (e.g., (Cortes, Pregibon, & Volinsky 2001; Blum *et al.* 2004; Macskassy & Provost forthcoming; Wang & Zhang 2006)). Networked data is data, generally of the same type such as web-pages or text documents, that are connected via various explicit relations such as one paper citing another, hyperlinks between web-pages, or people calling each other. This paper concerns itself mainly with the problem of *within*-network classification: given a partially labeled network (some nodes have been labeled), label the rest of the nodes in the network.

There have been two separate thrusts of work in this area; one assumes that the data is already in the form of a network such as a web-site, a citation graph, or a calling graph (e.g., (Taskar, Segal, & Koller 2001; Cortes, Pregibon, & Volinsky 2001; Macskassy & Provost 2003)). The second area of work has not been cast as a network learning problem, but rather in the area of semi-supervised learning in a transductive setting (Blum & Chawla 2001; Joachims 2003; Zhu, Ghahramani, & Lafferty 2003; Blum *et al.* 2004; Wang & Zhang 2006). These works assume that you are given a corpus of instances (consisting of labeled and unlabeled instances) and need to first create the links (e.g., given a set of text documents, generate pairwise similarity scores and create a link between two documents if their similarity score is above a given threshold), and then apply within-network classification on this created network. In both scenarios, as mentioned above, the assumption is that the final graph is fully specified (all nodes and edges are known), that the labels of some of the nodes are labeled and that the task is to predict the labels of the remaining nodes. In the work presented here we focus on the case where this is all that is used at classification time. We note that both thrusts of work in this area have independently developed near-identical algorithms to address this classification task: Guilt-by-association (or homophily-based)[1] models with approximate inference techniques have been used in statistical relation learning (Macskassy & Provost forthcoming) and harmonic functions with exact inference have been used in the semi-supervised setting (Zhu, Ghahramani, & Lafferty 2003; Wang & Zhang 2006). These methods empirically produce near-identical results with respect to predictions of nodes.

The main idea of this paper stems from the realization that the two existing approaches both ignore information that is readily available. The work in statistical relational learning has ignored local attributes altogether and focused on the univariate case where only the labels are used. Contrast this with the work in the semi-supervised work, where they have no relations and create links using only local attributes.

The thesis of this paper is that augmenting an existing network (such as a web-site or citation-graph) with links mined from the local attributes ought to increase the information in the network and hence improve the performance of the network classifier. We will show that a naive augmentation, while generally better than either network alone, can be further improved by using objective scaling measures for how to combine the two types of networks. We will show our results on six benchmark data sets, where we augment an existing network by adding K edges from each entity to the K most similar entities to it in the network. We will then show how to intelligently combine the two types of edges to improve performance even further.

We next describe related work, followed by a description

[1] *Homophily* in the context of this paper is the likelihood that a node of a specific class will link to another node of the same class.

of our approach to the within-network classification task, how text-mined links are created and how we combine these two types of networks. We then describe our case study in which we test our main thesis, and conclude with a discussion of the results.

Related Work

The focus of this paper is on within-network learning, an area that has not yet seen much attention in the relational learning community, with a few exceptions (e.g., (Chakrabarti, Dom, & Indyk 1998; Taskar, Segal, & Koller 2001; Macskassy & Provost forthcoming)). One important aspect of networked data is that it allows *collective inference*, meaning that various interrelated values can be inferred simultaneously. Within-network inference complicates such procedures by pinning certain values, but also offers opportunities such as the application of network-flow algorithms to inference as we describe below. More generally, network data allow the use of the features of a node's neighbors, although that must be done with care to avoid greatly increasing estimation variance and thereby error (Jensen, Neville, & Gallagher 2004).

Macskassy and Provost (2003) investigated a simple univariate classifier, the weighted-vote relational neighbor (wvRN). They instantiated node priors simply by the marginal class frequency in the training data. The wvRN classifier performs relational classification via a weighted average of the estimated class membership scores ("probabilities") of the node's neighbors. Collective inference is performed via a relaxation labeling method (Rosenfeld, Hummel, & Zucker 1976) similar to that used by Chakrabarti et al. (1998).

Chakrabarti et al. (1998) combined naive Bayes classifiers with relaxation labeling for collective inference to classify web-pages. In their experiments, performing network classification using the web-pages' link structure substantially improved classification as compared to using only the text. Previous work has shown that wvRN with relaxation labeling generally performs comparably or significantly outperforms this classifier (Macskassy & Provost forthcoming).

Relational Bayesian Networks (RBNs, a.k.a. Probabilistic Relational Models (Koller & Pfeffer 1998; Friedman *et al.* 1999; Taskar, Segal, & Koller 2001) were applied in a within-network classification by Taskar et al. (2001) to various domains, including a data set of published manuscripts linked by authors and citations. Loopy belief propagation (Pearl 1988) was used to perform the collective inferencing. The study showed that the PRM performed better than a non-relational naive Bayes classifier and that using both author and citation information in conjunction with the text of the paper worked better than using only author or citation information in conjunction with the text. Previous work has shown that wvRN with relaxation labeling performed comparably to PRMs on a data set where the PRM used both text and links and wvRN used only the citation edges (Macskassy & Provost 2003).

Techniques recently developed in the area of semi-supervised learning (e.g., (Blum & Chawla 2001; Joachims 2003; Zhu, Ghahramani, & Lafferty 2003; Wang & Zhang 2006)) in a transductive setting (cf. Vapnik (1998)) are directly relevant to the work presented in this paper. Specifically, they consider data sets where labels are given for a subset of cases, and classifications are desired for a subset of the rest. They connect the data into a weighted network, by adding edges (in various ways) based on similarity between cases. We draw upon the work of Zhu et al. (2003) and Wang and Zhang (2006) when creating our text-mined links below.

Classification in networked data

We define within-network classification as follows:

Given graph $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{X})$ where X_i is the (single) attribute of vertex $v_i \in \mathbf{V}$, and given known values x_i of X_i for some subset of vertices $\mathbf{V}^K$, *univariate collective inferencing* is the process of simultaneously inferring the values x_i of X_i for the remaining vertices, $\mathbf{V}^U = \mathbf{V} - \mathbf{V}^K$, or a probability distribution over those values.

As a shorthand, we will use $\mathbf{x}^K$ to denote the (vector of) class values for $\mathbf{V}^K$, and similarly for $\mathbf{x}^U$.

We use the weighted-vote relational neighbor classifier (wvRN) (Macskassy & Provost 2003)[2] paired with relaxation labeling (RL) for collective inference. Using wvRN with relaxation labeling has been shown to perform better than other more sophisticated relational learners on a variety of domains (Macskassy & Provost forthcoming).

The weighted-vote Relational Classifier (wvRN)

The wvRN classifier estimates class-membership probabilities based on two assumptions: (1) a first-order Markov assumption: the label of a node depends only on its immediate neighbors, and (2) the entities in the graph exhibit homophily—i.e., linked entities have a propensity to belong to the same class (cf. (McPherson, Smith-Lovin, & Cook 2001)). This homophily-based model is motivated by observations and theories of social networks (McPherson, Smith-Lovin, & Cook 2001), where homophily is ubiquitous.

Definition. Given $v_i \in V^U$, wvRN estimates $P(x_i|\mathcal{N}_i)$ as the (weighted) mean of the class-membership probabilities of the entities in $\mathcal{N}_i$:

$$P(x_i = X|\mathcal{N}_i) = \frac{1}{Z} \sum_{v_j \in \mathcal{N}_i} w_{i,j} \cdot P(x_j = X|\mathcal{N}_j),$$

where Z is the usual normalizer. As this is a recursive definition (for undirected graphs, $v_j \in \mathcal{N}_i \Leftrightarrow v_i \in \mathcal{N}_j$) the classifier uses the "current" estimate for $P(x_j = X|\mathcal{N}_j)$.

Relaxation Labeling (RL)

We use relaxation labeling (RL) as described in Macskassy and Provost (forthcoming). Rather than treat G as being in a specific labeling "state" at every point (e.g., as a Gibbs sampler does), relaxation labeling retains the uncertainty, keeping track of the current probability estimations for $\mathbf{x}^U$. The relational model must be able to use these estimations. Further, rather than estimating one node at a time and updating the graph right away, relaxation labeling "freezes" the current estimations so that at step $t + 1$, all vertices will be updated based on the estimations from step t. However, doing this often leads to oscillation between states. We therefore use a simulated annealing approach—on each subsequent iteration giving more weight to a node's own current estimate and less to the influence of its neighbors.

[2] Previously called the probabilistic Relational Neighbor classifier (pRN).

More formally, the relaxation labeling inference, using wvRN, is defined as:

$$\mathbf{c}_i^{(t+1)} = \beta^{(t+1)} \cdot \text{wvRN}(\mathbf{C}^{(t)}) + (1-\beta^{(t+1)}) \cdot \mathbf{c}_i^{(t)},$$

where $\mathbf{c}_i^{(t)}$ is a vector of probabilities (probability distribution) which represents an estimate of $P(x_i|N_i)$ at time step t and wvRN($\mathbf{C}^{(t)}$) denotes applying wvRN using all the estimates from time step t. We define the simulated annealing constants as $\beta^0 = k$ and $\beta^{(t+1)} = \beta^{(t)} \cdot \alpha$, where k is a constant between 0 and 1, which for the case study we set to 1.0, and α is a decay constant, which we set to 0.99. These values were set based on Macskassy and Provost (forthcoming).

Whenever we refer to wvRN-RL below, we will mean wvRN used with relaxation labeling as the collective inference method.

Creating text-mined links

In contrast to complex relational learners, wvRN and other graph-based methods are *univariate* in that they only consider class label of nodes–i.e., local attributes are unused. This is unfortunate as it is likely that there is considerable information in the local attributes that should be usable. Macskassy and Provost (2005) tried with limited success to make use of local attributes during classification either through setting priors on nodes or using an ensemble classifier which combined wvRN with other classifiers. However, these results were not encouraging, leaving open the question of how to best use local attributes with graph-based methods.

In this paper we take a different approach and use techniques from the semi-supervised setting (e.g., Wang and Zhang (2006)), where the corpus itself is non-relational, but links are created based on local attributes. The classifiers are then used in a within-network setting just as wvRN has been used with the networked data. Wang and Zhang (2006) create their edges by calculating similarity scores between instances and using the top-K of such links such that an instance will be responsible for creating K links to the K instances that are most similar to it, using the local attributes. Each instance will do so, keeping the similarity score as the weight of the edges that were created.

The idea that we investigate in this paper is that if we were to augment an existing explicit network with the links created from local attributes, and then applying wvRN-RL, then we will in effect be using both relational as well as local attribute information to predict labels of nodes in the network. The three key questions are how we compute similarity scores, what K to use, and how to intelligently combine edges with a weight based on similarity scores from the text-mined links with edges that are already present that likely have different weight statistics. We answer the first question presently and will return to the latter two in the next section and in the case study.

The data that we consider in this paper is textual in nature (in addition to having explicit links), and we therefore adopt a standard bag-of-words vector representation as used in information retrieval (Salton & McGill 1983): for each word in an instance, calculate the tfidf score for that word. The tfidf score is short for term frequency (tf) inverse document frequency (idf), where $\text{tf}(w) = \log(1 + \text{w}_{\text{doc}})$ and $\text{idf}(w) = \log(N/N_w)$, where w_{doc} is the number of times word w appears in a given instance, N is the size of the corpus and N_w is the number of instances that word w appears in. We represent these scores in a vector the dimensionality of all words seen in the corpus such that the i-th element in the vector represents the i-th word in the corpus. Such a vector is likely to be sparse for any given instance as it will only contain a subset of all words ever seen. The similarity of, and the weight of the edge between, two instances is then defined as the cosine of their respective tfidf vectors.

Combining multiple types of edges

One of the key questions in this paper is how we should combine the two networks which have very different semantics for their edge-weights. The text-mined links that we use in our case-study below uses a text-similarity score between 0 and 1 whereas the explicit network, as we will see below, uses discrete positive edge-weights that represent the number of observed links between instances (e.g., the number of hyperlinks between two web-pages or how often two companies have been mentioned together in a news story). We must somehow combine these edges and rescale their edge-weights in a meaningful manner.

The most naive way is to include both types of links with their native weights, an approach which the case study below reveals to provide a lift in performance over using either type of edge by itself. This approach feels intuitively wrong in that it pays no attention to edge-weights, semantics, or how much relative inherent information the edges carry.

We note that Macskassy and Provost (forthcoming) addressed a related question: given multiple types of edges, which type of edge should be used to get the best performance out of wvRN-RL? They used a variant of the *assortativity coefficient* (Newman 2003)–a metric to measure the amount of homophily in a network. Specifically, they developed a variant of this metric–the *node-based* assortativity metric, which was shown to be the best estimator out of three proposed metrics. The node-based assortativity score uses the correlation between the classes linked by edges in a graph. Specifically, it is based on the graph's node-based *assortativity matrix*—a CxC matrix, where cell e_{ij} represents, for (all) nodes of class c_i, the average weighted fraction of their weighted links that link them to nodes of class c_j, such that $\sum_{ij} e_{ij} = 1$. The node-based assortativity coefficient, A_E, is then calculated as follows:

$$A_E = \frac{\sum_i e_{ii} - \sum_i a_i \cdot b_i}{1 - \sum_i a_i \cdot b_i},$$

where $a_i = \sum_j e_{ij}$ and $b_j = \sum_i e_{ij}$.

The to use this measure is to have the overall influence of a type of edge be tied to its observed assortativity score, A_E in such a way that types of edges which have high assortativity count for more than types of edges which have low assortativity scores. Specifically, we do this in two steps for each of the edge types:

1. Normalize the edge-weights. Because of the formulation of wvRN, we note that the weights of edges for a given node in reality only count as much as their fraction of the overall weight. However, in the undirected case this fraction is likely to be different for each of the two nodes. We therefore first make two directed edges out of undirected edges. We then rescale the weights to sum to the correct

fraction of the weighted total for that node. This, in effect, puts all edge types on an even scaling.

2. Rescale the edges by multiplying them with the A_E score of this edge-type. If the A_E score is negative, then set the edge-weight to zero as the behavior of negative edge weights are undefined in the wvRN model.

The advantage of this approach is that it is very general and can easily be used with an arbitrary number of edge types, each having their own semantics of edge-weights and edge statistics.

Study

The thesis of this paper is that augmenting existing networked data with text-mined links will increase the performance of network classification methods. This case study will empirically test this thesis using the wvRN-RL classification method.

Data

We use 6 benchmark data sets from three domains that have been the subject of prior study in machine learning. As this study focuses on combining text-mined links and networked data, instances for which we have no text were removed. Therefore, the statistics of these data differ from those reported previously. We do not provide detailed statistics in this paper due to lack of space.

CoRA The CoRA data set (McCallum *et al.* 2000) comprises computer science research papers. It includes the full citation graph as well as labels for the topic of each paper (and potentially sub- and sub-sub-topics). Following a prior study (Taskar, Segal, & Koller 2001), we focused on 3670 papers within the machine learning topic with the classification task of predicting a paper's sub-topic (of which there are seven).

Papers can be linked in one of two ways: they share a common author, or one cites the other. Following prior work (Lu & Getoor 2003), we link two papers if one cites the other. This number ordinarily would only be zero or one unless the two papers cite each other.

For the text-mined links, we used the abstracts of the papers (we did not have access to the full text of the articles).

WebKB The second domain we draw from is based on the WebKB Project (Craven *et al.* 1998).[3] It consists of sets of web pages from four computer science departments, with each page manually labeled into 7 categories. As with other work (Neville *et al.* 2003; Lu & Getoor 2003), we ignore pages in the "other" category except as described below.

From the WebKB data we produce four networked data sets, one for each of the four universities, ranging in size from 227 to 298. Although the data contains six classes, two classes had so few instances that this, in effect, turned into a four-class problem.

Following prior work on web-page classification, we link two pages by co-citations (if x links to z and y links to z, then x and y are co-citing z) (Chakrabarti, Dom, & Indyk 1998; Lu & Getoor 2003). To weight the link between x and y, we sum the number of hyperlinks from x to z and separately the number from y to z, and multiply these two quantities. For example, if student x has 2 edges to a group page, and a fellow student y has 3 edges to the same group page, then the weight along that path between those 2 students would be 6. This weight represents the number of possible co-citation paths between the pages. We chose co-citations for this case study based on the prior observation that a student is more likely to have a hyperlink to her advisor or a group/project page rather than to one of her peers (Craven *et al.* 1998).

To produce the final data sets, we removed pages in the "other" category from the classification task, although they were used as "background" knowledge—allowing 2 pages to be linked by a path through an "other" page.

To create the text-based links we used the raw text of the web-pages.

Industry Classification (prNewsWire) The final domain we draw from involves classifying companies by industry sector. The prNewsWire data set is based on $38,127$ PR Newswire press releases gathered from April 1, 2003 through September 30, 2003. Each story was tagged with the companies that were mentioned in that story. The data set was then split into two sets: 2809 stories that mentioned more than one company and $35,318$ stories that mentioned only one company.

The former set of 2809 stories was used to create a network of companies where an edge is placed between two companies if they appeared together in the same press release. The weight of an edge is the number of such cooccurrences found in the complete corpus. The resulting network comprises 1274 companies that cooccurred with at least one other company. The latter set of $35,318$ stories was used to create text-mined links. To classify a company, we used Yahoo!'s 12 industry sectors.

Experimental Methodology

We use accuracy as the measure of performance, where accuracy is averaged over 10 runs. For the text-mined links, we set K to 5 unless stated otherwise. In within-network classification, part of the network is initially labeled and we test sensitivity to this by varying from 10% to 90% the amount of initially labeled examples in the network.

In order to combine the mined and explicit edges we must compute A_E. Since this computation requires labels in order to compute the individual e_{ij} cells in the assortativity matrix, we here use nodes for which the label is known.[4]

Results

The primary question we seek to answer is whether augmenting the explicit links with the text-mined links improve performance and secondarily how to we combine the edges to get the best performance. Specifically, we need to know whether using the combined types of edges is better than either edge-type by itself as well as whether it is better than using the local attributes explicitly to perform standard text-classification. Also, we seek to understand the sensitivity to K for the text-mined links.

We first investigate the best way to combine the explicit links and the text-mined links. We do this by comparing how well wvRN-RL performs using one of three ways to

[3] We use the WebKB-ILP-98 data.

[4] We performed the same study using the "true" A_E as computed when all nodes were known. Using only the labeled nodes generally did not significantly change performance.

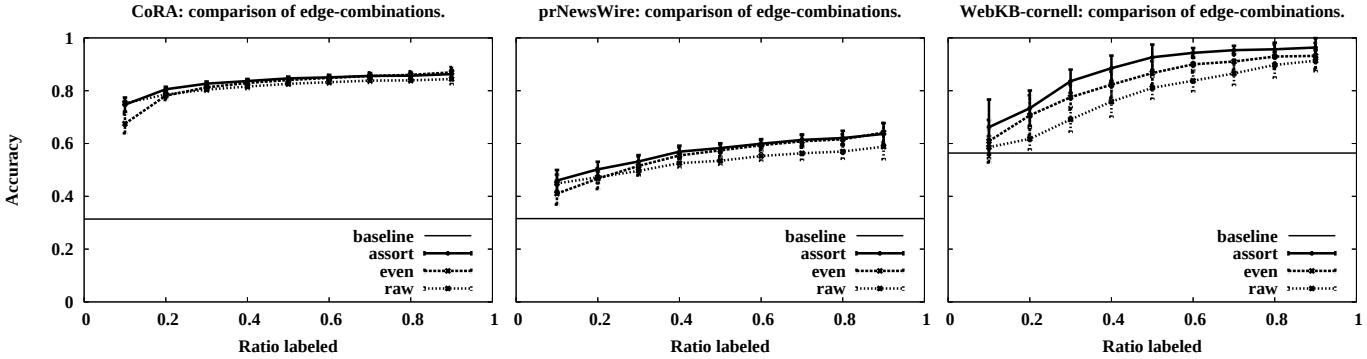

Figure 1: Comparison of three ways of combining edge types: use *raw* scores, count them *even*ly, or use *assort*ativity. The figure shows a representative of the six data sets. The x axis is the ratio of labels initially known in the network and the y axis is the average accuracy over 10 runs.

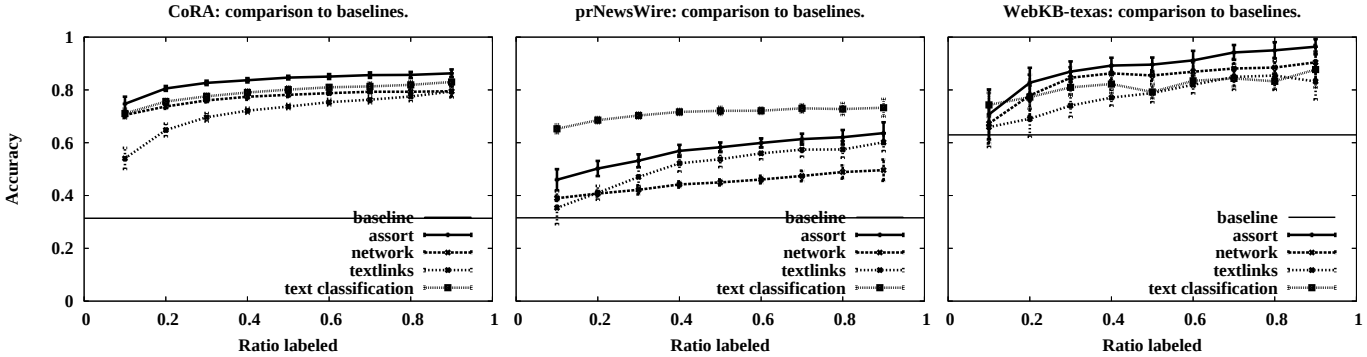

Figure 2: Comparison of using combined links vs. either type alone vs. text-classification. The figure shows a representative of the six data sets. The x and y axes are the same as in Figure 1.

combine the links: (1) use raw edge-weights (i.e., naively combine the two), (2) use edge-weights on an even footing by setting $A_E = 0.5$ for both types of links, and (3) use the computed A_E scores. Figure 1 shows the representative performance gains on three of the data sets. As we can see, using the assortativity measure consistently creates a network where wvRN-RL performs better than either of the other approaches. A Wilcoxon signed-rank test across the six data sets verifies that this finding is statistically significant at $p < 0.01$.

We next verify that using the combined links (using the A_E scores) improves wvRN-RL's performance over using either link-type by itself. We also compare wvRN-RL's performance against the baseline of doing standard text-classification, where we use a multinomial Bayes classifier from the Rainbow text classification system (McCallum 1996).[5] Figure 2 shows the performance comparison on three of the six data sets. The findings here are quite interesting. First, we note that, with the exception of prNewsWire, the combined edges significantly outperforms everything else. A Wilcoxon signed-rank test across the six data sets verifies this finding to be statistically significant at $p < 0.01$.

Interestingly enough we also find that the text-mined links by themselves, again with the exception of prNewsWire, always is the worst of them all. We also find that the text classification and the original explicit links generally performs similarly. However, a Wilcoxon signed-rank test clearly ranks the four methods (at $p < 0.01$) as *combined-links* $>$ *Bayes* $>$ *network* $>$ *text-mined-links*. The only exceptions are that Bayes performed better than the combined links when only 10% of the network was labeled, and the explicit network was statistically equivalent to the text-mined links when 90% of the network was labeled.

These results clearly show that augmenting the network with text-mined links the improves performance of wvRN-RL and that it never decreases performance (vs. other networks). When comparing against the different baseline of a text classifier, wvRN-RL was still the best on five of the six data sets. Note that on prNewsWire, the one case where this was not the case, the original network links were created by implied connections in the text–co-occurrences of companies in the stories–rather than links explicitly created by humans such as citations or hyperlinks in the other domains. However, we still see the lift of combining the links over not combining the links, which is what our experiments are evaluating.

Lastly, we investigate the sensitivity to K, although we don't show any graphs due to lack of space. We tested values of $K = \{5, 10, 20, 50\}$ and found that using $K = 5$ generally performed the best, although the overall performance is relatively robust for the augmented network. Although not shown here, the variability in performance when using only the text-mined links is much higher where the differences become significant (and $K = 5$ still performs the best). This

[5] We tested other text classification methods as well, but only report the Bayes classifier here for clarity of results. Other methods such as SVM with a linear kernel performed equivalently to multinomial Bayes.

argues that when using text-mined links alone, K should be carefully chosen, but in the augmented network this is less critical.

Discussion and Limitations

The thesis of this paper was that augmenting networked data with links mined from the local attributes would increase the amount of the information in the network and hence improve the performance of the network classifier.

We described a simple method of adding such links, using the similarity of nodes based on their local attributes as the criterion for adding edges. We noted that the inherent different semantics of edge-weights between explicit links and text-mined links made it important to combined these edges in the right way and proposed a general method using assortativity to rescale edges to combine them in a meaningful and objective manner.

We empirically tested our thesis by applying our wvRN-RL classifier on six data sets, where the local attributes were text. We used standard information retrieval measures to calculate similarities between instances and used, for each instance, the top K highest-weighted edges as the text-mined network. We first showed that using assortativity to combined the text-mined edges with the explicit edges significantly improved the performance of wvRN-RL. The results also clearly show that augmenting the data with text-mined links improved performance in all the cases and never hurt performance as measured with accuracy. When compared against a text classifier, the augmented network beat the text classifier in five out of the six cases. We noted that in the one case where the text classifier was the best was also the one case where we did not have explicit edges created by people.

Lastly, we conducted a sensitivity study on how many mined links should be added and found that the augmented network was not very sensitive to this beyond $K = 5$.

This work has shown that augmenting a network can indeed improve performance of graph-based methods. This opens the door for many interesting research questions such as what kinds of links we should use to augment the network with. We here proposed a very simple scheme of using instance similarity scores, but it stands to reason that mining for other types of links may very well improve performance even more. This is an edge-creation/searching problem, which is analogous to the feature-creation problem in standard machine learning. Another issue is how text-mined links should be added. We followed an approach from a prior study and selected the top K high-weight edges for a given instance. However, it may be the combining this with a similarity-threshold might make for a more salient network. This is a question we plan to answer in a future study.

References

Blum, A., and Chawla, S. 2001. Learning from Labeled and Unlabeled Data using Graph Mincuts. In *Proceedings of the International Conference on Machine Learning*.

Blum, A.; Lafferty, J.; Reddy, R.; and Rwebangira, M. R. 2004. Semi-supervised learning using randomized mincuts. In *Proceedings of the 21st International Conference on Machine Learning*.

Chakrabarti, S.; Dom, B.; and Indyk, P. 1998. Enhanced Hypertext Categorization Using Hyperlinks. In *ACM SIGMOD International Conference on Management of Data*.

Cortes, C.; Pregibon, D.; and Volinsky, C. T. 2001. Communities of Interest. In *Proceedings of Intelligent Data Analysis*.

Craven, M.; Freitag, D.; McCallum, A.; Mitchell, T.; Nigam, K.; and Quek, C. Y. 1998. Learning to Extract Symbolic Knowledge from the World Wide Web. In *15th Conference of the American Association for Artificial Intelligence*.

Friedman, N.; Getoor, L.; Koller, D.; and Pfeffer, A. 1999. Learning Probabilistic Relational Models. In *Sixteenth International Joint Conference on Artificial Intelligence*.

Jensen, D.; Neville, J.; and Gallagher, B. 2004. Why Collective Inference Improves Relational Classification. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Joachims, T. 2003. Transductive Learning via Spectral Graph Partitioning. In *Proceedings of the International Conference on Machine Learning*.

Koller, D., and Pfeffer, A. 1998. Probabilistic frame-based systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*.

Lu, Q., and Getoor, L. 2003. Link-Based Classification. In *Proc. of the 20th International Conference on Machine Learning*.

Macskassy, S. A., and Provost, F. 2003. A Simple Relational Classifier. In *Proceedings of the Multi-Relational Data Mining Workshop at the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Macskassy, S. A., and Provost, F. 2005. Suspicion scoring of entities based on guilt-by-association, collective inference, and focused data access. In *Annual Conference of the North American Association for Computational Social and Organizational Science*.

Macskassy, S. A., and Provost, F. forthcoming. Classification in Networked Data: A toolkit and a univariate case study. *Journal of Machine Learning Research*.

McCallum, A.; Nigam, K.; Rennie, J.; and Seymore, K. 2000. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval* 3(2):127–163.

McCallum, A. K. 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. `http://www.cs.cmu.edu/~mccallum/bow`.

McPherson, M.; Smith-Lovin, L.; and Cook, J. M. 2001. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology* 27:415–444.

Neville, J.; Jensen, D.; Friedland, L.; and Hay, M. 2003. Learning Relational Probability Trees. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Newman, M. E. J. 2003. Mixing patterns in networks. *Physical Review E* 67. 026126.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann.

Rosenfeld, A.; Hummel, R.; and Zucker, S. 1976. Scene labeling by relaxation operations. *IEEE Transactions on Systems, Man and Cybernetics* 6:420–433.

Salton, G., and McGill, M. J. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc.

Taskar, B.; Segal, E.; and Koller, D. 2001. Probabilistic Classification and Clustering in Relational Data. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*.

Vapnik, V. N. 1998. *Statistical Learning Theory*. John Wiley, NY.

Wang, F., and Zhang, C. 2006. Label propagation through linear neighborhoods. In *Proceedings of the 23rd International Conference on Machine Learning*.

Zhu, X.; Ghahramani, Z.; and Lafferty, J. 2003. Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In *Proc. of the 12th International Conference on Machine Learning*.