Relational Active Learning for Joint Collective Classification Models

Ankit Kuwadekar AKUWADEK@PURDUE.EDU
Jennifer Neville NEVILLE@CS.PURDUE.EDU
Computer Science Department, Purdue University, West Lafayette, IN 47907 USA

Abstract

In many network domains, labeled data may be costly to acquire—indicating a need for *relational active learning* methods. Recent work has demonstrated that relational model performance can be improved by taking network structure into account when choosing instances to label. However, in collective inference settings, *both* model estimation *and* prediction can be improved by acquiring a node's label—since relational models estimate a joint distribution over labels in the network and collective classification methods propagate information from labeled training data during prediction. This conflates improvement in learning with improvement in inference, since labeling nodes can reduce inference error without improving the overall quality of the learned model. Here, we use *across-network* classification to separate the effects on learning and prediction, and focus on reduction of learning error. When label propagation is used for learning, we find that labeling based on prediction *certainty* is more effective than labeling based on *uncertainty*. As such, we propose a novel active learning method that combines a network-based *certainty* metric with semi-supervised learning and relational resampling. We evaluate our approach on synthetic and real-world networks and show faster learning compared to several baselines, including the network based method of Bilgic et al. (2010).

1. Introduction

Recent work in statistical relational learning has demonstrated that learning joint models for *collective inference* in network domains, can often result in significant performance gains (see e.g., Getoor & Taskar, 2007). These models have been broadly applied in a wide range of domains, including bioinformatics, fraud detection, and social network analysis. In many of these applications, the observed improvement in classification accuracy is primarily due to the ability to identify and exploit dependencies among instances. Much of the past work in relational learning has focused on learning models from a fixed set of labeled training nodes—either with a fully or partially labeled network. The implicit assumption has been that labeled training data is either cheap to obtain or it is impossible to obtain additional labels for training. However, in many real-world domains, while it may be cheap to acquire the network topology (e.g., an email network in an organization), it may be costly (but not impossible) to acquire node labels for training (e.g., assessing whether an employee is involved in fraud). In these situations, *active learning* methods could be useful for learning accurate models while minimizing labeling costs.

However, there are a number of challenges to extend active learning methods to relational domains. First, in network domains where instances are dependent, the utility of labeling an instance may depend on more than just the properties of the instance itself. Indeed, recent work on *active inference* has shown that selectively querying for node labels based on network connectivity and model uncertainty can significantly improve the collective inference process (Rattigan et al., 2007; Bilgic & Getoor, 2008; Macskassy, 2009). Second, it is difficult to learn accurate *joint* models from partially-labeled networks. If learning methods ignore the unlabeled portion of the network, then there may not be enough connectivity to learn the relational dependencies accurately. However, in order to incorporate the unlabeled portion of the network, a relational semi-supervised learning algorithm must be able to infer the values of the unlabeled nodes while accurately estimating model parameters. Third, it is difficult to estimate prediction uncertainty for an instance when

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.
Copyright 2011 by the author(s)/owner(s).

joint relational models are applied with collective inference methods. This is due to dependencies between the inferences on neighboring nodes, which makes it is difficult to separate the uncertainty resulting from model estimation from that due to collective inference.

For collective models, expected classification error is comprised of errors due to *learning* and *inference* (Neville & Jensen, 2008). Acquiring a node's label can either improve estimation of the full joint distribution (i.e. reduce learning error) and/or improve collective inference through label propagation (i.e. reduce inference error). We note that relational models are typically applied in two different scenarios. In *across-network* relational learning, a model is learned on one network and then applied to a disconnected network, with the goal of generalizing to other networks in the same domain or nearly disjoint subgraphs within the same larger network. In *within-network* relational learning, a model is learned on a partially labeled network and then applied to predict the class labels in the remainder of the network (i.e. the unlabeled portion), with the goal of transductive inference. When active learning is employed in *within-network* settings, the improvement in *learning* is conflated with improvement in *prediction*, since labeling nodes can reduce inference error without improving the overall quality of the learned model. In this work, we aim to separate the effects of label propagation on estimation and prediction through the use of *across-network* classification.

Although there has been some work focusing on how to label nodes in a network to improve label propagation (Zhao et al., 2008; Guillory & Bilmes, 2009; Cesa-Bianchi et al., 2010) and collective inference (Rattigan et al., 2007; Bilgic & Getoor, 2008; Macskassy, 2009), there has been relatively little work focusing on how network dependencies can be exploited to improve the relational *learning* process. One exception is the ALFNET Bilgic et al. (2010), which is an active learning algorithm for within-network classification tasks where many (node) attributes are available for model learning. The algorithm combines predictions from a collective classifier with predictions from an independent node-level classifier, which ignores the relational structure, and information about the cluster structure in the network, to estimate uncertainty for each unlabeled node in the network. Although ALFNET was shown to improve model performance, since it relies on a node-level i.i.d classifier for uncertainty estimation, it will have limited applicability for domains with few node attributes. Furthermore, the algorithm was developed, and evaluated, for within-network classification tasks, so it may be more useful for transductive inference than for generalization.

In this work, we present a novel approach for active learning in relational and network domains. Our relational active learning (RAL) algorithm combines semi-supervised learning with relational resampling and a utility metric which *minimzes* network variance. We use a relational resampling method to generate many pseudosamples of the network data, from which we then learn an ensemble of collective inference models using semi-supervised learning. Note that this ensemble approach is applicable for network domains with any number of node attributes. Using the ensemble of models, we estimate not only the node level uncertainty in predictions, but also the uncertainty in the neighborhood of each node, in order to choose which nodes to label during the active inference process. We evaluate RAL on across-network classification tasks, comparing to several baselines including an implementation of ALFNET (Bilgic et al., 2010). On both synthetic and real-world datasets, RAL results in significantly faster learning.

2. Background and Related Work

2.1. Active Learning

Active learning is a learning strategy for domains where it is costly to acquire labeled training examples, but unlabeled examples are plentiful. In order to economize the number of labeled data instances needed to learn an accurate model, the objective is generally to select examples that will result in the greatest reduction in the model's generalization error (Saar-Tsechansky & Provost, 2004). The majority of active learning techniques use a utility-based approach, where instances are chosen for labeling based on a calculation of their expected contribution (i.e., utility) to the accuracy of the learned model. Generally, a utility-based active learning technique starts with a pool of labeled examples (typically empty: $L = \emptyset$), a pool of unlabeled examples (typically the dataset: $UL = D$), and an inducer I (e.g., an SVM learner). Then the algorithms proceed as follows:

- Compute the *utility* $u(i)$ for each unlabeled example $i \in UL$.
- Based on the utility scores (e.g., maximizing), choose one or more unlabeled examples to label (e.g., $L = L \cup \{i\}$).
- Apply I to learn a new model from L, repeat.

Most active learning methods differ chiefly in their choice of utility function $u(.)$. For example, Tong & Koller (2002) use the expected reduction in version space, while Roy & Mccallum (2001) use the expected

reduction in future error. Seung et al. (1992) use a committee of classifiers and define the utility to be the amount of disagreement between members of this committees. Saar-Tsechansky & Provost (2004) subsample the training data, learn a model on each subsample, and define the utility of an instance as the variance of the predictions of the set of classifiers.

Although there is a broad set of methods for active learning that have been successfully applied to high-cost or resource-constrained domains, a majority of this work has focused on independent and identically distributed (i.i.d.) data. As a result, the utility measures are calculated for each example *independently*. In relational and network domains, where the i.i.d. assumption does not hold, the benefit of labeling an instance can go beyond the instance itself, as it may improve the predictions about neighbors in the network. Thus, active learning methods need to consider the network structure in the utility calculation.

2.2. Relational Label Acquisition Methods

Recent work in statistical relational learning has explored the idea of active labeling for both learning and collective inference. Research on *active inference* has focused on acquiring labels that will improve the accuracy of collective inference by considering properties of the network structure (Bilgic & Getoor, 2008; Macskassy, 2009). Notably, these methods only query for labels during the inference process - either the model is learned from a fixed set of labels (Bilgic & Getoor, 2008) or the dependencies in the data are not learned (Rattigan et al., 2007; Macskassy, 2009).

Recently, Bilgic et al. (2010) developed ALFNET, an active learning method for network datasets. The method is based on a combination of uncertainty sampling, committee-based ensembles, and community-based network clustering. The algorithm clusters a dataset into at least k clusters and the initial label set is formed by choosing nodes from across the k clusters. At each iteration of ALFNET, the *local disagreement* (LD) score for each node v is calculated by comparing the predictions for v from three classifiers: (i) a content-only classifier based on the attributes of the node in isolation, (ii) a collective classifier based on the attributes and network structure, and (iii) the majority label of labeled nodes in v's cluster. The LD score for node v is defined as the entropy of the predictions from these three classifiers. The utility for a particular cluster is then defined as the sum of the LD scores over the unlabeled nodes in the cluster, normalized by the number of labeled nodes already in that cluster. The set of nodes to label are chosen from k clusters with highest utility scores.

Bilgic et al. (2010) showed that when ALFNET is combined with semi-supervised learning and (attribute) dimensionality reduction, it results in significant performance gains over baseline active learning methods. We note however, that the use of dimensionality reduction and a content-only classifier assumes the availability of many node attributes (e.g., document text for each node), which is not appropriate for many relational domains. In addition, since ALFNET was evaluated in a transductive setting, its performance may not be optimized for across-network classification tasks that require more generalization. More specifically, we note that in transductive network inference, acquiring a node's label can improve performance by informing model estimation or by informing the predictions of nearby unlabeled nodes. This conflates the improvement due to learning with the improvement due to inference, if evaluation focuses on predictive accuracy (since labeling nodes can reduce prediction error without improving the overall quality of the learned model). To separate these effects and focus on learning, we use *across-network* classification for evaluation.

3. Relational Active Learning (RAL)

In this section, we outline our Relational Active Learning (RAL) algorithm in detail. The primary differences between a conventional active learning method and RAL are: (i) the inducer I is a relational learning algorithm, and (ii) the utility function $u(.)$ considers the network structure when calculating the benefit of labeling each instance.

Here we use the relational dependency network (RDN) model (Neville & Jensen, 2007) as the inducer I and learn the RDN in a semi-supervised setting we use the pseudolikelihood EM method of Xiang & Neville (2008)[1]. The algorithm also employs a novel network-based utility measure for $u(.)$, which uses relational resampling (Eldardiry & Neville, 2008) for uncertainty estimation. We describe each of these in detail below.

3.1. General RAL Algorithm

In this work, we consider *across-network* relational learning tasks, where we learn a model from a (partially) labeled training network G_{tr}, and apply the model for collective inference on a separate (i.e., disjoint) testing network G_{te}. The input to the RAL algorithm is $G_{tr} = (V_{tr}, E_{tr})$, which initially contains

[1]We note that the RAL algorithm is not specific to RDNs. Any collective inference model with a semi-supervised learning method could be used for I.

no labeled examples (i.e., $L = \emptyset$, $UL = V_{tr}$), and an RDN inducer I. Then, given a fixed labeling budget B (which limits the number of instances we can label), the algorithm proceeds as follows:

1. WHILE $B \geq 0$:
   (a) Using $L \cup UL$, apply inducer I to learn an ensemble of m models $\mathbf{M} = \{M_1, ..., M_m\}$.
   (b) Using $\mathbf{M}$, compute the utility $u(i)$ for each unlabeled example $i \in UL$.
   (c) Randomly select k examples in proportion to their utility scores (i.e., $p_i = u(i)/\sum_j u(j)$).
   (d) Add the selected examples $\mathbf{S}_k$ to the label set $L = L \cup \mathbf{S}_k$, $UL = UL - \mathbf{S}_k$.
   (e) $B = B - k$.
2. Apply inducer I to learn a model M with $L \cup UL$, return M.

Therefore the two main components of the algorithm are the method by which we use semi-supervised learning to learn an ensemble of models (step 1a) and the utility measures (step 1b). We discuss each of these in detail below.

3.2. Semi-supervised Ensemble Learning

Our RAL algorithm takes a similar approach as the Bootstrap LV approach of (Saar-Tsechansky & Provost, 2004). Bootstrap LV uses *resampling* from the label set L to create multiple training sets $\mathbf{L}' = \{L'_1, ..., L'_m\}$ for learning. The inducer I is applied to each training set in $\mathbf{L}'$ to learn an *ensemble* of models $\mathbf{M} = \{M_1, ..., M_m\}$. Then the ensemble is applied to the examples in UL, resulting in a set of predictions for each example. These sets of predictions can be used to calculate the prediction variance for each instance and the utility measure can then rank the instances in descending order by variance. There are two key challenges to developing a similar approach for network datasets. First, we need a method to sample *with replacement* from a network, while adequately preserving the link structure of the network. To this end, we apply a recently developed method for resampling relational data (Eldardiry & Neville, 2008).

The second challenge is to get an accurate estimate of prediction variance, given the unique characteristics of network datasets. If we ignore the unlabeled part of the network during learning, this can significantly change the structure of the training examples (e.g., there is much lower degree at first). This may bias the models due to the fact that the labeled (training) examples may appear to be drawn from a different distribution than the unlabeled examples. This is particularly difficult in collective inference settings, where we are trying to *learn* the dependencies among neighboring nodes in the network. Semi-supervised learning is one approach to offset this issue, since both labeled and unlabeled examples will be used during learning. For this purpose, we will apply another recently developed semi-supervised relational learning method, which uses pseudolikelihood EM (PLEM) for estimation in RDNs (Xiang & Neville, 2008).

Relational Resampling

The Relational Subgraph Resampling (RSR) method (Eldardiry & Neville, 2008) uses a subgraph sampling approach to preserve the local relational dependencies while generating a pseudosample with sufficient global variance. It has been shown to result in significantly higher accuracy, compared to an i.i.d. resampling approach, when applied to estimate the variance of feature scores in network datasets (Eldardiry & Neville, 2008).

The first phase of the algorithm selects subgraphs based on snowball sampling. It repeatedly selects a subgraph of size b via breadth-first search from a randomly selected seed node. The second phase links up the selected subgraphs. The aim is to preserve the local relational dependencies among instances in each subgraph while randomizing the dependencies across the set of selected subgraphs, in order to generate a pseudosample with sufficient global variance.

Due to the varied link structure of relational data, there will be a large number of nodes on the periphery of the selected subgraphs. If the peripheral nodes are missing a significant portion of their neighbors, this could bias the properties of the sample. To deal with this issue, the RSR algorithm links up the peripheral nodes in the selected subgraphs, while attempting to maintain the global graph properties and attribute dependencies of the original data. More specifically, the relational autocorrelation is maintained by maximizing attribute similarity between nodes as they are linked, while the link structure is maintained by considering the neighborhood similarity when linking nodes.

Semi-supervised RDNs

RDNs typically use pseudolikelihood estimation to learn a model from a fully labeled network. For each data instance x_i, pseudolikelihood models use a local conditional probability distribution (CPD) to represent the conditional probability of the label value v_{x_i}, given its linked nodes, i.e. $P(v_{x_i}|P_a(x_i))$. However, the local CPDs are not required to factor the full joint

distribution. Instead of maximizing likelihood during learning, we maximize the following pseudolikelihood:

$$PL(X;\theta) = \prod_{x_i \epsilon X} p(v_{x_i}|P_a(x_i);\theta)$$

The pseudolikelihood EM (PLEM) approach to learning RDNs learns a joint model of labeled and unlabeled data in the network (Xiang & Neville, 2008). It has been shown that when there is a moderate number of labeled examples, the PLEM approach achieves significantly higher accuracy than other within-network relational learning techniques. The implication of this for active learning is that the model will be more stable and it will produce more accurate estimates of variance in partially-labeled networks. In conventional expectation maximization (EM) approaches to semi-supervised learning, the full data likelihood $P(X|Z,\theta)$ is considered, where Z is the set of unlabeled data. EM consists of two alternating steps:

- **E-Step:** Evaluate $p(Z|\theta^{old})$
- **M-step:** Update the estimator: $\theta^{new} = \arg\max_\theta \sum_Z p(Z|X,\theta^{old}) \log p(X,Z|\theta)$

In PLEM, this update equation is rewritten using the pseudolikelihood of the complete data (X,Z) rather than the full likelihood:

$$\theta^{new} = \arg\max_\theta \sum_Z p(Z|X,\theta^{old}) \sum_{x_i \epsilon X} \log p(v_{x_i}|P_a(x_i);\theta)$$

The complete data pseudolikelihood is defined as the product of CPDs of the observed instance labels, but conditioned on *all* related instances (i.e., $\text{Pa}(x_i)$ contains both the labeled and unlabeled related instances of x_i). Therefore, the M-step can be interpreted as a *collective learning* method. By contrast, in a disjoint learning approach, we only perform parameter estimation once, in which each CPD factor of the pseudolikelihood function is conditioned only on the labeled related instances, and hence the MPLE may be biased when only a few related instances are labeled.

3.3. Network-Based Utility Score

We outline our network-based utility metric below, which chooses to label nodes with low variance, and low disagreement among their neighbors. It is based on the idea that the most valuable unlabeled examples lie in high-density (unlabeled) regions and their predictions disagree the least with their immediate neighborhood and disagree the most with the mean prediction of the ensemble.

We define the *weighted density disagreement* (WDD) utility measure as the product of the divergence of the instance from the overall mean predictions of the ensemble, and the sum of the disagreement between the predictions of the instance with those of its neighbors. To compute the WDD utility of an unlabeled instance, we use the following method:

1. Compute the prediction for each instance i over the ensemble of m models.
2. Let $P_j(i) = p_{M_j}(y_i = +)$ refer to the predicted (marginal) probability that model M_j associates with instance i belonging to class $+$.
3. Let $P_j(UL) = \frac{1}{|UL|} \sum_{i \in UL} P_j(i)$ refer to the average probability that model M_j predicts for all instances in the unlabeled set UL.
4. To represent the certainty of our prediction for an unlabeled example i, we use the Kullback-Leibler (KL) divergence between the ensemble predictions for i and the average predictions for unlabeled instances: $v(i) = KL[P(i)||P(UL)] = \sum_{j \in m} P_j(i) \times \log \frac{P_j(i)}{P_j(UL)}$.
5. We then define WDD for i as follows:

$$u_{WDD}(i) = v(i) \times \sum_{j \in \mathcal{N}_i} e^{-KL[P(i)||P(j)]}$$

Our WDD measure has two parts. The first part, $v(i)$, considers the ensemble of predictions for a node i in isolation. We measure how much the predictions *diverge* from the current average predictions for the unlabeled nodes in the graph, which means nodes with highly confident predictions (i.e., average probabilities close to 0 or 1) will maximize this part of the measure. The second part of the measure considers the neighbors of node i. In order to maximize the second part, the neighbors' predictions should be close to the set of predictions for node i (i.e., disagree least). The overall result is that the WDD measure will favor nodes with highly confident predictions, which also lie in a neighborhood containing many similar predictions. We note that this approach is in contrast to many other utility metrics for i.i.d. settings, which favor nodes with high uncertainty. In relational settings, where inferences are propagated throughout the network during learning, we conjecture that it is more beneficial to label nodes with highly consistent neighborhoods, as it will improve *both* learning and inference.

4. Experimental Evaluation

We evaluated our RAL algorithm on both synthetic and real networks, comparing with several other competing baseline methods. The experiments are intended to evaluate the benefit of (1) the network-based

utility measure, and (2) the semi-supervised and resampling approach to variance estimation.

4.1. Data Sets

The synthetic datasets are generated with the latent group model described in the work of (Neville & Jensen, 2005). The model uses a hidden group structure to generate network data with autocorrelation. For this work, we generated graphs with 250 nodes, in groups with an average size of 25 nodes. Each node has one binary class label and three other boolean attributes. The class label has autocorrelation $\simeq 0.5$.

The first real world dataset is drawn from the Adolescent Health (AddHealth) data, which consists of survey information from middle and high schools, collected in 1994-1995. The survey questions queried for the students social networks along with myriad behavioral/academic attributes. In this paper, we consider the social networks of schools with similar autocorrelation and link patterns. The classification task is to predict whether the student has ever smoked, based on the behavior of their friends in the social network. For the experiments, we selected three similar schools, with sizes ranging from 300-500 nodes, average degree of 7-8, and autocorrelation in the range [0.25,0.35].

The second real world dataset is drawn from the Internet Movie Database (IMDB), which consists of information about movies that were released between 2001 and 2007. We considered movies released in each of these years that were linked with other movies if they shared the same producer. The classification task is to predict whether the movie was a 'blockbuster' with box-office earnings of at least \$60 million . For the experiments, we used datasets for each year—with sizes ranging from 250-300 nodes, average degree of 3-4, and autocorrelation in the range of [0.15, 0.20]

4.2. Methodology

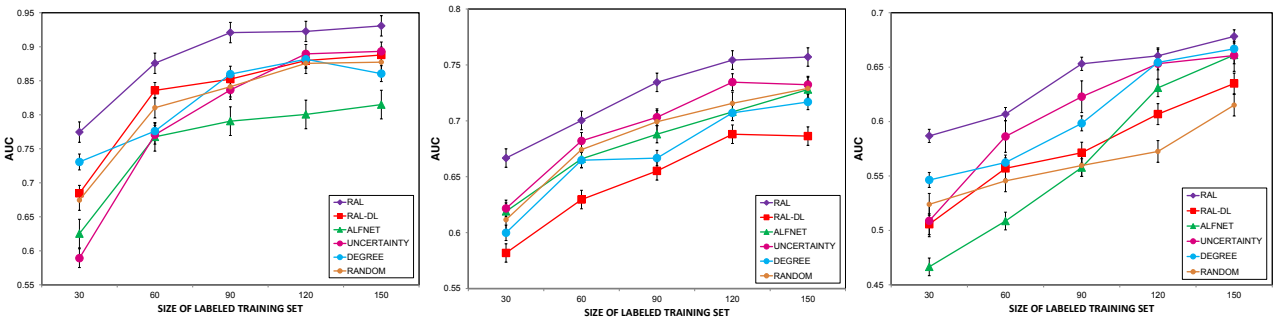
We evaluate the RAL algorithm in an across-network classification setting, where we learn the model on a partially labeled training network, and apply the learned model to another network (drawn from the same distribution) for prediction. At each step of the active learning phase we chose $k = 30$ examples to label. We resampled 10 times to learn an ensemble of $m = 10$ models at each iteration. So the size of the training set increases from 30, 60, ..., 150. At each iteration, we evaluate performance of the learned model on a disjoint test set. For evaluation, we measure area under the ROC curve (AUC) while allowing the model to see the true labels of their neighbors during inference. This measures the *ceiling* performance of the learned collective inference model—which we use to evaluate how the active learning methods reduce learning error (Neville & Jensen, 2008), while controlling for inference error in test set predictions.

For the synthetic data experiments, we generated ten different synthetic datasets and used five of them for training, five for testing. For each train/test pair, we ran the experiment five times to control for random variation in labeling choices. The reported results are averaged over the 5X5=25 trials. For the AddHealth data, we repeatedly selected one school network as the training set for learning and then applied the learned model to the remaining two school networks for evaluation. For the IMDB data, we used one dataset for a randomly chosen year to learn a model and evaluate the model on the dataset for all other years. For each train/test pair, we ran the experiment 25 times to control for variation, thus the reported results are averages of 3X25=75 trials.

We compare our proposed approach (RAL) with several other competing baseline methods.

- **RANDOM:** This uses the RAL algorithm but selects nodes uniformly at random to label instead of using the WDD metric.
- **DEGREE:** This approach is included to measure the effectiveness of labeling based on graph structure alone. Here we use the RAL algorithm that uses a degree-based utility instead of WDD, where nodes with higher degree are more likely to be selected for labeling.
- **UNCERTAINTY:** This approach uses only local estimates of variance for choosing instances to label in RAL, similar to current active learning methods for i.i.d. data. We simply calculate the prediction variance for each instance independently, over the set of ensemble models, and select nodes with maximum uncertainty to label.
- **ALFNET:** We implemented a version which uses the same clustering method and committee ensemble proposed in Bilgic et al. (2010)[2]. To control for model effects and focus on the active learning method, we use RDNs with semi-supervised learning for the collective classifier and an RDN for the content-only classifier. We do not use dimensionality reduction since nodes have less than five attributes in each domain we consider for the experiments.

[2]ALFNET selects a cluster and labels one node at each iteration. We report performance at 30 iterations for consistency. The reported results are with network clusterings into 4-6 groups, which resulted in best performance.



(a) Synthetic data (b) AddHealth (c) IMDB

Figure 1. RAL performance compared to baseline methods on three datasets.

- **RAL-DL:** This approach uses the WDD metric with a non-collective RDN classifier when choosing instances to label. We include it to assess the benefit of using semi-supervised learning in the active learning process.

4.3. Results

Figure 1 shows the evaluation of the RAL algorithm compared with the various baselines on the synthetic and real datasets. The results show that RAL outperforms the other measures by learning a more accurate model with fewer labeled nodes. In particular, we note that RAL outperforms RAL-DL by a large margin, which illustrates the benefit of considering unlabeled nodes in the active learning process. Also, RAL outperforms UNCERTAINTY, which shows that in a joint model, labeling based on *minimum* uncertainty has a larger impact on estimation accuracy than labeling based on *maximum* uncertainty. Lastly, the DEGREE metric does not perform better than RANDOM, indicating that a measure based on network topology alone is not very effective.

In addition, RAL outperforms ALFNET in all three datasets, with a particularly large margin in the IMDB data. These results may be due to differences in our methodology compared to Bilgic et al. (2010). First, we do not have the same number of node attributes in our domains, so dimensionality reduction was not incorporated and the content-only classifier may not have been a good choice for the ensemble. Second, the networks may not have the same cluster structure as those used in Bilgic et al. (2010), or the approach to scoring and labeling by cluster may be more useful in a transductive setting. Finally, ALFNET's focus on uncertainty labeling may not be as effective for reducing learning error (similar to UNCERTAINTY).

In order to explore the impact of the WDD metric, we compared it to three natural alternatives. Recall that WDD prefers nodes with *confident* predictions that lie in neighborhoods with *consistent* predictions. Instead of maximizing confidence and consistency, we can also consider variations that minimize one and/or both aspects of the metric—i.e. (Most/Least) Confident and (Most/Least) Consistent.

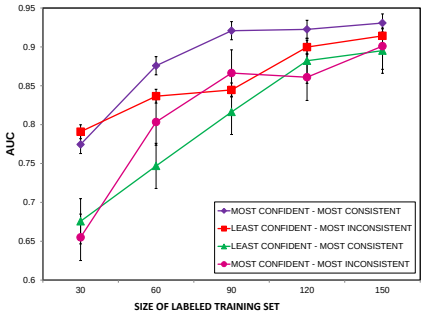


Figure 2. Comparison of WDD Variations

We evaluated all four variations in RAL. Figure 2 plots the results for the synthetic data, which confirms the benefit of labeling by certainty rather than uncertainty. Results are qualitatively the same for the other datasets. The superior performance of the WDD metric observed here contradicts the wide usage of uncertainty in most active learning methods for i.i.d. data. Our findings indicate that, when learning joint models for collective inference, the choice of examples in more consistent neighborhoods may allow for more accurate label propagation throughout the network, which thus increases the accuracy of the learned model. In contrast, picking examples with high uncertainty (for a single node and/or its neighborhood) fails to improve model estimation. An interesting direction for future work, would be to analyze this effect theoretically, possibly by bounding the effect of label propagation on learning error as Guillory & Bilmes (2009) have for prediction error.

5. Conclusion

In this paper, we outlined a novel approach to active learning in relational and collective inference domains, where we combine a network-based utility measure with semi-supervised learning and relational resampling. Our experimental results show that RAL results in significantly faster learning (i.e., higher accuracy with fewer labeled examples). In comparison with ALFNET (Bilgic et al., 2010), RAL exhibits a significant improvement in model accuracy with fewer labeled examples. Moreover, our approach is general enough to apply in relational domains with any number of attributes and can be used for both within-network or across-network classification tasks.

The key idea behind our approach is to consider the nature of the neighborhood, when (i) selecting instances whose predictions are similar to its neighbors, (ii) estimating the models with semi-supervised learning, and (iii) using relational resampling to reconstruct the network structure for ensemble estimation. Our experimental results illustrate the benefit of considering the similarity of an instance's predictions to that of its immediate neighborhood, when identifying instances to actively label.

The WDD metric favors nodes that have (1) highly confident predictions and (2) neighbors with similar predictions. This is inconsistent with many conventional utility metrics used in i.i.d. settings, which favor labeling nodes with high uncertainty. We conjecture that the reason for gains due to *certainty* labeling is from the use of semi-supervised learning for joint relational models, which propagates inferences during learning and may be unduly biased if nodes with less consistent neighborhoods are labeled initially. In future work, we will explore this effect analytically and attempt to formalize the process by which learning error and inference error interact during active learning in networks.

Acknowledgments

We thank Hoda Eldardiry and Rongjing Xiang for their help with the RSR and PLEM code. This research is supported by DARPA and NSF under contract numbers NBCH1080005, SES-0823313, and IIS-0916686.

References

Bilgic, M. and Getoor, L. Effective label acquisition for collective classification. In *KDD'08*, 2008.

Bilgic, M., Mihalkova, L., and Getoor, L. Active learning for networked data. In *ICML'10*, 2010.

Cesa-Bianchi, N., Gentile, C., Vitale, F., and Zappella, G. Active learning on trees and graphs. In *ICML'10*, 2010.

Eldardiry, H. and Neville, J. A resampling technique for relational data graphs. In *Proceedings of the 2nd SNA Workshop, KDD'08*, 2008.

Getoor, L. and Taskar, B. (eds.). *Introduction to Statistical Relational Learning*. MIT Press, 2007.

Guillory, A. and Bilmes, J. Label selection on graphs. In *NIPS'09*, 2009.

Macskassy, S. A. Using graph-based metrics with empirical risk minimization to speed up active learning on networked data. In *KDD'09*, 2009.

Neville, J. and Jensen, D. Leveraging relational autocorrelation with latent group models. In *ICDM'05*, 2005.

Neville, J. and Jensen, D. Relational dependency networks. *The Journal of Machine Learning Research*, 8, 2007.

Neville, J. and Jensen, D. A bias-variance decomposition for collective inference models. *Machine Learning Journal*, 2008.

Rattigan, M., Maier, M., and Jensen, D. Exploiting network structure for active inference in collective classification. In *Proceedings of the 7th ICDM Workshops*, ICDMW '07, pp. 429–434, 2007.

Roy, N. and Mccallum, A. Toward optimal active learning through sampling estimation of error reduction. In *ICML'01*, 2001.

Saar-Tsechansky, M. and Provost, F. Active sampling for class probability estimation and ranking. *Machine Learning*, 54(2):153–178, 2004.

Seung, H. S., Opper, M., and Sompolinsky, H. Query by committee. In *COLT'92*, 1992.

Tong, S. and Koller, D. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.

Xiang, R. and Neville, J. Pseudolikelihood em for within-network relational learning. In *ICDM'08*, 2008.

Zhao, W., Long, J., Zhu, E., and Liu, Y. A scalable algorithm for graph-based active learning. In *Frontiers in Algorithmics*, volume 5059 of *Lecture Notes in Computer Science*, pp. 311–322. 2008.