Active Learning for Networked Data

Mustafa Bilgic MBILGIC@CS.UMD.EDU
Lilyana Mihalkova LILY@CS.UMD.EDU
Lise Getoor GETOOR@CS.UMD.EDU
Department of Computer Science, University of Maryland, College Park, MD 20742 USA

Abstract

We introduce a novel active learning algorithm for classification of network data. In this setting, training instances are connected by a set of links to form a network, the labels of linked nodes are correlated, and the goal is to exploit these dependencies and accurately label the nodes. This problem arises in many domains, including social and biological network analysis and document classification, and there has been much recent interest in methods that *collectively* classify the nodes in the network. While in many cases labeled examples are expensive, often network information is available. We show how an active learning algorithm can take advantage of network structure. Our algorithm effectively exploits the links between instances and the interaction between the local and collective aspects of a classifier to improve the accuracy of learning from fewer labeled examples. We experiment with two real-world benchmark collective classification domains, and show that we are able to achieve extremely accurate results even when only a small fraction of the data is labeled.

1. Introduction

In many domains of interest, the instances are connected via a set of links, thus forming a network, in which neighboring instances frequently have correlated labels. For example, in document classification, documents that cite each other often have similar topics, and in social networks, people that are friends often have similar characteristics. A long tradition in machine learning has focused on exploiting such network information to achieve better predictive accuracy by classifying instances *collectively*, rather than treating them as independent samples (see Sen et al. (2008) for an overview). This approach is appealing because in many cases link information is readily available. For example, in document classification, citation or hyperlinks can be automatically collected. On the other hand, labeling instances requires human attention and may be expensive. For instance, if the task is to predict the effect of a new substance on organisms in a biological network, labeling new examples may require laboratory experiments, whereas the network information regarding interactions among the organisms may be well-known.

Therefore, an important research question is to develop algorithms that reduce the amount of labeling effort required in such tasks. One promising approach is to use *active learning*. In this setting, rather than being presented with a labeled training set from the start, the learner is allowed to request labels for particular examples with the goal of decreasing the number of labels needed in order to achieve a desired level of accuracy. While many effective active learning algorithms have been developed (see Settles (2010) for a survey), to the best of our knowledge, efficient active learners that take direct advantage of explicit network structure in the data have not been considered. The main contribution of this paper is a novel active learning algorithm that addresses this setting. Our algorithm, called ALFNET (for active learning for networked data), exploits the network structure of the domain and the interaction between the local and collective aspects of a classifier to select more informative examples to be labeled, thus improving the accuracy of learning from fewer labeled instances. We demonstrate the effectiveness of ALFNET in several real-world collective classification tasks.

Another important consideration for active learning from networked data is that to learn how to exploit label correlations in the network, the collective classification algorithms need access to the labels of linked nodes. However, because labels are scarce, it is rarely the case that labels of neighboring nodes are known. We introduce a novel semi-supervised technique that

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

can effectively handle the problem of missing labels, thus providing the collective classification algorithms sufficient supervision for learning label correlations.

Further, we argue in favor of combining dimensionality reduction techniques with active learning. Even though it is well known in the literature that high dimensionality is an important problem, especially when labeled data is limited, dimensionality reduction is often overlooked in the active learning community. In this paper, we employ unsupervised dimensionality reduction as a first step of learning and show that it leads to significant performance gains.

Such semi- and unsupervised algorithms are of great importance to active learning settings in which labeled data is typically severely limited. By using them, we ensure that our proposed active learning algorithm improves over strong base learners and obtains improvements beyond those achievable by simpler methods.

The remainder of the paper is organized as follows. In Section 2 we introduce some background and notation. The ALFNET algorithm is described in Section 3, and an empirical evaluation is presented in Section 4. Section 5 discusses related work, and Section 6 concludes.

2. Background

This section introduces necessary background and notation on collective classification and active learning. We assume that our data is represented as a graph $G = (\mathcal{V}, \mathcal{E})$. Each node $V_i \in \mathcal{V}$ is described by an attribute vector $\vec{X}_i$ and a class label Y_i, $V_i = \langle \vec{X}_i, Y_i \rangle$. $\vec{X}_i$ is a vector of individual attributes $\langle X_{i1}, X_{i2}, \ldots, X_{ip} \rangle$. The domain of X_{ij} can be either discrete or continuous whereas the domain of the class label Y_i is discrete and denoted as $\{y_1, y_2, \ldots, y_m\}$. Each edge $E_{ij} \in \mathcal{E}$, where $E_{ij} = \langle V_i, V_j \rangle$, describes some relationship or link between V_i and V_j. For example, in a citation network, the nodes are publications, the node attributes include words, the node labels may be the topics of the papers, and the edges represent citations.

2.1. Collective Classification

In network data, the labels of neighboring nodes are often correlated (though not necessarily positively correlated). For example, papers that cite each other are likely to have similar topics, and proteins that interact are likely to have complementary functions. Exploiting these correlations can significantly improve classification performance over using only the attributes, $\vec{X}_i$, for the nodes. However, when predicting the label of a node, the labels of the related instances are also unknown and need to be predicted. *Collective classification* is the term used for simultaneously predicting the labels $\mathcal{Y}$ of $\mathcal{V}$ in the graph G, where $\mathcal{Y}$ denotes the set of labels of all of the nodes, $\mathcal{Y} = \{Y_1, Y_2, \ldots, Y_n\}$. In general, the label Y_i of a node can be influenced by its own attributes $\vec{X}_i$ as well as the labels Y_j and attributes $\vec{X}_j$ of other nodes in the graph.

The variety of collective classification models that have been proposed make different modeling assumptions about these dependencies. Here, we focus on *local collective classification models*, which consist of a collection of local vector-based classifiers, such as logistic regression, applied iteratively. For this category of collective models, each object is described as a vector of its local attributes $\vec{X}_i$ and an aggregation of attributes and labels of its neighbors. In particular, we use an Iterative Classification Algorithm (ICA) (Neville & Jensen, 2000; Lu & Getoor, 2003), which we briefly explain next. However, our active learning algorithm is largely independent of the underlying collective classification model.

Let $\mathcal{N}_i$ denote the labels of the neighboring nodes of V_i, $\mathcal{N}_i = \{Y_j | \langle V_i, V_j \rangle \in \mathcal{E}\}$. A typical modeling assumption that we also make here is that, once we know the values of $\mathcal{N}_i$, then Y_i is independent of the attribute vectors $\vec{X}_j$ of all neighbors and non-neighbors, as well as of the labels Y_j of all non-neighbors.

In ICA, each node in the graph is represented as a vector that is a combination of node features, $\vec{X}_i$, and features that are constructed using the labels of the nodes' immediate neighbors. Because nodes can have varying numbers of neighbors, we use an aggregation function `aggr` over the neighbor labels in order to get a fixed-length vector representation. For example, `count` aggregation constructs a fixed-size feature vector by counting the number of neighbors with each label; other examples of aggregations include `proportion`, `mode`, etc. Once the features are constructed, then an off-the-shelf probabilistic classifier can be used to learn $P(Y_i \mid \vec{X}_i, \texttt{aggr}(\mathcal{N}_i))$ Here we refer to a classifier that learns $P(Y_i \mid \vec{X}_i, \texttt{aggr}(\mathcal{N}_i))$ as `CC`, for collective classifier. We refer to a classifier that uses only the local node features and learns $P(Y_i \mid \vec{X}_i)$ as `CO`, which stands for content-only classifier.

A key component of this approach is that during inference, the labels of the neighboring instances are often not known. ICA addresses this issue, and performs collective classification by using the predicted labels for the neighbors for computing the aggregates. ICA iterates over all nodes making a new prediction based on the predictions made for the unknown labels of the neighbors in the previous iteration; in the first step of the algorithm, initial labels can be inferred based

solely on attribute information, or based on attribute and any observed neighboring labels.

2.2. Active Learning

Active learning addresses the problem of minimizing the labeling cost by letting the base learner choose which examples to label. A variety of active learning settings have been studied (see Settles (2010) for a survey). Here, we consider the pool-based setting, in which the learner is initially provided with a pool of unlabeled examples $\mathcal{P}$. At each step, it is allowed to select a batch of k instances that are added to its labeled corpus $\mathcal{L}$ and removed from $\mathcal{P}$. We utilize and build upon uncertainty sampling (Lewis & Gale, 1994), committee-based sampling (Seung et al., 1992), and clustering (Dasgupta & Hsu, 2008). A more thorough discussion of related work is provided in Section 5.

3. ALFNET

ALFNET is a novel active learning algorithm for collective classification. Before describing it in detail, we provide a precise statement of the problem we study.

Problem Statement: We are given a graph $G = (\mathcal{V}, \mathcal{E})$, where a subset $\mathcal{P} \subset \mathcal{V}$ is the pool of unlabeled examples, a classification model (e.g., logistic regression), which will be used to train `CC` and `CO`, a batch size k, and a budget B. The task is, within the constraints of B, to make a series of selections of k elements from $\mathcal{P}$ to be labeled by an oracle so that the accuracy of `CC` on unseen data, after training it on the acquired labeled examples $\mathcal{L}$, is maximized.

This is an inductive set-up, in which the test data, $\mathcal{V} \setminus \mathcal{P}$, is *not* available during the active learning process, i.e., testing is done on unseen instances and not on the remaining part of the pool $\mathcal{P}$. However, we assume that the labeled data and the remaining unlabeled instances are available at test time. In the remainder of this section, for simplicity of notation we assume that the test nodes and their adjacent edges have been removed from the training graph G, and so, the initial pool consists of all nodes in $\mathcal{V}$.

The difference between the problem addressed in this and previous active learning approaches is that here we assume that the instances to be classified form a network structure, as defined by the edge set $\mathcal{E}$ of the graph G. ALFNET can take advantage of this additional information in order to select more informative instances. It proceeds by first using the network structure to cluster the data. It then requests the labels of examples that belong to clusters in which `CC` and `CO` (1) disagree about the class assignments of the yet unobserved instances and (2) make predictions that do not match the distribution of observed labels in the cluster.

Algorithm 1: ALFNET: Active Learning for Networked Data

Input: $G = (\mathcal{V}, \mathcal{E})$: the network, `CO`: content-only learner, `CC`: collective learner, k: the batch size, B: the budget
Output: $\mathcal{L}$: the training set

1 $\mathcal{L} \leftarrow \emptyset$
2 $\mathcal{C} \leftarrow$ Cluster the nodes $\mathcal{V}$ of the network G into at least k clusters
3 $\mathcal{C}^k \leftarrow$ Pick k clusters from $\mathcal{C}$
4 **foreach** Cluster $\mathcal{C}_i \in \mathcal{C}^k$
5 $V_j \leftarrow$ Pick an item from $\mathcal{C}_i$
6 Add V_j to $\mathcal{L}$
7 **while** $|\mathcal{L}| < B$
8 Re-train `CO` and `CC`
9 **foreach** Cluster $\mathcal{C}_i \in \mathcal{C}$
10 $score(\mathcal{C}_i) \leftarrow$ `Disagreement(CC,CO,`$\mathcal{C}_i, \mathcal{L}$`)`
11 $\mathcal{C}^k \leftarrow$ Pick k clusters based on the *score*s
12 **foreach** Cluster $\mathcal{C}_i \in \mathcal{C}^k$
13 $V_j \leftarrow$ Pick an item from $\mathcal{C}_i \cap \mathcal{P}$
14 Add V_j to $\mathcal{L}$
15 Remove V_j from $\mathcal{P}$

The high-level pseudo code for ALFNET is described in Algorithm 1. First, in line 2, the network structure, as given by the edge set $\mathcal{E}$, is used to cluster the nodes of G into at least k clusters, where k is the batch size, as defined above. To obtain initial data for training the base learner, k clusters are selected, and one item from each of them is picked and labeled (lines 3-6). This forms the initial labeled set $\mathcal{L}$. ALFNET then proceeds in iterations until the budget B is exhausted (lines 7-15), as follows. Although only the accuracy of `CC` is tested in the evaluation, both `CC` and `CO` are trained in parallel so that their predictions can be compared for the purposes of computing a disagreement score. In each iteration, `CO` and `CC` are re-trained using the currently labeled data $\mathcal{L}$ (line 8). For each cluster, ALFNET computes a score of the disagreement of `CO` and `CC` and selects k clusters based on their scores. We provide details on how the disagreement score is computed later in this section. One unlabeled item from each of the selected clusters is labeled, added to $\mathcal{L}$, and removed from $\mathcal{P}$ (lines 13-15).

Next, we provide more detail on how the clusters are computed, how clusters and elements from them are selected, and how the disagreement score is calculated. For each of these, a variety of options can be explored.

Here we focus on the choices made in our implementation.

Clustering the nodes (step 2): There are many options on how to cluster the nodes $\mathcal{V}$ of the graph G. While in previous work, (Dasgupta & Hsu, 2008), clustering was performed based only on object attributes, here, we take advantage of the available network structure and use a graph clustering algorithm to find clusters. For our experiments we chose modularity clustering by Newman (2006). The algorithm was allowed to split larger clusters into sub-clusters until one of two conditions was met: splitting the cluster further either did not add to the modularity score (Newman, 2006), or it would result in clusters with size smaller than a pre-defined threshold θ. In the experiments, we set $\theta = 200$ and did not consider other values. Clustering the nodes based on network structure is promising because it identifies groups of related nodes in the data, and thus helps the active learner obtain a balanced training set, while avoiding areas of the data for which sufficient supervision is already acquired.

Computing the disagreement score of a cluster (step 10): Intuitively, the disagreement score of a cluster $\mathcal{C}_j$, captures the degree to which CC and CO differ in their predictions from each other, as well as from the observed labels in the cluster. The overall disagreement score of $\mathcal{C}_j$ is defined as the sum of the local disagreement (LD) scores for each unlabeled node in cluster C_j:

$$\texttt{Disagreement}(\texttt{CC},\texttt{CO},\mathcal{C}_j,\mathcal{L}) = \sum_{V_i \in \mathcal{C}_j \cap \mathcal{P}} \texttt{LD}(\texttt{CC},\texttt{CO},V_i,\mathcal{L}).$$

To define the local disagreement LD for an unlabeled node V_i, we collect the predictions of three classifiers, regarding the label of V_i. The first two are the most likely labels predicted by CC and CO, respectively, and the third one is the majority class in the already observed nodes in $\mathcal{C}_j \cap \mathcal{L}$. Let $\mathcal{S}_i$ be the set of all predicted categories by the above three classifiers and $\mathcal{D}_i = \{p_i^h | h \in \mathcal{S}_i\}$, where p_i^h is the proportion of the above three classifiers that predicted category h for V_i. The local disagreement LD of a node V_i is defined as the entropy of V_i's label according to the class distribution $\mathcal{D}_i$:

$$\texttt{LD}(\texttt{CC},\texttt{CO},V_i,\mathcal{L}) = \mathrm{H}_{\mathcal{D}_i}(V_i).$$

Therefore, the more diverse the predictions of the three classifiers, the greater the disagreement about an instance.

Picking clusters (steps 3 and 11): ALFNET picks k of the clusters, from which it selects items for labeling. In general, the clusters may differ in size, and thus the cluster sizes should be taken into account. In step 3 of the algorithm, clusters are picked probabilistically in proportion to their sizes. In step 11, the top k clusters are picked, where clusters are sorted according to their disagreement scores, divided by the number of already labeled items from each cluster. This selection strategy is needed in order to avoid over-investing in the clusters that have already been explored.

Picking an item from a given cluster (step 5 and 13): Given a cluster, an item is chosen randomly at step 5, and the item with the highest local disagreement LD score is picked for labeling at step 13.

3.1. Semi-supervision and Dimensionality Reduction

An important aspect of active learning for networked data is that the collective classification algorithms need access to the labels of linked nodes in order to learn how to exploit label correlations in the network. More specifically, the collective classifier is trained on the combined local and neighborhood feature sets $(\vec{X}_i, \texttt{aggr}(\mathcal{N}_i))$, where $\texttt{aggr}(\mathcal{N}_i)$ is computed *only* over neighbors for which observed labels are available. When labeled data is scarce, there is an insufficient number of observed neighbors. We introduce a novel semi-supervised collective classification method, which is simple, but yet quite effective, as we show in the experiments. In this technique, CO is used to predict labels for the unobserved neighbors of V_i. The aggregation function $\texttt{aggr}(\mathcal{N}_i)$ is then computed over actual (predicted) labels for the observed (unobserved) neighbors. This results in much stronger supervision for the neighborhood features.

Further, we argue for combining dimensionality reduction techniques with active learning. We employ unsupervised dimensionality reduction as a first step of learning and show that it leads to significant performance gains. Specifically, we used principal component analysis (PCA) to transform the original feature space into a smaller one, over which learning from less data is more effective.

4. Experiments

We experimented with ALFNET in two benchmark collective classification tasks. Our experimental study is structured as follows. First, we use the techniques described in Sect. 3.1 to strengthen the base learner. We then compare the accuracy of ALFNET to that of several competitive baselines. Finally, we perform an ablation study in which we test the importance of different aspects of ALFNET. Next, we describe the data

sets and experimental methodology.

4.1. Data

We experimented with two real-world publication data sets – Cora and CiteSeer, prepared by Sen et al. (2008) and available at `http://www.cs.umd.edu/projects/linqs/projects/lbc/`. Cora contains 2708 instances, each belonging to one of seven classes, while CiteSeer contains 3312 instances, each of which is in one of six classes. In both data sets, instances correspond to documents and are described as 0/1 feature vectors, which indicate the absence/presence of a word. The size of the vocabulary in Cora is 1433 and CiteSeer is 3703 words. The network structure of both domains is provided by the citations between documents. Cora contains 5429 citation links, while CiteSeer contains 4732. We ignored the direction of the links, treating two documents as connected if either of them cited the other. In a preliminary analysis, we discovered that in each of the data sets one connected component contained a large percentage of all documents, whereas the remaining documents were sparsely connected in components of average size 2.86/2.75 for Cora/CiteSeer. We attribute this sparsity to missing information in the data. Because in this work we are interested in collective classification, and thus the presence of links between the documents is essential, we cleaned up the data by removing instances that do not belong to the largest connected component. In this way, we were left with 92% of all instances in Cora and 64% in CiteSeer. The cleaned-up version of the data is available from the above URL.

4.2. Methodology

We performed 10-fold cross-validation by randomly partitioning the data into 10 pieces. During training, in each fold of cross-validation, the instances from one of the partitions were held for testing, and all of their links to the rest of the data were removed to avoid contaminating the test set. The instances from the remaining nine partitions had their labels hidden and constituted the pool $\mathcal{P}$, from which the active learner selected $k = 5$ instances to be labeled and added to the training set in each iteration. During testing, the links between the test instances and the rest of the data were restored. Data labeled during the learning stage was available during testing. However, to ensure that all systems were tested on the same set of examples, we evaluated their accuracy only on the held-out test set, and not on unlabeled examples from $\mathcal{P}$. In each fold, we performed three runs for each of the systems; thus, each point on the learning curves presented here is an average of 30 runs.

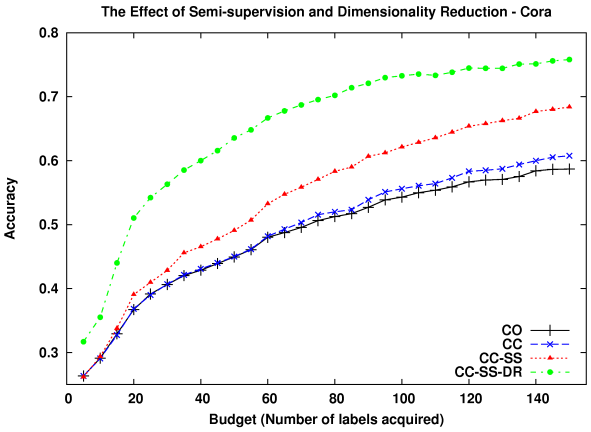


Figure 1. The effect of semi-supervision and dimensionality reduction in Cora. Both semi-supervision and dimensionality reduction provide significant improvements.

The base classifier used for `CO` and `CC` was logistic regression (LR). During preliminary experiments with these data sets, we additionally experimented with SMO and Naive Bayes, and selected LR as the best among the three. For aggregating the label information from the neighboring nodes for `CC`, we used `proportion` where for each class, we take the proportion of neighbors of V_i belonging to that class.

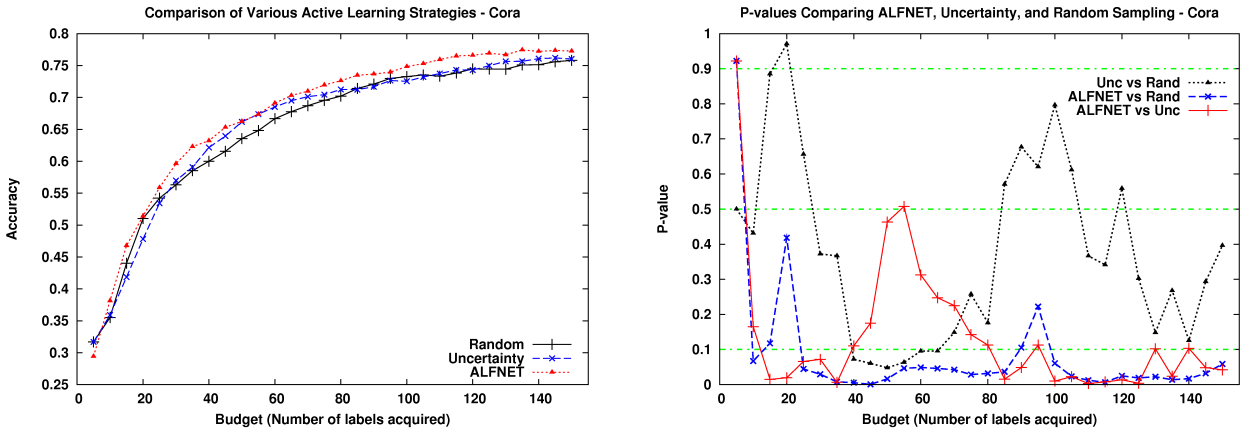
4.3. Results

4.3.1. Semi-supervision and Dimensionality Reduction

In the first set of experiments, we investigate the effect of the techniques from Sect. 3.1. Because this is not the main focus of the paper, we present results only on Cora. Figure 1 compares the performance of content-only `CO` classification, in which LR uses only the local features, collective classification `CC`, in which both local and collective features are included, semi-supervised collective classification (`CC-SS`), which is as described in Sect. 3.1, and semi-supervised collective classification with dimensionality reduction (`CC-SS-DR`), where the number of features is reduced to 100 using PCA. Figure 1 shows that `CC` outperforms `CO` but only slightly. Adding semi-supervision provides a statistically significant improvement, measured using t-test.[1] Furthermore, performing dimensionality reduction provides additional significant benefits over using semi-supervision.

Although the issues considered in the above experi-

[1] All significance claims are at the 0.1 level.



(a) (b)

Figure 2. a) Relative accuracy of ALFNET in Cora. b) P-values of a paired t-test between pairs of systems in Cora. A detailed description is in the text.

ments are orthogonal to the main contribution of this work, we emphasize their importance as a means of ensuring that any improvements obtained by active learning are not over a weak "strawman," but over a carefully selected base learner that already reaches almost optimal accuracy, as reported by Sen et al. (2008), and is very challenging to improve upon. These experiments also provide strong empirical evidence in favor of coupling semi- and un- supervised techniques with active learning. For the remaining experiments, we use `CC-SS-DR` as the base learner and perform active learning using this classifier.

4.3.2. ALFNET

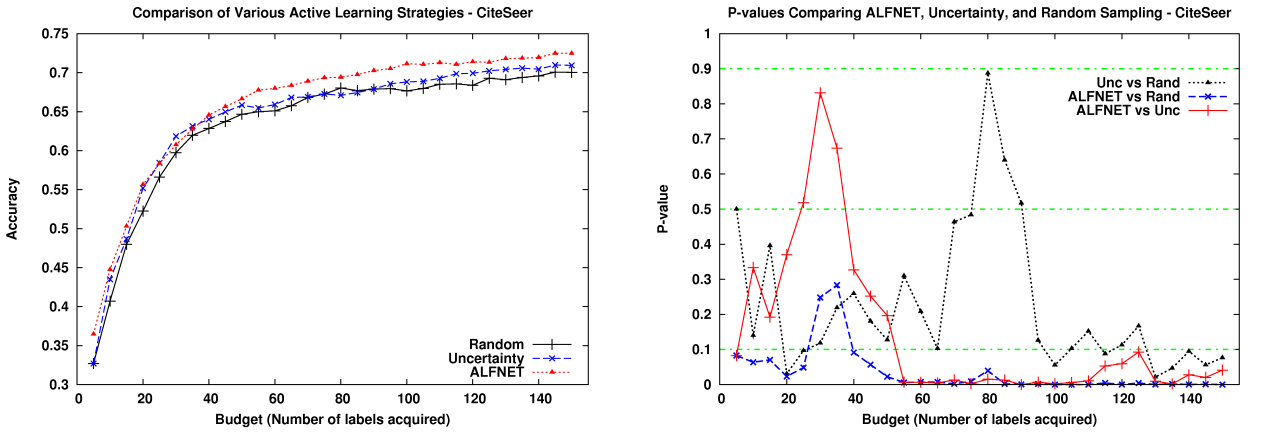
In this set of experiments, we compare the accuracy of ALFNET to that of two baselines–**Random**, which randomly selects examples to be labeled, and **Uncertainty sampling**, which selects the instances about whose labels `CC-SS-DR` is most uncertain (Lewis & Gale, 1994). Uncertainty in our experiments is measured as the expected conditional error of `CC-SS-DR`. To pick k items in each batch, we follow Saar-Tsechansky & Provost (2004) and use the uncertainties to weight the samples and then probabilistically choose k items. This contrasts with picking the top k most uncertain items, which is known to perform poorly (Lewis & Gale, 1994; Saar-Tsechansky & Provost, 2004).

We present two figures for each data set; the first one shows the accuracies of the different active learning systems, whereas the second one shows the p-values of paired t-tests between couples of systems. The accuracy results and the p-values are shown in Figures 2(a) and 2(b) for Cora and in Figures 3(a) and 3(b) for CiteSeer respectively. Figures 2(b) and 3(b) are organized as follows. The X-axis matches the X-axis of the corresponding accuracy graph. For a curve labeled A vs B, any point that falls below the bottom dashed green line indicates a significant win of system A, and any point above the top dashed green line indicates a significant win of system B. For example, for the curve labeled ALFNET vs Uncertainty, ALFNET is significantly better than Uncertainty at points below the bottom dashed green line, and Uncertainty is significantly better at points above the top dashed green line.

One of the first observations is that, even though LR is known to be difficult to improve especially using uncertainty sampling (Schein & Ungar, 2007), for these datasets, uncertainty sampling improves over random sampling. The ALFNET algorithm improves over uncertainty sampling for both Cora and CiteSeer. As the p-values in Figures 2(b) and 3(b) show, ALFNET loses significantly to Random and Uncertainty only once. It wins significantly over Uncertainty in half of the cases in Cora and most of the cases in CiteSeer. It is significantly better than Random in most cases. Finally, we observe that in most cases Uncertainty is not significantly better than Random.

4.3.3. Ablation Experiments

Finally, we test the contribution of each of ALFNET's components by comparing the complete ALFNET to two variants. The first one, **disagreement,** utilizes the disagreement between `CO` and `CC`, but does not exploit the cluster structure of the data. The second variant, **clustering,** pre-clusters the data but selects the instances randomly from each cluster, rather than



(a) (b)

Figure 3. a) Relative accuracy of ALFNET in CiteSeer. b) P-values of a paired t-test between pairs of systems in CiteSeer. A detailed description is in the text.

Table 1. Comparing ALFNET with individual components of it. For both datasets, we show significant wins (as measured by a t-test with 90% confidence interval), ties, i.e. no significant differences, and significant loses of ALFNET over disagreement and clustering.

	Cora			CiteSeer		
	W	T	L	W	T	L
Disagreement	7	22	1	7	23	0
Clustering	9	21	0	12	18	0

using disagreement. We present statistically significant wins and loses, as well as ties (i.e. no significant difference) of ALFNET over disagreement and clustering in Table 1.

These results suggest that the most important component of ALFNET is disagreement; however, using the clustering information provides significant gains over using just the disagreement information.

5. Related Work

ALFNET is most closely related to active learning algorithms for structured prediction tasks, in which the label of an instance is not a single variable but some structured object, such as a sequence or a tree, e.g., (Anderson & Moore, 2005; Culotta & McCallum, 2005; Roth & Small, 2006). The setting explored in this paper differs from structured prediction in that here each example has a single-variable label, but the examples are linked in an arbitrary network structure, whereas in structured prediction, the individual instances are not directly connected, but structure is present in the complex label of each example.

Clustering of the data as a means of avoiding sampling bias has been explored before (Nguyen & Smeulders, 2004; Dasgupta & Hsu, 2008). ALFNET builds on these ideas but bases clustering on the network structure of the data, rather than on the local features of the examples. Moreover, ALFNET treats the labels in a cluster as only one member of a committee, which additionally includes `CO` and `CC`. The idea of using disagreement to identify interesting training instances is inspired by a long tradition of active learning algorithms that use the disagreement between alternative hypotheses, e.g., (Seung et al., 1992). However, previous disagreement-based approaches did not address collective classification.

Graph-based active learning has been addressed before (Zhu et al., 2003; Macskassy, 2009), however, they employed the empirical risk minimization technique (Roy & McCallum, 2001), which is known to be a very expensive procedure, and thus (Zhu et al., 2003; Macskassy, 2009) optimize it specifically for Gaussian Random Fields. In this paper, we present a general active learning technique that is largely independent of the underlying collective model. Finally, (Rattigan et al., 2007; Bilgic & Getoor, 2009) consider the problem of label acquisition for collective classification; however, they assume that the collective model is given and trained, and they perform label acquisition to improve the performance of the model only at inference time.

6. Conclusion

Active learning, semi-supervised learning and collective classification are all important concepts within machine learning. In this work, we have shown how all of them can be leveraged in the setting where we have network data. We developed an algorithm, ALFNET, which leverages network structure in a variety of ways to select samples for labeling in an informed manner. We show how to adapt classic active learning ideas such as disagreement and clustering to the setting in which we have network structure as well as attribute information. In addition, we show how to significantly boost the baseline performance of our active learner by combining dimensionality reduction with semi-supervised learning. We have performed an extensive experimental evaluation, and even over our strong baseline, we are able to show that principled use of structure using ALFNET provides significant improvements over our baseline.

Acknowledgment

We thank the anonymous reviewers for their comments. The first author is supported by ARO Grant #W911NF-08-1-0466 and NSF Grant #0746930. The second author is supported by a CI Fellowship under NSF Grant #0937060 to the Computing Research Association. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the ARO, NSF, or the CRA.

References

Anderson, B. and Moore, A. Active learning for hidden Markov models: Objective functions and algorithms. In *Proceedings of ICML*, 2005.

Bilgic, M. and Getoor, L. Reflect and correct: A misclassification prediction approach to active inference. *ACM Transactions on Knowledge Discovery from Data*, 3(4):1–32, 2009.

Culotta, A. and McCallum, A. Reducing labeling effort for structured prediction tasks. In *Proceedings of AAAI*, 2005.

Dasgupta, S. and Hsu, D. Hierarchical sampling for active learning. In *Proceedings of ICML*, 2008.

Lewis, D. and Gale, W. A sequential algorithm for training text classifiers. In *Proceedings of SIGIR*, 1994.

Lu, Q. and Getoor, L. Link-based classification. In *Proceedings of ICML*, 2003.

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

Neville, J. and Jensen, D. Iterative classification in relational data. In *Proceedings of the Workshop on Statistical Relational Learning at the 17th National Conference on Artificial Intelligence*, 2000.

Newman, M. E. J. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103(23):8577–8582, 2006.

Nguyen, H. T. and Smeulders, A. Active learning using pre-clustering. In *Proceedings of ICML*, 2004.

Rattigan, M., Maier, M., and Jensen, D. Exploiting network structure for active inference in collective classification. In *ICDM Workshop on Mining Graphs and Complex Structures*, 2007.

Roth, D. and Small, K. Margin-based active learning for structured output spaces. In *Proceedings of ECML*, 2006.

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

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

Schein, A. and Ungar, L. Active learning for logistic regression: an evaluation. *Machine Learning*, 68(3): 235–265, 2007.

Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., and Eliassi-Rad, T. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

Settles, B. Active learning literature survey. Technical Report 1648, University of Wisconsin - Madison, Computer Sciences Department, 2010.

Seung, H. S., Opper, M., and Sompolinsky, H. Query by committee. In *Proceedings of the ACM Workshop on Computational Learning Theory*, 1992.

Zhu, X., Lafferty, J., and Ghahramani, Z. Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions. In *ICML workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, 2003.