Active Inference for Collective Classification*

Mustafa Bilgic and **Lise Getoor**
University of Maryland
College Park, MD 20742
{mbilgic,getoor}@cs.umd.edu

Abstract

Labeling nodes in a network is an important problem that has seen a growing interest. A number of methods that exploit both local and relational information have been developed for this task. Acquiring the labels for a few nodes at inference time can greatly improve the accuracy, however the question of figuring out which node labels to acquire is challenging. Previous approaches have been based on simple structural properties. Here, we present a novel technique, which we refer to as *reflect and correct*, that can learn and predict when the underlying classification system is likely to make mistakes and it suggests acquisitions to correct those mistakes.

Introduction

Information diffusion, viral marketing, graph-based semi-supervised learning, and collective classification all attempt to exploit relationships in a network to reason and make inferences about the labels of the nodes in the network. The common intuition is that knowing (or inferring) something about the label of a particular node can tell us something useful about the other nodes' labels in the network. The labels of the linked nodes often tend to be correlated (not necessarily a positive correlation) for many domains; hence, finding the correct label of a node is useful for not only that particular node, but the inferred label also has an impact on the predictions that are made about the nodes in the rest of the network. For example, friends in a social network tend to have similar interests, scientific papers that cite one another tend to have similar topics, and interacting proteins tend to have complementary functions.

It has been shown that methods such as *collective classification*, i.e., classifying the nodes of a network simultaneously, can significantly outperform *content-only classification* methods, which make use of only the attributes of nodes and ignore the relationships between them (see (Sen et al. 2008) for an overview). However, sometimes, the advantage of exploiting the relationships can become a disadvantage. In addition to the typical errors made by content-only classification models (errors due to model limitations, noise in the data, etc.), collective classification models can also make mistakes by propagating misclassifications in the network. This can sometimes even have a domino effect leading to misclassification of most of the nodes in the network. This misclassification of the whole network (or part of it) can occur for both simple models such as iterative classification (Lu and Getoor 2003; Neville and Jensen 2000) and for more complex models that define a global objective function to be optimized, such as pairwise Markov random field models (Taskar, Abbeel, and Koller 2002).

When a prediction system is deployed in a real-life setting, it often has some sort of interaction with its end-users. These end-users can often provide feedback to the system and the system can use this feedback to improve on its predictions. For example, users can rate items for a recommender system, the users interact with the image segmentation feature of the photo editing softwares, speech recognition and spam detection systems can be provided examples, and targeted laboratory experiments can be performed to determine protein functions. However, obtaining feedback is often costly. The users are willing to provide only very little feedback before they can see any value in the system and laboratory experiments can be expensive. Thus, it is imperative to collect feedback for the right entities in the domain.

In our previous work (Bilgic and Getoor 2008)[1] and (Bilgic and Getoor 2009), we developed novel label acquisition strategies for collective classification. In this paper, we summarize two of the approaches. The first and most direct approach is based on approximating the objective function (which we define formally later) and acquiring the label that provides the greatest improvement in the objective value. The second approach, which we refer to as *reflect and correct*, is a simple yet effective acquisition method that learns the cases when a given collective classification model makes mistakes, finds islands of nodes that the collective model is likely to misclassify, and suggests acquisitions to correct these potential mistakes. We compare the acquisition strategies on the synthetic datasets under varying settings and on real-world datasets, and we empirically show that the *reflect and correct* method we propose significantly

*This work was supported by the National Science Foundation, NSF #0746930 and NSF #0438866, with additional support from ARO #W911NF0710428.

[1]Winner of the ACM SIGKDD'08 Best Student Paper Award.

outperforms other methods.

The label acquisition problem has received ample attention within the context of active learning (Cohn, Ghahramani, and Jordan 1996). The biggest difference here is that we assume that we have available an already trained classification model, and thus the learning has been done offline, but we have the option to acquire labels to seed the classification during inference (i.e. the users are interacting with the model). This is the setting (Rattigan, Maier, and Jensen 2007) introduced and referred to as "active inference."

Active Inference in the Broader Context of AI

There are a growing number of intelligent applications which require making classifications for nodes in a network. Examples include product recommendation systems, credit scoring and fraud detection in financial transaction networks, spam detection in email and on the web, topic prediction in hypertext, and influence and community identification in social networks. When these intelligent systems are deployed for real-world use, they often have access to user and expert feedback, albeit it is very limited. Users are willing to rate few movies to access to good recommendations, laboratory experiments can be performed to determine protein functions, user interaction can be utilized to segment an image in an photo editing software, etc. It is essential to gather the user and expert feedback for the right decisions and not waste their effort. A system that requires a tremendous amount of user input and labeled data, is impracticable, while a system that provides an unacceptable rate of incorrect predictions is useless if not harmful. Additionally, for some domains such as bioinformatics, obtaining expert information can require costly laboratory experiments. It is imperative to develop systems that can provide correct predictions with the least amount of feedback possible. Active inference looks at the problem of minimizing user interaction cost while maximizing the benefit and performance of the system.

Problem Formulation

In this problem, we assume that our data is represented as a graph with nodes and edges, $G = (\mathcal{V}, \mathcal{E})$. Each node $V_i \in \mathcal{V}$ is described by an attribute vector $\vec{X}_i$ and a class label Y_i pair, $V_i = \langle \vec{X}_i, Y_i \rangle$. Each edge $E_{ij} \in \mathcal{E}$ describes some sort of relationship between its endpoints, $E_{ij} = \langle V_i, V_j \rangle$ and $\mathcal{N}_i$ denotes the neighbors of Y_i as defined by the edge set $\mathcal{E}$ of the graph.

In collective classification, the label Y_i does not depend on only its attributes $\vec{X}_i$; rather, it in principle can depend and influence other labels Y_j in the graph and their attributes $\vec{X}_j$. A typical assumption is the first order Markovian assumption where the label Y_i depends only on its own attributes and the labels of its immediate neighbors $\mathcal{N}_i$. However, this assumption is not necessary, and various structure learning techniques can be used to learn arbitrary dependencies in the graph.

In the active inference problem, we assume that the underlying collective model has already been learned on a training graph G^{tr}, and we want to maximize the classification performance on a given test graph G. We assume we are given a cost for misclassifying a node; when we classify a node as y_k whereas the correct assignment is y_l, we incur a cost of c_{kl}. The expected misclassification cost (EMC) for a node is then given by:

$$EMC(Y_i|\mathcal{X} = x) = \min_{y_k} \sum_{y_l \neq y_k} P(Y_i = y_l|\mathcal{X} = x) \times c_{kl}$$

Formally, the objective is, given a budget B, to find the optimal set $\mathcal{A}$ of labels to acquire such that the total cost of acquisition, $C(\mathcal{A})$, and the expected misclassification cost EMC over the labels $\mathcal{Y}$ given the feature values $\mathcal{X}$ is minimized:

$$L(\mathcal{A}) \triangleq C(\mathcal{A}) + \sum_{Y_i \in \mathcal{Y} \setminus \mathcal{A}} \sum_{a} P(\mathcal{A} = a) EMC(Y_i|\mathcal{X} = x, \mathcal{A} = a)$$

However, determining the optimal set of labels to acquire is intractable under relatively general assumptions. Krause and Guestrin (Krause and Guestrin 2005) show that finding the optimal set is $\mathbf{NP^{PP}}$-hard for discrete polytrees. Given that we are considering arbitrary networks, such as citation, friendship, and protein networks, finding the optimal solution is at least as hard as, if not harder than, considering discrete polytrees. Therefore, we are forced to resort to approximate and heuristic techniques to get practical solutions.

Active Inference

We first introduce the most obvious approach which is based on approximating the value of the objective function and greedily acquiring the label for the node that provides the highest expected improvement. Then, we introduce our approach, which is based on learning and predicting the misclassifications of a collective classifier. Both techniques associate a $utility$ value with each label (or sets of labels) and makes acquisition decisions based on the utility values.

Approximate Inference and Greedy Acquisition

Finding the optimal set $\mathcal{A}^*$ requires us to consider all possible subsets $\mathcal{A} \subseteq \mathcal{Y}$, and we need to compute the value of the objective function $L(\mathcal{A})$ for each candidate set $\mathcal{A}$, which requires us to compute exact probability distributions over $\mathcal{Y}$. To deal with these inherent obstacles, we first introduce the most obvious approach: approximate inference and greedy acquisition (`AIGA`). In `AIGA`, instead of considering all candidate sets, we consider acquiring one label at a time. That is, we define the $utility$ of a label to be the amount of improvement it provides in the current objective value and we greedily acquire the label that has the highest $utility$:

$$utility(Y_i) \triangleq L(\mathcal{A} \cup \{Y_i\}) - L(\mathcal{A})$$

In essence, the $utility$ function is computing the *expected value of information* for each label (Howard 1966). To address the intractability of the exact probability computations, we resort to approximate inference techniques. With these two approximations, `AIGA` iteratively finds the label that has

the highest $utility$, adds it to the acquisition set, and repeats this step until the budget is exhausted. Note that, even though we make the problem tractable through approximate inference and greedy selection, we still need to run approximate inference for each iteration, for each node, and for each possible value of the label of the node under consideration. This requirement makes this approach still quite expensive, especially if the number of nodes is relatively high and the underlying approximate inference technique is slow. Additionally, the accuracy of this method depends heavily on the precision of the estimated probability values. If the probability estimates are not well-calibrated, then the expected misclassification costs will be incorrect (Zadrozny and Elkan 2001), making the $utility$ values inaccurate.

Reflect and Correct

The method we propose is based on a simple intuition: the sets of nodes that the collective classification model misclassifies tend to be clustered together because misclassifying one node makes it very likely that its neighbors will be misclassified as well (propagation of incorrect information). Thus, there are islands (or peninsulas) of misclassification in the graph – sets of connected nodes that are misclassified. We call such nodes the *flooded* nodes. If we can find these islands of misclassification, then we can potentially trigger correct classification of those islands by acquiring labels for a few of the nodes in the islands. The question is then how to find the islands of misclassification.

We first focus on finding out when a prediction for a particular node is incorrect. Typically, this is done by analyzing the probability estimates of the underlying collective model. However, obtaining reliable probability estimates from a collective model is known to be an NP-hard problem in general. Instead, we propose the problem of finding out whether a node is misclassified itself as a classification problem. We associate a random variable T_i with each $Y_i \in \mathcal{Y}$ where T_i denotes whether the prediction for Y_i was indeed incorrect. We then construct features $\vec{F}_i$ that are possible indicators of whether a node is misclassified and learn a classifier to capture the dependence of T_i on $\vec{F}_i$. The acquisition problem can then be solved by running the collective inference on the graph G, predicting which nodes are misclassified, acquiring a label for a central node among the potentially flooded ones, and repeating the process until the budget is exhausted. This process is illustrated in Figure 1. Because we reflect back on our inference results on the test graph and try to correct the mistakes by acquiring a label, we call this method *reflect and correct* (RAC).

Many different kinds of features can be constructed and included in $\vec{F}_i$ to be used for predicting whether a node is misclassified. Some example features we used in our experiments are: i) whether the collective model and a content-only model disagrees on the label Y_i, ii) how likely that the neighbors of Y_i are also misclassified, and iii) the difference in class distributions in the training set and in the prediction set. However, our approach is general and different kinds of features can be constructed and included as well, especially if there are known domain specific ones. Having constructed

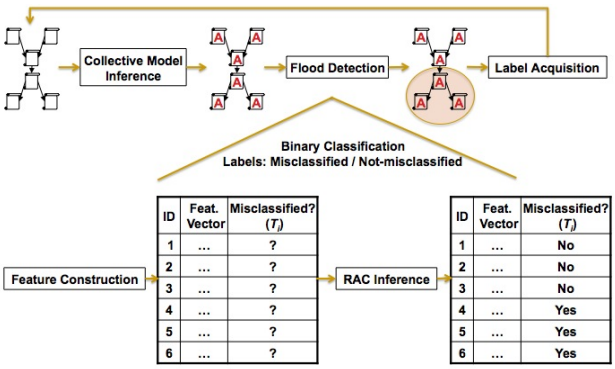


Figure 1: Active inference using the RAC method. We iteratively label the nodes using the collective model, predict which nodes are misclassified, acquire the central node among the misclassified ones, and repeat the process until the budget is exhausted. To predict which nodes are misclassified, we use a classifier whose input consists of a set features that are constructed using the content information of the nodes, information from the neighbors, and global statistics.

these three features, we learn a classifier for estimating the distribution $P(T_i \mid \vec{F}_i)$. To learn this classifier, we need training data, which requires two pieces of information per node: the feature vector $\vec{F}_i$, and the value of T_i. To obtain this information, we use our collective model and the training graph G^{tr}. As a first step, we run collective inference on G^{tr} assuming the labels are unknown, to obtain a new graph where the node labels are now the predicted ones. Let this new graph be called the prediction graph G^{pr}. Then, we obtain the values of $\vec{F}_i$ and T_i by comparing the information in G^{tr} with the information in G^{pr}. Having constructed the training data, we can use any probabilistic classifier to learn the distribution $P(T_i \mid \vec{F}_i)$.

The question that remains to be answered is how to define the utility of label Y_i given $P(T_i \mid \vec{F}_i)$. The most obvious way is to have $utility(Y_i) \triangleq P(T_i \mid \vec{F}_i)$. However, given that we have a limited budget, we want each of the acquisitions to correct as many misclassifications as possible. The node that has the highest probability of misclassification $P(T_i \mid \vec{F}_i)$ can be an isolated node in the network; then acquiring the label for that node might not have a big impact on the predictions for the labels of the rest of the nodes. Based on these intuitions, we want the utility of a label to be a function of whether the corresponding node is misclassified, and how many misclassified neighbors it has. More formally:

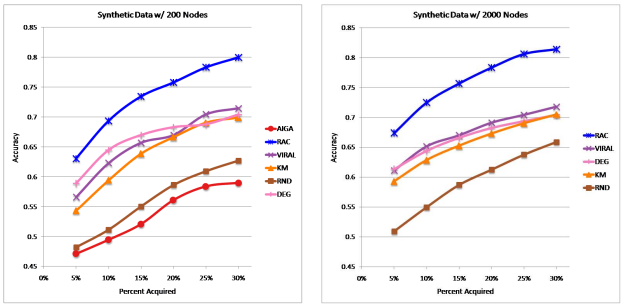
$$utility(Y_i) \triangleq \delta(P(T_i \mid \vec{F}_i) > \sigma) \times (1 + \sum_{Y_j \in \mathcal{N}_i} \delta(P(T_j \mid \vec{F}_j) > \sigma))$$

where $\delta(predicate) = 1$ if the $predicate$ is true, 0 otherwise, and σ is the threshold used to decide if a node is misclassified.

Experiments

We compared the performance of RAC, AIGA, and several other methods on both synthetic and real-world datasets using pair-wise Markov Random Fields (MRF) as the underlying collective model. The other acquisition methods that we compared against include an analogy we drew between active inference and viral marketing (VIR), a competitive method based on K-Mediods clustering of the graph (KM) introduced by (Rattigan, Maier, and Jensen 2007), degree centrality (DEG), and random acquisition (RND). The details on these methods, the synthetic data generation process, and more experiments including iterative classification (Lu and Getoor 2003; Neville and Jensen 2000) as the collective model can be found in (Bilgic and Getoor 2009). Because AIGA is a very expensive method to test, we present results for it only on a small synthetic graph with 200 nodes.

Figure 2(a) shows that AIGA indeed performs worse than random. This result is surprising at first, as one would expect AIGA to perform the best, because it is directly based on approximating the objective function. We think this surprising result is due to at least two factors. The first factor is that AIGA needs well-calibrated probability estimates and obtaining them for a collective model is hard. The second factor could be that it is a greedy approach. RAC on the other hand significantly outperforms other methods on both small and large graphs as shown in Figures 2(a) and 2(b).



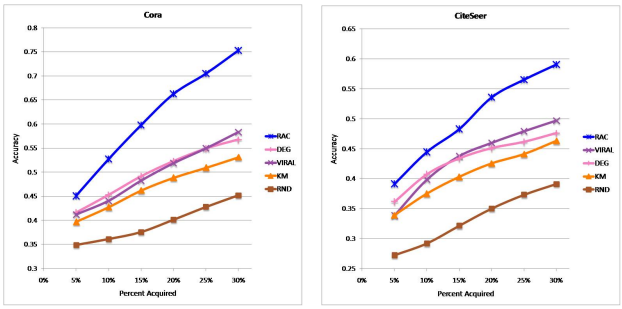
(a) (b)

Figure 2: (a) Experiments comparing AIGA with other methods on small size graphs (200 nodes). (b) Experiments with the rest of the methods on bigger graphs (2000 nodes).

The real-world experiments are based on two publication datasets, Cora and CiteSeer, where the task is to categorize a paper into its topic (Sen et al. 2008). Again, the RAC method significantly outperforms other methods on both Cora (Figure 3(a)) and CiteSeer (Figure 3(b)).

Conclusions

In many real-world applications, a collective inference framework is required to make predictions. These models are often used to guide a human expert that makes the final decisions. Our work on active label acquisition helps to focus the efforts of the expert on feedback that will have the highest impact. It also highlights the complex processes involved in collective classification, and hopefully raises awareness about the sensitivity of these models to errors, and provides some insight in how one might detect these types of errors.



(a) (b)

Figure 3: Experiments on the real-world datasets. (a) The Cora dataset, (b) the CiteSeer dataset.

References

Bilgic, M., and Getoor, L. 2008. Effective label acquisition for collective classification. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 43–51.

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

Cohn, D. A.; Ghahramani, Z.; and Jordan, M. I. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research* 4:129–145.

Howard, R. A. 1966. Information value theory. *IEEE Transactions on Systems Science and Cybernetics* 2(1):22–26.

Krause, A., and Guestrin, C. 2005. Optimal nonmyopic value of information in graphical models - efficient algorithms and theoretical limits. In *Int. Joint Conference on Artificial Intelligence*.

Lu, Q., and Getoor, L. 2003. Link based classification. In *International Conference on Machine Learning*, 496–503.

Neville, J., and Jensen, D. 2000. Iterative classification in relational data. In *SRL Workshop in AAAI*.

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

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

Taskar, B.; Abbeel, P.; and Koller, D. 2002. Discriminative probabilistic models for relational data. In *Annual Conference on Uncertainty in Artificial Intelligence*.

Zadrozny, B., and Elkan, C. 2001. Learning and making decisions when costs and probabilities are both unknown. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 204–213.