Reflect and Correct: A Misclassification Prediction Approach to Active Inference

MUSTAFA BILGIC and LISE GETOOR
University of Maryland at College Park

Information diffusion, viral marketing, graph-based semi-supervised learning, and collective classification all attempt to model and exploit the relationships among nodes in a network to improve the performance of node labeling algorithms. However, sometimes the advantage of exploiting the relationships can become a disadvantage. Simple models like label propagation and iterative classification can aggravate a misclassification by propagating mistakes in the network, while more complex models that define and optimize a global objective function, such as Markov random fields and graph mincuts, can misclassify a set of nodes jointly. This problem can be mitigated if the classification system is allowed to ask for the correct labels for a few of the nodes during inference. However, determining the optimal set of labels to acquire is intractable under relatively general assumptions, which forces us to resort to approximate and heuristic techniques. We describe three such techniques in this article. The first one is based on directly approximating the value of the objective function of label acquisition and greedily acquiring the label that provides the most improvement. The second technique is a simple technique based on the analogy we draw between viral marketing and label acquisition. Finally, we propose a method, which we refer to as *reflect and correct*, that can learn and predict when the classification system is likely to make mistakes and suggests acquisitions to correct those mistakes. We empirically show on a variety of synthetic and real-world datasets that the reflect and correct method significantly outperforms the other two techniques, as well as other approaches based on network structural measures such as node degree and network clustering.

Categories and Subject Descriptors: I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*

General Terms: Algorithms

Additional Key Words and Phrases: Active inference, label acquisition, collective classification, viral marketing, information diffusion

ACM Reference Format:

Bilgic, M. and Getoor, L. 2009. Reflect and correct: A misclassification prediction approach to active inference. ACM Trans. Knowl. Discov. Data. 3, 2, Article 20 (November 2009), 32 pages.
DOI = 10.1145/1631162.1631168 http://doi.acm.org/10.1145/1631162.1631168

This work was supported by the National Science Foundation, NSF #0746930 and NSF #0438866, with additional support from ARO #W911NF0710428.
Authors' address: A. V. Williams Building, University of Maryland, College Park, MD 20742-3255; email: {mbilgic, getoor}@cs.umd.edu.

DOI 10.1145/1631162.1631168 http://doi.acm.org/10.1145/1631162.1631168

1. 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. For instance, 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. Thus, it has been shown that methods such as collective classification, that is, 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 [Chakrabarti et al. 1998; Neville and Jensen 2000; Getoor et al. 2001; Taskar et al. 2002; Lu and Getoor 2003a; Jensen et al. 2004; Macskassy and Provost 2007; Sen et al. 2008]. 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. For example, consider a simple binary classification problem where an island of nodes that should be labeled with the positive label are surrounded with a sea of negatively labeled nodes. The island may be *flooded* with the labels of the neighboring sea of negative nodes.

This flooding of the whole network (or part of it) can occur for simple models such as iterative classification [Lu and Getoor 2003a; Neville and Jensen 2000] and label propagation [Zhu and Ghahramani 2002]. A misclassification can be propagated to the rest of the network, especially if the misclassification is systematic and common, such as misclassifying nodes as belonging to the majority class. Flooding can also happen for more complex models that define a global objective function to be optimized. For example, for pairwise Markov random field models [Taskar et al. 2002] with parameter values that prefer intra-class interactions over inter-class interactions, the most probable configuration of the labels might be the one where most of the network is labeled with one class. Or, for a graph mincut formulation [Blum and Chawla 2001], where we pay a penalty for inter-class interactions, the best objective value might be achieved by assigning only one label to each connected component in the graph.

One strategy for avoiding flooding is to have an expert in the loop during inference, who can guide the inference and constrain the solution space in the right directions by providing the correct labels for a few nodes. Depending on the application, labels can be acquired by asking the expert to rate specific items, a company can provide free samples to a small set of customers and customers' viral networking or purchasing behavior can be observed, or targeted laboratory experiments can be performed to determine protein functions, etc. However,

providing these additional labels is often costly and we are often limited to operate within a given budget. As we show later, determining the optimal set of labels to acquire is intractable under relatively general assumptions. Therefore, we are forced to resort to approximate and heuristic techniques to get practical solutions.

In this article, we describe three polynomial-time label acquisition strategies. The first and most direct approach is based on approximating the objective function (which we define formally in Section 2) and greedily acquiring the label that provides the greatest improvement in the objective value. The second approach draws on an analogy between viral marketing and label acquisition, and translates one of the existing viral marketing formulations into a label acquisition strategy. The third 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.

In addition to these three methods, we also experiment with acquisition strategies that are based on network structural measures such as node degree and network clustering. To compare the different acquisition strategies, we use two representative collective models: one that consists of a collection of local classifiers, and one that defines and optimizes a global objective function. Using synthetic datasets, we analyze the cases when flooding might happen and its degree of severity. 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 outperforms all of the other methods.

The label acquisition problem has received ample attention within the context of active learning [Cohn et al. 1996; McCallum and Nigam 1998; Tong and Koller 2002]. There are two main differences between the scenario we address and the active learning scenario. First, active learning has traditionally been concerned with non-relational data; here, we are interested in network data. The second (and the biggest) difference is that we assume that we have available an already trained model of the domain, and thus the learning has been done offline, but we have the option to acquire labels to seed the classification during inference. This is the setting Rattigan et al. [2007] introduced and referred to as "active inference." They looked at the relational network classifier, introduced by Macskassy and Provost [2003], in which there are no node attributes; only labels are propagated. Here, we build on this, and look at networks in which the nodes have attribute information and compare to the structural strategy that they introduced.

This article builds upon our earlier work [Bilgic and Getoor 2008]. Our contributions in this article include:

—We introduce three novel label acquisition strategies for collective classification.

—We describe the three acquisition strategies using a common utility-based active inference framework.

—We investigate the flooding and active inference using two different types of collective models, which provides additional insights.
—We experiment with different content-only classifiers for feature construction for the proposed method.
—We explore the spectrum of flooding using synthetic datasets with varying attribute noise levels and label correlations.
—We perform extensive experiments comparing different acquisition strategies in different settings.

We formulate the label acquisition problem and state the objective function in Section 2. Then, we explain the three approaches in Sections 3.1, 3.2, and 3.3. We then show experimental results on both synthetic and real datasets (Section 4). Finally, we discuss related work (Section 5), summary and future work (Section 6 and Section 7) and then conclude (Section 8).

2. PROBLEM FORMULATION

We begin by reviewing the collective classification problem and define the objective function for label acquisition for collective classification. 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, $V_i = \langle \vec{X}_i, Y_i \rangle$. $\vec{X}_i$ 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 $E_{ij} \in \mathcal{E}$ describes some sort of relationship between its endpoints, $E_{ij} = \langle V_i, V_j \rangle$. Examples include:

—*Social Networks*. Here, the nodes are people, the attributes may include demographic information such as age and income and the edges are friendships. The labels indicate categories of people, for example we may be interested in labeling the people that are likely to partake in some activity (e.g., smoking, IV drug use), have some disease (e.g., tuberculosis, obesity), or exhibit some behavior (buying a product, spreading a rumor).
—*Citation Networks*. The nodes are publications, the attributes include content information and the edges represent citations. The labels may be the topics of the publications, or an indication of the reputation of the paper, for example whether the paper is seminal or not.
—*Biological Networks*. Where for example, the nodes represent proteins, attributes include annotations, and edges represent interactions. In this domain for example, we may be interested in inferring protein function.

2.1 Collective Classification

In graph data, the labels of neighboring nodes are often correlated (though not necessarily positively correlated). For example, friends tend to have similar smoking behaviors, papers are likely to have similar topics to the papers that they cite, and proteins 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.

There are many collective classification models proposed to date that make different modeling assumptions about these dependencies. They can be grouped into two broad categories. In the first category, *local collective classification models*, the collective models consist of a collection of local vector-based classifiers, such as logistic regression. For the 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. Examples include Chakrabarti et al. [1998], Neville and Jensen [2000], Lu and Getoor [2003a], Macskassy and Provost [2007], and McDowell et al. [2007]. The second category of collective classification models are *global collective classification models*. In this case, the collective classification is defined as a global objective function to be optimized. In many cases, a relational graphical model is learned over all the attributes and labels in the graph, and a joint probability distribution over these attributes and labels is learned and optimized. Examples of this category include conditional random fields [Lafferty et al. 2001], relational Markov networks [Taskar et al. 2002], probabilistic relational models [Getoor et al. 2002], and Markov logic networks [Richardson and Domingos 2006].

In this article, we use an example model from each category, which we explain briefly here. For the local collective classification model, we use Iterative Classification Algorithm (ICA) [Neville and Jensen 2000; Lu and Getoor 2003a], and, for the global collective classification model, we use a pairwise Markov Random Fields (MRF) based on the relational Markov network of Taskar et al. [2002]. We first introduce notations and assumptions common to both models and then describe the two approaches.

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 general assumption that is made is the Markov assumption that Y_i is directly influenced only by $\vec{X}_i$ and $\mathcal{N}_i$. Given the values of $\mathcal{N}_i$, Y_i is independent of $\mathcal{Y} \setminus \mathcal{N}_i$ and is independent of $\mathcal{X} \setminus \{\vec{X}_i\}$, where $\mathcal{X}$ denotes the set of all attribute vectors in the graph, $\mathcal{X} = \{\vec{X}_1, \vec{X}_2, \ldots, \vec{X}_n\}$. That is, once we know the the values of $\mathcal{N}_i$, then Y_i is independent of attribute vectors $\vec{X}_j$ of all neighbors and non-neighbors, and it is independent of labels Y_j of all non-neighbors.

2.1.1 *Iterative Classification Algorithm (ICA).* In the ICA model, 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 each node can have a varying number of neighbors, we use an aggregation function over the neighbor labels in order to get a fixed-length vector representation. For example, the `count` aggregation constructs a fixed-size feature vector by counting how many of the neighbors belong to 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 $P(Y_i \mid \vec{X}_i, \texttt{aggr}(\mathcal{N}_i))$, where aggr is an aggregate function that converts a set of inputs into a fixed length vector. One can use a single classifier to learn $P(Y_i \mid \vec{X}_i, \texttt{aggr}(\mathcal{N}_i))$ or can use a structured classifier to learn $P(Y_i \mid \vec{X}_i)$ and $P(Y_i \mid \texttt{aggr}(\mathcal{N}_i))$ separately, which can be combined in a variety of ways to compute $P(Y_i \mid \vec{X}_i, \texttt{aggr}(\mathcal{N}_i))$.

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 feature construction. 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. ICA learning is typically done using fully labeled training data, however there are also approaches that use semi-supervised techniques for learning ICA [Lu and Getoor 2003b; Xiang and Neville 2008].

2.1.2 *Pairwise Markov Random Fields (MRF).* Now, we briefly describe the MRF model we used. In MRF, the joint probability of $P(\mathcal{Y}|\mathcal{X})$ is given by:

$$P(\mathcal{Y} \mid \mathcal{X}) = \frac{1}{Z} \prod_{Y_i \in \mathcal{Y}} \phi(Y_i, \vec{X}_i) \phi(Y_i, \mathcal{N}_i),$$

where ϕ are the "compatibility" functions that in effect capture the degree and the strength of the relationships between different values of Y_i and $\vec{X}_i$ and Y_i and $\mathcal{N}_i$. For example, in social networks, these compatibility functions can be considered to capture the degree of correlations between the smoking behaviors of friends in a network. The Z is a normalization function ensuring that $P(\mathcal{Y} \mid \mathcal{X})$ is a legitimate probability function.

Note that in this representation, the number of arguments for ϕ are neither fixed nor uniform; that is, for one particular node $\phi(Y_i, \mathcal{N}_i)$ can have just two arguments whereas for a different node it can have hundreds of arguments. This property is undesirable for many reasons including representational inefficiency, lack of sufficient data for accurate parameter estimation, and difficulty of generalizing from train data to the test data. To get around these problems, one trick is to assume a functional form for the ϕ. For example, we can approximate the interactions between many nodes as the product of the pairwise interactions; this assumption leads to pairwise Markov Random Fields:

$$P(\mathcal{Y} \mid \mathcal{X}) = \frac{1}{Z} \prod_{Y_i \in \mathcal{Y}} \left(\prod_{j=1}^{p} \phi(Y_i, X_{ij}) \right) \left(\prod_{Y_j \in \mathcal{N}_i} \phi(Y_i, Y_j) \right). \quad (1)$$

One other trick to make sure that the ϕ are generalizable from train to test data and also to make sure that $P(\mathcal{Y} \mid \mathcal{X})$ is integrable is to represent ϕ as log-linear combinations of a set of indicator functions of the form $f_{y_i, x_{ij}}(Y_i, X_{ij}) \triangleq \sigma(Y_i = y_i, X_{ij} = x_{ij})$, and $f_{y_i, y_j}(Y_i, Y_j) \triangleq \sigma(Y_i = y_i, Y_j = y_j)$. Then, the

compatibility functions are represented as:

$$\phi(Y_i, X_{ij}) = e^{\left(\sum_{ij} w_{y_i,x_{ij}} f_{y_i,x_{ij}}(Y_i,X_{ij})\right)}; \phi(Y_i, Y_j) = e^{\left(\sum_{ij} w_{y_i,y_j} f_{y_i,y_j}(Y_i,Y_j)\right)},$$

where ws are weights to be learned from the train data. With this representation, the products in Eq. (1) turn into sums in the exponent of e. Then, maximum likelihood learning can be done by the taking log of ($P(\mathcal{Y} \mid \mathcal{X})$, which is now sum of the products of the weights and the indicator features, and maximizing it through gradient ascent methods [Taskar et al. 2002].

2.2 Label Acquisition

For active inference for both ICA and MRF, we assume that we are given a training graph $G^{tr}(\mathcal{V}^{tr}, \mathcal{E}^{tr})$ where labels of all the nodes are known. Let CM represent the collective model we use, here either ICA or MRF. We train our collective model CM using this training graph. Given a test graph G, a trained model CM, and assuming the values of the attribute vectors $\mathcal{X}$ are known, but the labels for the nodes are unknown, our goal is to correctly predict $\mathcal{Y}$. 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:

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

The total expected misclassification cost is then sum of the expected misclassification costs for the individual nodes:

$$\sum_{Y_i \in \mathcal{Y}} EMC(Y_i|\mathcal{X} = x, CM).$$

As mentioned in the introduction, we are interested in settings where we are able acquire additional information, or to ask for the labels for some of the nodes. More formally, we consider the case where we can acquire the values for a subset of the labels $\mathcal{A} \subseteq \mathcal{Y}$. The acquisition of $\mathcal{A}$ changes the misclassification cost as follows:

$$\sum_{Y_i \in \mathcal{Y} \setminus \mathcal{A}} EMC(Y_i|\mathcal{X} = x, \mathcal{A}, CM).$$

However, we do not know the values of the labels in $\mathcal{A}$ before we acquire them. Thus, we take an expectation over possible values.

$$\sum_{Y_i \in \mathcal{Y} \setminus \mathcal{A}} \sum_{a} P(\mathcal{A} = a) EMC(Y_i|\mathcal{X} = x, \mathcal{A} = a, CM).$$

In this general setting, we also attach costs to acquiring labels. Let the cost of acquiring the value of the label Y_i be C_i. Extending it to sets, $C(\mathcal{A}) = \sum_{Y_i \in \mathcal{A}} C_i$. Then, the total cost we incur is just the sum of the acquisition cost and the

expected misclassification cost:

$$L(\mathcal{A}) = C(\mathcal{A}) + \sum_{Y_i \in \mathcal{Y} \setminus \mathcal{A}} \sum_{a} P(\mathcal{A} = a) EMC(Y_i | \mathcal{X} = x, \mathcal{A} = a, CM). \quad (2)$$

Given a spending budget B, the label acquisition problem, and our objective, is then to find the optimal subset

$$\mathcal{A}^* = \underset{\mathcal{A} \subseteq \mathcal{Y}, C(\mathcal{A}) \leq B}{\operatorname{argmin}} L(\mathcal{A}),$$

minimizing the sum of expected misclassification cost and acquisition cost.

Finding the optimal $\mathcal{A}^*$ requires us to evaluate the objective function for each candidate $\mathcal{A}$ along with an efficient search and exploration of the candidate space. Krause and Guestrin [2005] discuss this problem in the context of value of information calculations in graphical models. They associate a reward for observing the value at a node, which is equivalent to acquiring a label in our case, and they show that reward computations are **#P**-complete even for discrete polytrees. They also show that finding the optimal set to acquire in batch mode, where the acquisition decisions are made all at once, is $\mathbf{NP}^{\mathbf{PP}}$-complete for discrete polytrees. Finally, they show that finding the optimal set in a conditional plan, that is the next acquisition is conditioned on the previous acquisitions, is $\mathbf{NP}^{\mathbf{PP}}$-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. The details of these theoretical limits can be found in Krause and Guestrin [2005].

3. ACTIVE INFERENCE

Since finding the optimal solution to the label acquisition problem is intractable under relatively general assumptions, we must resort to approximate and/or heuristic acquisition techniques. In this article, we introduce three such techniques. Each technique associates a *utility* value with each label (or sets of labels) and makes acquisition decisions based on the utility values. The first strategy that we propose approximates the objective function and defines the utility of a label in terms of the improvement it achieves in the objective value. The second method draws an analogy between viral marketing and active inference and associates utilities with labels accordingly. The third is a simple yet effective and intuitive approach based on learning and predicting the misclassifications of a collective classifier.

3.1 Approximate Inference and Greedy Acquisition (AIGA)

There are two reasons why finding the optimal set $\mathcal{A}*$ is intractable: (1) unless the probability distribution for $\mathcal{A}*$ can be factored with the acquisition of a single label Y_i, we need to consider all possible subsets $\mathcal{A} \subseteq \mathcal{Y}$, which is exponential in the size of $\mathcal{Y}$, (2) for each candidate set $\mathcal{A}$, we need to compute the value of the objective function $L(\mathcal{A})$ (Eq. (2)), which requires us to compute exact probability distributions over $\mathcal{Y}$.

To tackle these two obstacles, we 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*:

$$utility_{aiga}(Y_i) \triangleq L(\mathcal{A} \cup \{Y_i\}) - L(\mathcal{A}).$$

In essence, the $utility_{aiga}$ 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. For the collective models (such as ICA) that are a collection of local classifiers, we use iterative approaches to approximate the conditional probability distributions for the labels. For the collective models that define and optimize a global objective function (such as MRF), there exist a variety of approximate inference techniques, including loopy belief propagation [Yedidia et al. 2000], variational methods [Jordan et al. 1999], and Gibbs sampling [Gilks et al. 1996]; in this article, we use loopy belief propagation.

With these two approximations, AIGA iteratively finds the label that has the highest $utility_{aiga}$, 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 $utility_{aiga}$ values inaccurate.

3.2 Viral Marketing Acquisition (VMA)

Another approach to label acquisition is based on an analogy to viral marketing [Richardson and Domingos 2002; Kempe et al. 2003; Leskovec et al. 2007]. In the viral marketing setting, we have customers that are potential buyers of a product, and the customers have relationships between each other, such as family, friendship, co-worker, etc. When a customer buys a product, the customer advertises it (by word of mouth) to his or her neighbors in the network. The objective of viral marketing is to maximize the sales for a product by marketing it to the right set of customers, while minimizing the marketing costs. Thus, similar to the label acquisition problem, there is then the question of which subset of customers we should target, in the hope that these customers will like the product, buy it, and recommend it to their neighbors, who will hopefully buy and recommend it in turn.

The analogous mapping to label acquisition for collective classification is as follows. There are nodes (customers) that we need to classify and we have

Table I. The Analogy between Viral Marketing and Active Inference

	Viral Marketing	Active Inference
Objects	Customers	Nodes
States	Bought / Did not buy	Classified correctly / Misclassified
Action	Market a product to a subset of customers	Acquire labels for a subset of nodes
Objective	Maximize the number of customers that buy the product	Maximize the number nodes that are classified correctly
Constraint	A budget for the marketing costs	A budget for acquisition costs

the choice to acquire the labels for (market to) some of them. Our task is to acquire the labels for the right subset of nodes so that the number of correctly classified nodes (the customers who buy the product) in the end is maximized, while minimizing the acquisition cost. This analogy between viral marketing and label acquisition is summarized in Table I.

There are many viral marketing approaches that differ in the formulation of the problem, the assumptions that they make, and the solutions that they offer [Richardson and Domingos 2002; Kempe et al. 2003; Leskovec et al. 2007]. We chose the formulation of Richardson and Domingos [2002]; an advantage of their approach is that it has an exact solution.

In this formulation, we introduce a new random variable T_i for each node V_i, which indicates whether Y_i is predicted correctly. Whether a prediction for a node is correct depends on the informativeness of the node's attributes X_i, whether its neighbors $\mathcal{N}_i$ are classified correctly, and which labels are acquired, $\mathcal{A}$. Following Richardson and Domingos [2002], we make the assumption that this probability is a linear combination of a local probability and a relational probability as follows:

$$P(T_i|\mathcal{N}_i, X_i, \mathcal{A}) \triangleq \beta_i P_l(T_i|X_i, \mathcal{A}) + (1-\beta_i)P_r(T_i|\mathcal{N}_i, \mathcal{A}),$$

where β_i denotes how much the label of a node depends on the node's local attributes versus its neighbors. Here, P_l stands for the local probability, which is defined as:

$$P_l(T_i|X_i, \mathcal{A}) \triangleq \begin{cases} 1 & \text{if } Y_i \in \mathcal{A} \\ \max\limits_{y_k} P(Y_i = y_k|X_i) & \text{otherwise} \end{cases} \tag{3}$$

and P_r stands for the relational probability, which is a linear combination of the statuses of the neighbors:

$$P_r(T_i|\mathcal{N}_i, \mathcal{A}) = \frac{1}{|\mathcal{N}_i|} \sum_{Y_j \in \mathcal{N}_i} T_j.$$

The probability $P(Y_i = y_k|X_i)$ in Eq. (3) can be computed by learning a classifier on the nodes of the train graph G^{tr}.

The objective here is to maximize the total probability of correctly classifying the nodes in the network. With this objective, we define the $utility_{vma}(Y_i)$ to be the increase in this total probability that Y_i causes once it is acquired. To compute $utility_{vma}(Y_i)$, we first calculate two intuitive measures. The first one corresponds to how much a unit change in $P_l(T_i|X_i, \mathcal{A})$ affects the total

probability in the network:

$$\Delta(Y_i) \triangleq \sum_{V_j \in \mathcal{V}} \frac{\partial P(T_j = 1|X_j, \mathcal{A})}{\partial P_l(T_i|X_i, \mathcal{A})}.$$

The second one measures how much an instance's probability of correct classification is increased when we acquire the label for it:

$$\Delta P(Y_i) = \beta_i \left(P_l(T_i|X_i, \mathcal{A} \cup Y_i) - P_l(T_i|X_i, \mathcal{A}) \right).$$

Then, the effect that acquiring a label Y_i will have in the network, i.e., the $utility_{vma}$ of a label is, just a product of the two:

$$utility_{vma}(Y_i) \triangleq \Delta(Y_i)\Delta P(Y_i).$$

We omit some of the details about how to derive these equations. The interested reader can refer to Richardson and Domingos [2002].

With these assumptions, our formulation is the same as that of Richardson and Domingos [2002] with one subtle difference. In the viral marketing domain, when a person is marketed a product, there is still a non-zero probability of that person not buying the product. In label acquisition, however, we assume that we can acquire labels with perfect information; that is, there is no uncertainty about a node's label after we acquire it. Because this particular formulation of viral marketing has an exact solution, we compute the $utility_{vma}$ values for all candidate labels only once, and then we acquire the labels that have the highest utility values. This property of the approach makes it quite fast and thus attractive.

3.3 Reflect and Correct (RAC)

The next method that we introduce 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. We again associate a random variable T_i with each $V_i \in \mathcal{V}$, like we did in the viral marketing formulation, but this time we take a reverse perspective and T_i denotes whether the prediction for Y_i was indeed incorrect. Additionally, instead of using a viral marketing approach to learn and predict T_i, we formulate it as a classification problem by constructing features that are possible indicators of whether a node is misclassified, and learning a classifier to capture the dependence of T_i on the constructed features. Then, the acquisition problem can be solved by running the collective inference on the test graph, predicting which nodes are misclassified, acquiring a label for a central node among the potentially flooded ones, and repeating the process until the budget

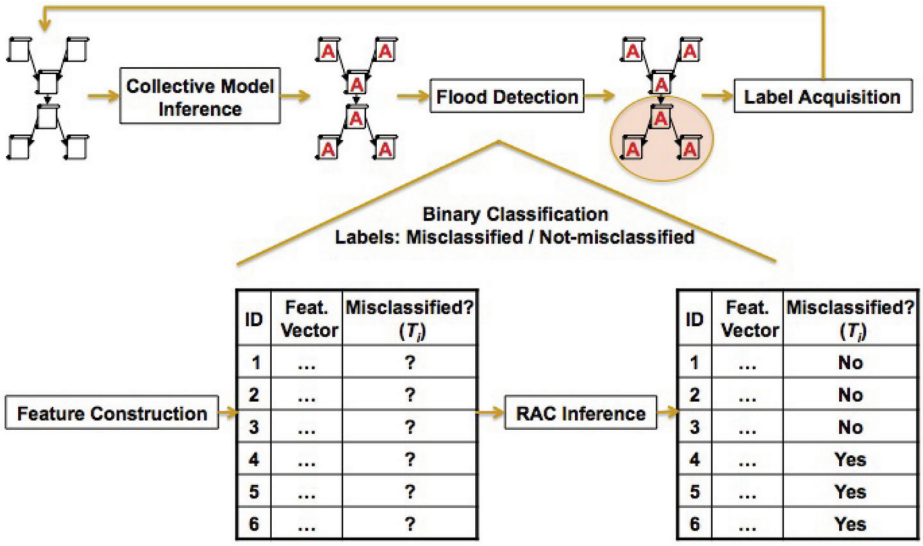


Fig. 1. Active inference using the RAC method. We iteratively label the nodes using the collective model, predict which nodes are misclassified, acquire the central node among the misclassified ones, and repeat the process until the budget is exhausted. To predict which nodes are misclassified, we use 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.

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 to be used for predicting whether a node is misclassified. In this paper, we present the general framework for RAC and construct and experiment with only three simple and intuitive features as examples. The list can be extended with more features; especially, one can think of incorporating domain knowledge as features as well. The first of the three features we constructed is based on the content information of the node, the second one is based on the neighbors of the node, and the last one is based on global statistics. Intuitively, the content indicator captures how much the node attributes disagree with the classification decisions of the collective model. The relational indicator captures how likely it is that the neighbors of a node are also misclassified. Lastly, the global indicator captures how different the posterior distribution of the labels is from the expected prior distribution. We next explain these features in detail and provide their formal definitions.

The *content indicator* measures how far the prediction of the collective model is from the truth according to the attributes. Assume that the collective model predicts $Y_i = y_j$. Then, we define the content indicator for node V_i to be:

$$ci_i \triangleq 1 - P(Y_i = y_j \mid X_i)$$

Again, we can compute $P(Y_i = y_j \mid X_i)$ by learning a local classifier for the nodes of the train graph G^{tr}. The intuition behind the content indicator ci is that if the attributes of a node disagree with the prediction made by the collective

model, then it is a signal for a possible misclassification. The content indicator is a measure of the strength of the disagreement between the local classifier and the collective model. However, the content indicator alone will not be sufficient for misclassification detection; otherwise, we could just replace the collective model with the local classifier.

The *relational indicator* captures how likely it is that a node's neighbors are also misclassified. The intuition is that if a node's neighbors are misclassified, then the node itself is probably misclassified as well (because the classification model is a collective one). There are different possibilities for defining the relational indicator; for instance, it can be defined as a recursive function of T_i, and then it can be computed iteratively. We take the simplest approach and define it as the average of the content indicators, ci_j, of the neighbors of the node V_i.

$$ri_i \triangleq \frac{1}{|\mathcal{N}_i|} \sum_{Y_j \in \mathcal{N}_i} ci_j.$$

Finally, the *global indicator* captures the difference between our prior belief about the class distributions and the posterior distribution that we get based on the predictions. For example, based on our prior belief, if we expect to classify 20% of the nodes with label y_j, but the collective model predicts 60% of the nodes as label y_j, then some of the nodes that are classified as y_j are probably misclassified. Let the prior distribution of the class y_j be denoted by $Prior(y_j)$ and let the posterior distribution based on the predictions of the collective model be denoted by $Posterior(y_j)$. Then, we define the global indicator for the node V_i that is predicted as y_j as follows:

$$gi_i \triangleq \frac{Posterior(y_j) - Prior(y_j)}{1 - Prior(y_j)}.$$

Having constructed these three features, we learn a classifier for estimating the distribution $P(T_i \mid ci_i, ri_i, gi_i)$. To learn this classifier, we need training data, which requires four pieces of information per node: the three indicators described above, 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}. To obtain the content indicator ci_i, we first learn a content-only classifier on the attribute vectors of the nodes of the training graph G^{tr} and then use this content-only classifier and the predicted labels in the prediction graph G^{pr} to compute ci_i. To obtain the relational indicators ri_i, we use the content indicators that we just computed. To obtain the global indicators gi_i, we collect the prior class distribution statistics on the training graph G^{tr} and the posterior class distribution statistics on the prediction graph G^{pr}. Finally, to obtain the value of T_i, we compare the correct labels from the training graph G^{tr} with the predicted labels from the prediction graph G^{pr}. Having constructed the training data, we can use any probabilistic classifier to learn the distribution $P(T_i \mid ci_i, ri_i, gi_i)$. This process is illustrated in Figure 2.

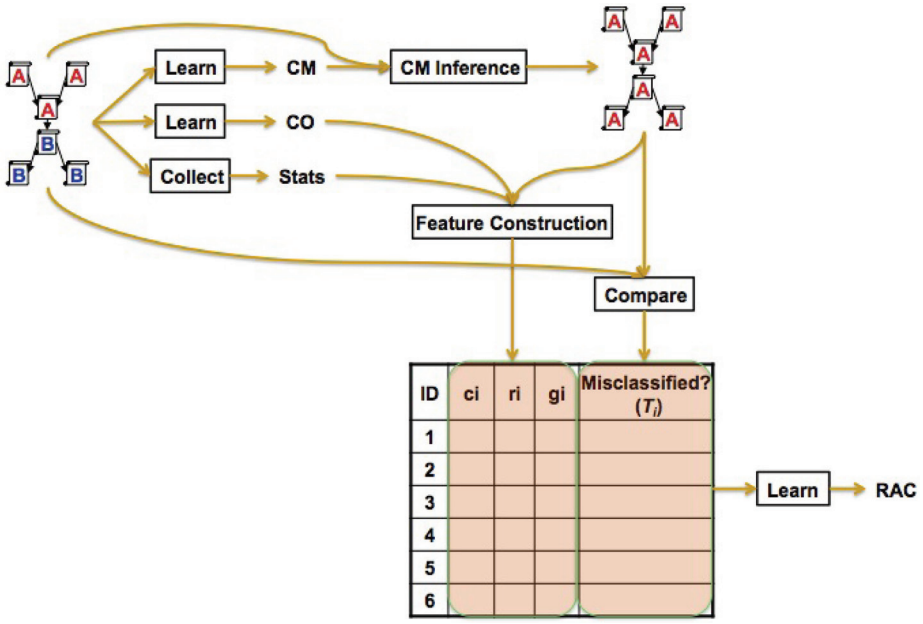


Fig. 2. The process of learning $P(T_i \mid ci_i, ri_i, gi_i)$. We use the collective model CM to predict the labels for the training graph. To construct the content indicator, we use the predicted labels and a content-only classifier that was learned on the training graph. To construct the global indicator, we collect prior class distribution statistics on the training graph and the posterior class distribution statistics on the predicted labels. To obtain the class information T_i, we compare the true labels and the predicted labels.

The question that remains to be answered is how to define the utility of label Y_i given $P(T_i \mid ci_i, ri_i, gi_i)$. The most obvious way is to have $utility_{rac} Y_i \triangleq P(T_i \mid ci_i, ri_i, gi_i)$. 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 $P(T_i \mid ci_i, ri_i, gi_i)$ 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:

$$utility_{rac}(Y_i) \triangleq \begin{cases} 0 & \text{if } P(T_i \mid ci_i, ri_i, gi_i) < \sigma \\ 1 + \sum_{Y_j \in \mathcal{N}_i} \delta(P(T_j \mid ci_j, ri_j, gi_j) > \sigma) & \text{otherwise,} \end{cases}$$

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

We can have a fixed threshold σ, like 0.5, however, we want to set it adaptively based on the training data and the underlying collective model. Specifically, we want σ to be a function of the prior probability of the misclassifications of the collective model on the training data. The way we set it is as follows: let p

be the percentage of the nodes in the training data that are misclassified by the collective model. Then, we first sort all the test data in decreasing order of $P(T_i \mid ci_i, ri_i, gi_i)$ and then we set σ to the misclassification probability of the last node in the top p percent in this sorted list. Instead of continuously updating the σ after each acquisition step, we fix it at this value before any acquisitions are made because we expect the percentage of misclassified nodes in the test graph to decrease as we acquire more labels. With the misclassification predictor learned and the utility function defined, the missing pieces of the acquisition process, which is illustrated in Figure 1, are now complete.

We provided a general framework to perform RAC and also described three example features and an example utility function. The relative merits of each feature and utility function depend on the domain, the noise level in the attributes of the nodes, the strength of the correlation between the node labels, and the degree of class skew present in the data. For example, the benefit of the content indicator correlates negatively with the noise level in the attributes of the nodes; the relational indicator's benefit correlates with the degree of label correlation; the global indicator's benefit correlates with the degree of class skew in the data. Similarly, a utility function that takes the misclassification predictions for neighbors into account is useful for correcting flooded regions, whereas the utility function that takes only the individual scores into account is useful for domains where flooding is not the main problem but instead the focus is to correct single mistakes.

3.4 Generalized Utility-Based Active Inference

In this section, we describe a generic active inference algorithm that unifies the three acquisition methods described above. The algorithm also serves as a generic utility-based active inference technique. In this general algorithm which we formally describe in Algorithm 1, we iteratively find the label Y_i that

Algorithm 1. Generalized utility-based active inference algorithm

Input:
G – the test graph
CM – the learned collective model
c_{ij} – misclassification costs
C_i – the acquisition costs
B – the budget
Output:
$\mathcal{A}$ – the set of acquisitions

$\mathcal{A} \leftarrow \emptyset$
while $C(\mathcal{A}) < B$ **do**
  $Y_{max} \leftarrow nil$
  $maxValue \leftarrow -\infty$
  for $Y_i \in \mathcal{Y} \setminus \mathcal{A}$ **do**
    $utility_{Y_i} \leftarrow utility(Y_i \mid \mathcal{X}, \mathcal{A}, \mathcal{Y} \setminus \mathcal{A}, c_{ij}, CM)$
    if $utility_{Y_i} > maxValue \wedge C(\mathcal{A} \cup \{Y_i\}) \leq B$ **then**
      $maxValue \leftarrow utility_{Y_i}$
      $Y_{max} \leftarrow Y_i$
  $\mathcal{A} \leftarrow \mathcal{A} \cup \{Y_{max}\}$

has the highest utility and whose acquisition cost does not cause us to exceed the given budget, we add it to our acquisition set, and then repeat the process until we exhaust the budget.

The utility function at step six of the algorithm is replaced with $utility_{aiga}$, $utility_{vma}$, and $utility_{rac}$ for the acquisitions we described previously. The utility function in its most general form is a function of the label under consideration Y_i, the set of all attribute vectors $\mathcal{X}$, what has been acquired thus far $\mathcal{A}$, the set of remaining labels $\mathcal{Y} \setminus \mathcal{A}$, the cost model c_{ij}, and the underlying collective model.

This algorithm is very similar to the general utility-based active learning algorithms used by many different techniques such as Lewis and Gale [1994], Melville and Mooney [2004], Roy and McCallum [2001], and Saar-Tsechansky and Provost [2004] with one notable difference; in active learning, we update the underlying classification model at each step. However, in active inference, we assume that we have enough training data to learn the collective model. One thing that might not be obvious from this algorithm is that the utility computation at step six of the algorithm might require running the collective inference for some of the acquisition strategies.

4. EXPERIMENTAL EVALUATION

We begin our experimental evaluation with a study aimed to better understand the effects of flooding in collective classification. Then, we evaluate our proposed label acquisition strategies with a variety of baselines. We evaluate on both synthetic and real-data and perform a relatively comprehensive exploration of options and settings for the algorithms.

4.1 Understanding Flooding

Collective classification models classify a node in a network based on both local attributes and characteristics of the node, such as words in a document, and information contained in the neighboring nodes, such as topics of the documents that reference this document. Thus, a prediction for a node in a given network both affects and depends on the predictions for the other nodes in the network. Due to these structural dependencies, in addition to the typical errors made by the non-collective models (errors due to noise in the data, incorrect modeling assumptions, etc), collective models can make a second type of error by spreading an incorrect decision to the neighboring nodes or by committing to an incorrect decision jointly. As mentioned in the introduction, we call this kind of error *flooding*. Depending on the characteristics of the data and the underlying collective model, flooding can be quite severe; in extreme cases, most of the network can be flooded with just one label.

Before we evaluate different acquisition strategies, we explore the spectrum of flooding as a function of noise in the attributes and correlation of the node labels in the network. In order to quantify flooding, we introduce two measures. The first measure, which we refer to as *perfect information* (PI) accuracy, is the accuracy that corresponds to the setting where to classify a node, the collective model is allowed to look at the true labels (according to the ground-truth) of the node's neighbors instead of the predicted labels. Obviously, this is a hypothetical

situation, but it is quite useful for quantifying floods and comparing different collective models as to how well their modeling assumptions fit to the data and how well they can exploit attribute and neighborhood information. The second measure, which we refer to a *no acquisition* (NOACQ) accuracy, is the accuracy that is achieved before any label is acquired. Given these two measures, we define the *flood percentage* (FP) as the difference between the two:

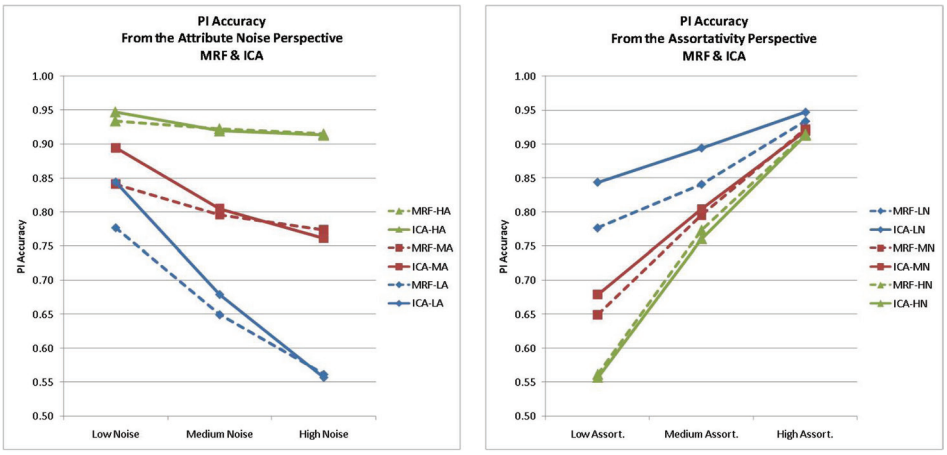
$$FP \triangleq PI - NOACQ.$$

To explore the spectrum of flooding, we generated synthetic data where we change the attribute noise level and the level of the correlation between the labels of the neighboring nodes, which we measure using the assortativity coefficient of Newman [2003]. As our collective models, we used a pairwise Markov Random Field (MRF) [Taskar et al. 2002] and Iterative Classification Algorithm (ICA) [Lu and Getoor 2003a; Neville and Jensen 2000]. For MRF inference, we used loopy belief propagation [Yedidia et al. 2000]. For ICA, we used the count aggregation for feature construction, and used logistic regression as the underlying classifier. We next describe the procedure we used to generate synthetic data.

4.1.1 *Synthetic Data Generation.* We generated synthetic networks using the forest-fire graph generation model [Leskovec et al. 2007]. The forest fire model is shown to exhibit many real-world phenomena such as power law degree distribution, small world effect, and shrinking diameters. However, the forest-fire method, like most random network generators, does not generate labels and attributes for the nodes. In order to label the nodes, we used the method described in Rattigan et al. [2007]. In their method, for each label an initial number of random nodes are selected and labeled with it. Then, an each iteration, nodes that have not received a label yet are labeled based on their neighbors' labels. The number of nodes that are labeled at random at the initial phase of the algorithm controls the assortativity of the network; the higher the initial number of labelings, the less the assortativity. We varied the initial number of labelings to obtain different levels of assortativity.

Rattigan et al. [2007] did not generate attributes for the nodes. We generated attributes using a simple Naive Bayes model after we labeled the nodes. Assuming we have m labels, we generated $m \times k$ binary attributes, a_{ij}, where k is a parameter controlling the total number of attributes generated, i ranges from 1 to m and j ranges from 1 to k; that is, the attributes are grouped by the value of the class label. $P(a_{ij} = True \mid class = l)$ is set to $p > 0.5$ if $i = l$, and it is set to $q < 0.5$ if $i \neq l$. The values of the attributes were sampled by conditioning on the label of the node. In our experiments, we set $m = 5$ and $k = 4$, giving us 20 attributes per node. We varied the p and q parameters to obtain different levels of attribute noise.

4.1.2 *The Spectrum of Flooding.* We generated 10 train-test graph pairs, each with 2000 nodes and with varying attribute noise and assortativity levels. We experimented with three noise levels: low, medium, and high. We distributed these levels uniformly between 0.2 (pure random classification accuracy) and



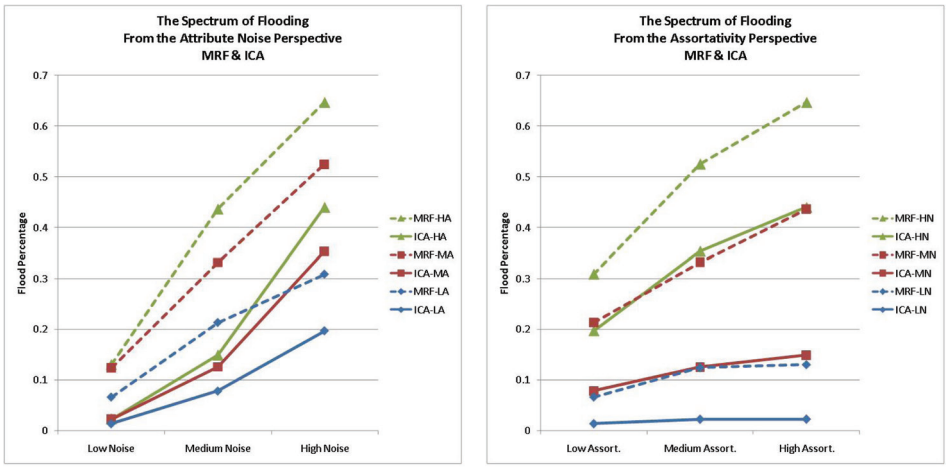
(a) (b)

Fig. 3. Analysis of the PI accuracies achieved by MRF and ICA under low noise (LN), medium noise (MN), high noise (HN), and low assortativity (LA), medium assortativity (MA), and high assortativity (HA) settings. (a) PI accuracy from the attribute noise perspective. (b) PI accuracy from the assortativity perspective.

1; thus, accuracies corresponding to high, medium, and low noise levels are 0.4, 0.6, and 0.8 respectively. Similarly, we distributed the assortativity levels between 0 and 1 uniformly, having 0.25, 0.5, and 0.75 assortativity coefficients for the networks of low, medium, and high assortativity.

Before presenting the flooding percentages of MRF and ICA, we present the PI accuracies they achieved under varying attribute noise and assortativity level settings. The corresponding plots are shown in Figure 3. Not surprisingly, both MRF and ICA achieve the highest PI accuracies under low attribute noise or high assortativity settings. However, when the assortativity level is not high and if the attributes are not very noisy, ICA outperforms MRF in terms of PI accuracy. This result indicates that ICA is possibly better at exploiting the attribute information compared to MRF under comparably low assortative settings. The fact that they achieve comparable PI accuracies when the assortativity level is high regardless of the attribute noise level signals that MRF is able to exploit the neighborhood information at least as well as ICA.

We present the flood percentage results in Figure 4. As one expects, as the attribute noise level increases, the flood percentage also increases, and this is true for both MRF and ICA (Figure 4(a)). Because the attributes are noisy (1) a misclassification is more likely, (2) classification decisions depend more on the labels of the neighboring nodes, and (3) there are not many nodes that can prevent misclassification propagation (nodes whose attributes carry enough signal in the degree that they can be classified using only their attribute information). If we look at the problem from the perspective of assortativity (Figure 4(b)), when the attribute noise is high, the higher the assortativity, the higher the flooding; however, when the attribute noise is low, the assortativity does not play a significant role for flooding. Again, higher assortativity means greater dependence on the neighbors, but when the attribute noise is low, attributes



(a) (b)

Fig. 4. Analysis of the error caused by flooding, measured as the difference between PI accuracy and NOACQ accuracy, under low noise (LN), medium noise (MN), high noise (HN), and low assortativity (LA), medium assortativity (MA), and high assortativity (HA) settings. (a) Flood percentage from the attribute noise perspective. (b) Flood percentage from the assortativity perspective.

also play a significant role in the classification decision. Thus, nodes are less likely to be misclassified and there are enough nodes that can prevent misclassification propagation. Finally, we observe that MRF floods more than ICA does. This observation is also in line with the observation we made earlier that ICA is able to exploit attribute information more than MRF.

With these experiments, we provide only a flavor of the different settings under which flooding occurs. A comprehensive analysis of the flooding is beyond the scope of this article. Having analyzed the flooding, we next show how active inference can be used to detect and correct it.

4.2 Experiments Comparing Different Active Inference Techniques

Next, we move on to the main evaluation of the proposed algorithms. We compared six acquisition methods: the three acquisition methods described earlier (AIGA, VMA, and RAC), two acquisition methods that are based on the structural properties of the network, and the random acquisition (RND) as a baseline. The two structural acquisition methods are: degree (DEG) and k-mediods clustering (KM). DEG ranks the nodes according to their degree in the network and acquires the highest degree ones. The intuition is that the high degree nodes affect more nodes in the network. The KM method, which was proposed by Rattigan et al. [2007] and shown to outperform many other structure-based acquisition strategies, clusters the network into k clusters using k-mediods clustering and acquires the mediods of the clusters. To compute the similarity of nodes, it uses geodesic distances of nodes in the network. The intuition is to spread the acquired labels in the network evenly and to choose central nodes whenever possible.

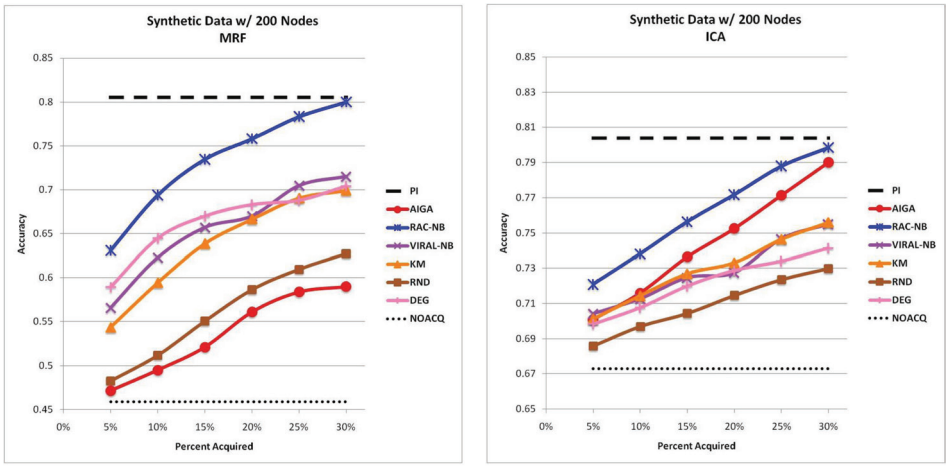
We compare these acquisition strategies on both synthetic and real-world datasets using accuracy as the performance measure. To assess how well each acquisition methods deals with flooding, we also plot PI and NOACQ accuracies. For different acquisition methods, we computed accuracy over the labels that were not acquired (i.e., $\mathcal{Y} \setminus \mathcal{A}$), whereas for PI and NOACQ, we measured accuracy on all the labels $\mathcal{Y}$. Even though PI and NOACQ accuracies are quite useful to know to asses how well the acquisition strategies perform, neither PI is an upper bound nor NOACQ is a lower bound because the acquisition strategies and the PI and NOACQ are evaluated on different sets. An acquisition strategy, in practice, can acquire the labels that even PI misclassifies, thus can surpass PI accuracy. Similarly, an acquisition strategy might acquire labels that even NOACQ does not make mistakes on, thus even though both the acquisition strategy and NOACQ make the same number of mistakes, the acquisition strategy has worse accuracy than NOACQ simply because the acquisition strategy is evaluated on a smaller set ($\mathcal{Y} \setminus \mathcal{A}$ versus $\mathcal{Y}$).

Finally, both RAC and VMA require content-only classifiers, classifiers that use only the attribute and label information for each node independently; RAC requires a content-only classifier for feature construction (the content and relational indicators) and it requires a content-only classifier for predicting whether a node is misclassified given the constructed features, whereas VMA requires a content-only classifier for local probability computation. For RAC feature construction and VMA local probability computation, we used Naive Bayes for the synthetic datasets (because we generated the attributes using Naive Bayes) and we compared using Naive Bayes versus logistic regression for the real-world datasets. To classify nodes as misclassified or not based on the constructed features, we used logistic regression for both synthetic and real-world datasets.

Even though the AIGA method is a polynomial-time algorithm, each single acquisition decision requires running inference for each node and for each possible value of its label. Thus, it is impractical to run AIGA on large networks. We first present a series of results on networks small enough to run AIGA, and then present results on larger networks, which do not include AIGA results.

4.2.1 *Experiments on Synthetic Networks with 200 Nodes.* We trained and tested our collective models on ten training-testing network pairs; the training networks had 2000 nodes in order to learn a reliable collective model and the testing networks had 200 nodes. The nodes had medium noise and assortativity levels. We varied the percentage of labels acquired from 5% to 30% in 5% increments. The results are shown in Figure 5.

One of the striking results is that for MRF, AIGA performed worse than random (Figure 5(a)). This is surprising at first, because one would expect the AIGA method to perform the best. However, recall that we used loopy belief propagation for MRF inference, which is an approximate method for probability computation and it is known to produce suboptimal results when there are short cycles in the graph. Because of the assortativity of the nodes, we observed that beliefs about the nodes' labels reinforced one another iteratively, and thus most of the probability distributions for the nodes' labels were extreme: 1 for one label



(a) (b)

Fig. 5. Experiments comparing AIGA with other methods on small size graphs. (a) MRF results. (b) ICA results.

value and 0 for the other values. Because the probabilities were extreme, AIGA made acquisition decisions based on $utility_{aiga}$ that were extremely similar for many nodes. As for ICA (Figure 5(b)), AIGA performed better than all methods except RAC. This suggests that the probabilities for ICA were better calibrated than for MRF, however, this claim needs further investigation.

As for the time it took for different acquisition methods to complete, at 30% acquisition level for MRF, AIGA took about 38 minutes, RAC took 4 seconds, the other methods took less than a second. For ICA, AIGA took 11 minutes, RAC took 3 seconds, and the other methods again took less than a second. Because AIGA takes much more time and its accuracy depends heavily on the calibration of the probability estimates, AIGA is an undesirable acquisition method. Some approaches to speeding up AIGA are to work with a sample rather than using all of the test data and to acquire more labels at a time, however, these modifications will most likely further reduce its accuracy.

4.2.2 *Experiments on Synthetic Networks with 2000 Nodes.* Next, we compared the acquisition strategies on larger test graphs with 2000 nodes under nine settings: the cross product of the variations in the attribute noise level and assortativity level. For each plot, we zoom in to the area of interest; that is, the low point of the y-axis starts from NOACQ accuracy, and the highest point in the y-axis is the accuracy of either PI or the accuracy achieved by the best performing acquisition method, whichever is higher. The reason we zoom in is to be able to highlight the differences between different acquisition methods.

We present results in the order of high noise, medium noise, and low noise settings. For each noise setting, we vary the assortativity levels. The results for high attribute noise and varying assortativity levels for both MRF and ICA are shown in Figure 6. For MRF, RAC outperforms all other methods at all levels. The differences are statistically significant at all levels for high and medium

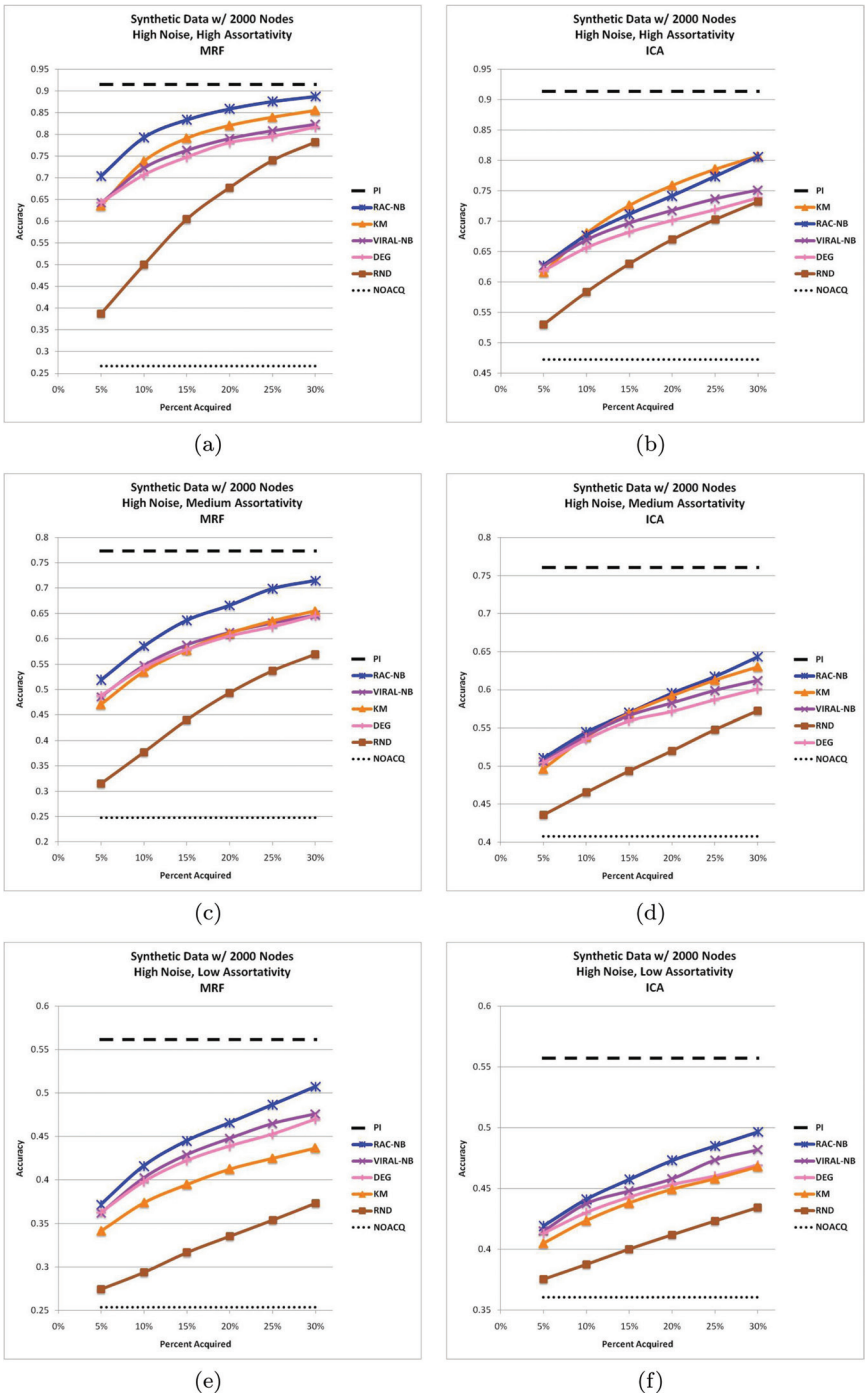


Fig. 6. Accuracy comparisons for the high attribute noise case. (a) and (b) High assortativity, (c) and (d) Medium assortativity, (e) and (f) Low assortativity.

assortativity level, and after 15% acquisition level for the low assortativity case, where significance is measured using paired t-test at 95% confidence level. For ICA, RAC does slightly worse than KM for the high assortativity case, comparable for the medium assortativity case, and slightly better for the low assortativity case; the differences are not statistically significant except at 20% and 30% acquisition levels for the low assortativity case. Note that two of the three features that RAC uses are based on the content-only model's predictions; even though the node attributes are quite noisy and thus the content-only model is quite unreliable, RAC is either comparable or better than the other methods in the high attribute noise case.

When we compare the remaining methods, they all significantly outperform random acquisition in almost all cases (except random has comparable results to DEG and VMA only for ICA when the assortativity is high and if we acquire 30% of the labels). For high assortativity levels, KM significantly outperforms other methods for both MRF and ICA at almost all acquisition levels and this result is in line with the findings of Rattigan et al. [2007]. For medium assortative levels, there is not a clear winner between VMA, KM, and DEG, but when the assortativity level is low, VMA outperforms other methods slightly.

One important observation is that MRF is easier to improve than ICA; even though MRF accuracies start very low compared to ICA due to high flood percentages, MRF accuracies are better than ICA accuracies for all acquisition methods (except RND) starting from 10% acquisition level for high assortativity and starting 15% for medium assortativity. This observation also confirms that MRF is more dependent on neighbor information than ICA.

Next, we present results on the medium attribute noise case in Figure 7. RAC significantly outperforms all other methods at all levels for both MRF and ICA. It is also able to reach beyond PI accuracy at high acquisition levels for medium assortativity and starting 20% acquisition for low assortativity case. As for the other methods, KM again has better accuracy than other methods in the high assortativity case; for the medium assortativity and low assortativity, VMA outperforms other methods, and the differences are statistically significant for most acquisition levels.

The final set of results on low attribute noise synthetic data are shown in Figure 8. Remember that in this setting, the flood percentage was low, especially for ICA. Thus, the remaining errors to correct are the errors due to model imperfection, attribute noise, etc., that is, the difference between 1 and PI accuracy, and this type of error is higher for the low assortativity case. Thus, one would expect the acquisition methods to perform better than PI accuracy in this low attribute noise setting. We observe that RAC outperforms all other methods significantly at all levels for both MRF and ICA. It is also able to perform better than PI accuracy in almost all acquisition levels. As for the other methods, VMA significantly outperforms DEG, KM, and RND in most cases; it is also able to perform better than PI but only in half of the cases. This result is not surprising because VMA makes use of a content-only classifier whereas DEG, KM, and RND do not. And, for the same reason, DEG, KM, and RND are never able to get beyond PI accuracy, except DEG for only the low assortativity case for only MRF.

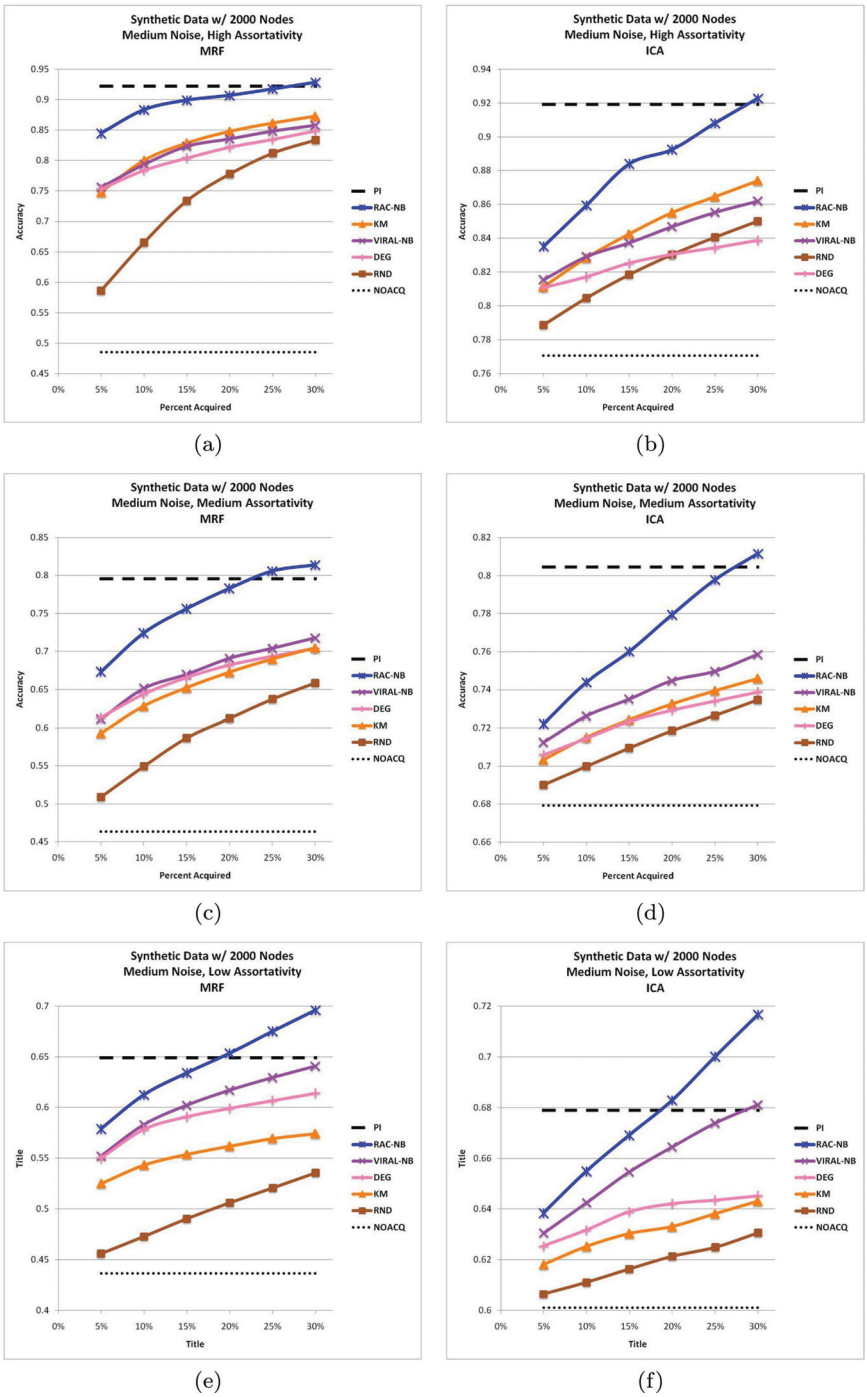


Fig. 7. Accuracy comparisons for the medium attribute noise case. (a) and (b) High assortativity, (c) and (d) Medium assortativity, (e) and (f) Low assortativity.

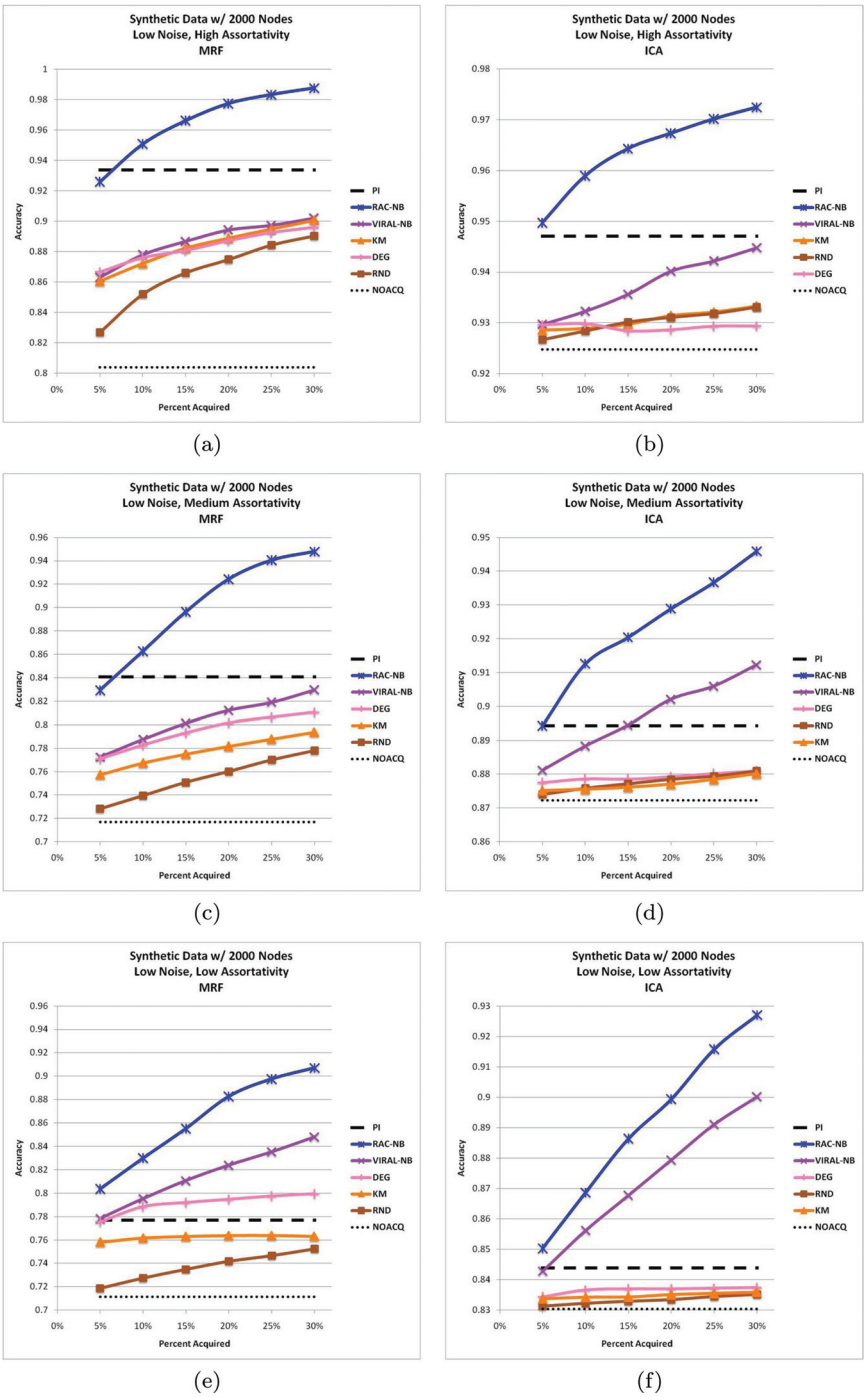


Fig. 8. Accuracy comparisons for the low attribute noise case. (a) and (b) High assortativity, (c) and (d) Medium assortativity, (e) and (f) Low assortativity.

4.2.3 *Experiments on Real-World Datasets.* We experimented on two real publication datasets that are publicly available, the Cora dataset [McCallum et al. 2000] and the CiteSeer dataset [Giles et al. 1998]. The Cora dataset contains 2708 machine learning papers that are divided into seven classes, while CiteSeer dataset has 3312 documents that are divided into six classes.

Our evaluation methodology for these datasets is slightly different from the general practice. In real-world scenarios, we typically have only a small percentage of the data labeled. This makes the interactions between the unlabeled nodes more common than the interactions between the labeled and unlabeled nodes. To mimic these two observations, we adopted the following evaluation strategy. We divided each dataset into three disjoint splits and repeatedly trained on one split and tested on the remaining two (in contrast to training on two splits and testing on the other). Additionally, we did not make use of the edges between the labeled nodes and the unlabeled ones during inference. Because of these changes in the evaluation strategy, which we believe results in a more realistic evaluation, the accuracies corresponding to NOACQ are very low compared to the numbers reported in the literature. The primary reason is that the test graphs are more amenable to flooding now, because they are large and there are no interactions between the test graph and the training graph. However, the PI accuracies are close to the previously reported numbers [Sen et al. 2008], only being slightly lower because we are using less training data.

With these real-world datasets, we also experimented with using different content-only classifiers for RAC and VMA. The purpose of this experiment is to explore whether and how much the strength of the underlying content-only model affects the accuracy results for RAC and VMA. We experimented with using Naive Bayes and logistic regression; the content-only classification accuracies for Naive Bayes were worse than logistic regression accuracies; Naive Bayes had an average accuracy of 0.61 for Cora and 0.57 for CiteSeer, whereas logistic regression accuracies were 0.69 and 0.62 respectively. Finally, the assortativity coefficient for Cora is 0.79, whereas for CiteSeer it is slightly lower with 0.68. The accuracies for different acquisition strategies are shown in Figure 9.

One of the first observations is that MRF again floods more than ICA does on both datasets; the flood percentage for MRF on Cora is 0.43 and for CiteSeer is 0.35 whereas for ICA they are 0.11 and 0.06 respectively. For MRF, both versions of RAC outperform all other methods significantly for both datasets. There are not significant differences between RAC-LR and RAC-NB for MRF, except for Cora at 30% acquisition level. Because MRF floods more than ICA does, this result suggests that RAC might not need a very strong local classifier for detecting the floods. Differences emerge when most of the flooded nodes are corrected; RAC-LR outperforms RAC-NB significantly for MRF only at 30% acquisition level.

For ICA, RAC-LR outperforms all other methods, including RAC-NB, for both datasets. Again, this suggests that when the flood percentage is not high, which is the case for ICA, a stronger local classifier is more helpful. RAC-NB outperforms all other methods, including VMA-LR, for CiteSeer but it has comparable performance to VMA-LR for Cora, while still outperforming the remaining methods.

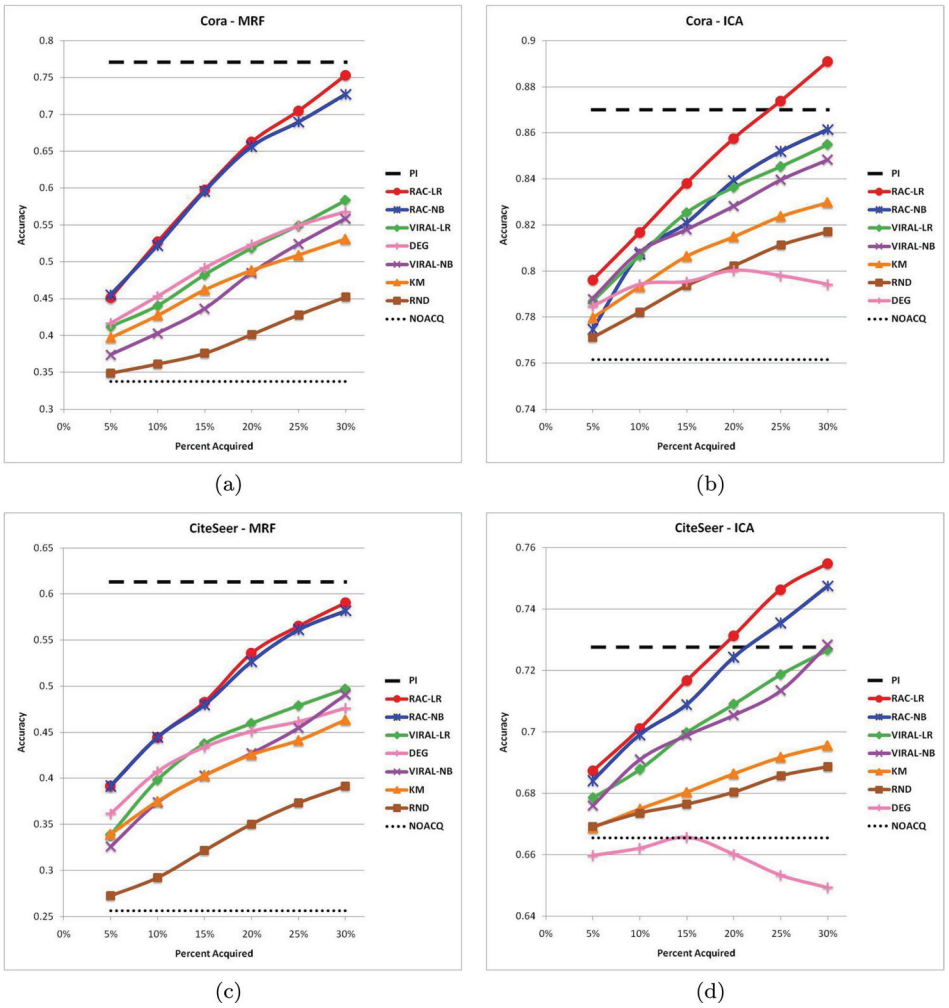


Fig. 9. Experiments on the real-world datasets. (a) MRF results on Cora, (b) ICA results on Cora, (c) MRF results on CiteSeer, and (d) ICA results on CiteSeer.

One of the interesting results is that DEG performed quite poorly for ICA for both Cora and CiteSeer; for Cora, it was initially better than RND but it became worse after 20% acquisition; for CiteSeer, it was always worse than even NOACQ. This observation suggests that, at least in these real datasets, higher degree nodes can be easier to classify for ICA, and thus, acquiring labels for them is not only useless but in fact, we are investing our budget in nodes that we have a high chance of being correctly classified, which we should definitely avoid. Because we evaluated the performance of different acquisition strategies, including DEG, on the labels that were not acquired ($\mathcal{Y} \setminus \mathcal{A}$), acquiring the labels that were already classified correctly made the performance of DEG worse than NOACQ. The reason is that even though the number of misclassified nodes did not change for both DEG and NOACQ, the *percentage* of misclassified nodes

increased for DEG simply because the denominator got smaller ($|\mathcal{Y}|$ for NOACQ versus $|\mathcal{Y} \setminus \mathcal{A}|$ for DEG).

5. RELATED WORK

Substantial research has been done in the area of active learning [Cohn et al. 1994; Seung et al. 1992]. While active learning is related to label acquisition during inference, there are two fundamental differences between existing active learning work and active inference. The first is that the objective of the two problems are different. The aim of active learning is to learn a good model, while active inference assumes that a learned model (or training data) already exists. The second difference is that most of the existing active learning work is on non-graph data, while active inference makes more sense in graph data.

Nonetheless, there are many similarities between the two approaches and some of the active learning techniques could be used for active inference as well. The closest active learning work to approximate inference greedy acquisition (AIGA) method is that of Roy and McCallum [2001]. They estimate the usefulness of a label in terms of the expected error reduction; similarly, AIGA estimates the usefulness of a label in terms of the improvement in the value of the objective function. The closest works to reflect and correct (RAC) are uncertainty sampling [Lewis and Gale 1994] and query by committee [Seung et al. 1992; Freund et al. 1997]. Uncertainty sampling acquires the label that the underlying classifier is most uncertain about, where uncertainty is measured in terms of the posterior probability distribution of the label [Lewis and Gale 1994]. RAC on the other hand, queries the label that the collective model is most likely to misclassify, and the possibility of misclassification is not measured in terms of the probability distribution of the label according to the collective model, but measured using a misclassification prediction classifier. Similarly, the query by committee work samples a committee of classifiers from the set of possible hypotheses using a sampling strategy like Gibbs sampling, and queries the label on which the disagreement of the committee members is the highest [Seung et al. 1992; Freund et al. 1997]. We can think of the content-only classifier and the collective classifier as two committee members, though it is not in the same sense that Seung et al. [1992] used (they do not belong to the same classes of hypotheses). RAC capitalizes on the disagreement between the content-only classifier and the collective classifier, but it is only one of the three features that the misclassification prediction classifier uses.

Another related area to active inference is viral (or targeted) marketing [Richardson and Domingos 2002; Kempe et al. 2003; Leskovec et al. 2007; Provost et al. 2007] where a subset of customers need to be selected for targeted advertisement so as to maximize the product sales. We showed how viral marketing is related to label acquisition and used Richardson and Domingos's model [2002] to compare against. Other models could very well be used and compared against; one of the reasons we chose [Richardson and Domingos 2002] is because the exact solution was tractable. The work in feature-value acquisition during testing [Sheng and Ling 2006; Bilgic and Getoor 2007] is very related

to the label acquisition problem; however, the focus in this field has been on acquiring feature values, not labels.

The most closely related work that we are aware of is that of Rattigan et al. [2007]. They are the first to directly describe label acquisition during inference. They compared different acquisition methods based on network structural measures, such as degree and betweenness, and they suggested a method based on clustering the network. They showed empirically that the clustering method performed the best. They assumed that the nodes did not have any attributes, thus their method did not require any training data. We made different assumptions about the data; that is, the nodes have attributes and we have training data available.

6. SUMMARY AND CONTRIBUTIONS

We have formulated the active inference problem in terms of expected misclassification costs and label acquisition costs and discussed why finding the optimal solution was hard under relatively general assumptions. We discussed the problem of flooding and experimentally showed that it was an important problem for two representative collective classification models: pairwise Markov Random Field (MRF) with loopy belief propagation and Iterative Classification Algorithm (ICA). Through synthetic data, we explored the degree of flooding under varying attribute noise and label assortativity settings. We empirically showed that MRF flooded more than ICA.

We introduced three informed active inference strategies and compared them with two acquisition strategies that are based on structural properties of the network. The first informed active inference strategy that we proposed is based on approximating the value of the objective function through approximate inference and acquiring labels greedily. We experimentally showed that this method did not perform well in practice, especially for MRF. When we analyzed the reasons further, we observed that the probability estimates were not calibrated, which caused the failure of this acquisition method.

Next, we described the analogy between viral marketing and the active inference problem, and introduced a second informed active inference algorithm based on viral marketing. We showed the details of the mapping between active inference and the viral marketing formulation of Richardson and Domingos [2002]. We empirically showed that this method performed equally well with the structural methods under high noise settings, and performed better as the attributes got more useful.

Of the methods that were based on structural properties of the network, the method that acquired labels according to the degree of the nodes had the most erratic performance; sometimes, it performed better than the other structure-based method, K-Mediods, while other times, it performed worse than random. K-Mediods on the other hand performed better than most acquisition methods under high noise and high assortative settings, and it had much more stable performance compared to the degree method in the remaining settings.

Finally, we proposed a third active inference strategy called reflect and correct (RAC) that is based on learning when a collective model makes mistakes

and suggests acquisitions to correct those mistakes. RAC learned a misclassification predictor using three features that we constructed by (1) comparing the predictions of a non-collective classifier and the collective model, (2) using the neighborhood information, and (3) comparing the class prior and posterior distribution statistics on the training and testing networks. We empirically showed that RAC outperformed other methods on most cases, most of the time with statistically significant differences, and it had comparable performance on the remaining few cases, never losing significantly. We also experimented with using Naive Bayes and logistic regression for constructing the features for the misclassification predictor. We showed that when the flooding was significant, RAC did not require a strong content-only classifier; otherwise, logistic regression classifier lead to better results.

7. LIMITATIONS AND FUTURE WORK

One of the limitations of the RAC method is that it is based on the assumptions that the misclassification costs are symmetric and the acquisition costs are uniform. The latter assumption can be lifted by making use of the probabilities that the RAC classifier produces about whether a node is misclassified. However, lifting the first assumption requires further research.

We formally analyzed and experimentally investigated the active inference problem for collective classification approaches, however, active inference can be used for related areas such as semi-supervised learning [Chapelle et al. 2006] and viral marketing [Kempe et al. 2003; Leskovec et al. 2007; Provost et al. 2007; Richardson and Domingos 2002], where flooding also occurs. For example, flooding is acknowledged to occur in graph-based semi-supervised learning techniques such as Gaussian random fields [Zhu et al. 2003] and graph mincuts [Blum and Chawla 2001].

8. 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.

REFERENCES

BILGIC, M. AND GETOOR, L. 2007. VOILA: Efficient feature-value acquisition for classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 1225–1230.

BILGIC, M. AND GETOOR, L. 2008. Effective label acquisition for collective classification. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, 43–51.

BLUM, A. AND CHAWLA, S. 2001. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the International Conference on Machine Learning*. 19–26.

CHAKRABARTI, S., DOM, B., AND INDYK, P. 1998. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. ACM, New York, 307–318.

CHAPELLE, O., SCHÖLKOPF, B., AND ZIEN, A., Eds. 2006. *Semi-Supervised Learning*. MIT Press, Cambridge, MA.

COHN, D., ATLAS, L., AND LADNER, R. 1994. Improving generalization with active learning. *Mach. Learn. 15*, 2, 201–221.

COHN, D. A., GHAHRAMANI, Z., AND JORDAN, M. I. 1996. Active learning with statistical models. *J. Artif. Intell. Res. 4*, 129–145.

FREUND, Y., SEUNG, H. S., SHAMIR, E., AND TISHBY, N. 1997. Selective sampling using the query by committee algorithm. *Mach. Learn. 28*, 2, 133–168.

GETOOR, L., FRIEDMAN, N., KOLLER, D., AND TASKAR, B. 2002. Learning probabilistic models of link structure. *J. Mach. Learn. Res. 3*, 679–707.

GETOOR, L., SEGAL, E., TASKAR, B., AND KOLLER, D. 2001. Probabilistic models of text and link structure for hypertext classification. In *Proceedings of the IJCAI Workshop on Text Learning: Beyond Supervision*. ACM, New York, 24–29.

GILES, C. L., BOLLACKER, K. D., AND LAWRENCE, S. 1998. CiteSeer: An automatic citation indexing system. In *Proceedings of the ACM Conference on Digital Libraries*. ACM, New York, 89–98.

GILKS, W. R., RICHARDSON, S., AND SPIEGELHALTER, D. J. 1996. *Markov Chain Monte Carlo in Practice*. Interdisciplinary Statistics. Chapman & Hall/CRC.

HOWARD, R. A. 1966. Information value theory. *IEEE Trans. Syst. Sci. Cybernet. 2*, 1, 22–26.

JENSEN, D., NEVILLE, J., AND GALLAGHER, B. 2004. Why collective inference improves relational classification. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, 593–598.

JORDAN, M. I., GHAHRAMANI, Z., JAAKKOLA, T. S., AND SAUL, L. K. 1999. An introduction to variational methods for graphical models. *Mach. Learn. 37*, 2, 183–233.

KEMPE, D., KLEINBERG, J., AND TARDOS, E. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, 137–146.

KRAUSE, A. AND GUESTRIN, C. 2005. Optimal nonmyopic value of information in graphical models—efficient algorithms and theoretical limits. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 1339–1345.

LAFFERTY, J. D., MCCALLUM, A., AND PEREIRA, F. C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*. 282–289.

LESKOVEC, J., ADAMIC, L. A., AND HUBERMAN, B. A. 2007. The dynamics of viral marketing. *ACM Trans. Web 1*, 1, 5.

LESKOVEC, J., KLEINBERG, J., AND FALOUTSOS, C. 2007. Graph evolution: Densification and shrinking diameters. *ACM Trans. Knowl. Discov. Data 1*, 1, 177–187.

LEWIS, D. D. AND GALE, W. A. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, 3–12.

LU, Q. AND GETOOR, L. 2003a. Link based classification. In *Proceedings of the International Conference on Machine Learning*. 496–503.

LU, Q. AND GETOOR, L. 2003b. Link-based classification using labeled and unlabeled data. In *Proceedings of the ICML Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*.

MACSKASSY, S. AND PROVOST, F. 2003. A simple relational classifier. In *Proceedings of the ACM Workshop on Multi-Relational Data Mining*. ACM, New York.

MACSKASSY, S. AND PROVOST, F. 2007. Classification in networked data: A toolkit and a univariate case study. *J. Mach. Learn. Res. 8*, 935–983.

MCCALLUM, A. AND NIGAM, K. 1998. Employing EM and pool-based active learning for text classification. In *Proceedings of the International Conference on Machine Learning*. 350–358.

MCCALLUM, A. K., NIGAM, K., RENNIE, J., AND SEYMORE, K. 2000. Automating the construction of internet portals with machine learning. *Inf. Retrieval 3*, 2, 127–163.

MCDOWELL, L., GUPTA, K. M., AND AHA, D. W. 2007. Cautious inference in collective classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 596–601.

MELVILLE, P. AND MOONEY, R. J. 2004. Diverse ensembles for active learning. In *Proceedings of the International Conference on Machine Learning*. 584–591.

NEVILLE, J. AND JENSEN, D. 2000. Iterative classification in relational data. In *Proceedings of the SRL Workshop in AAAI*.

NEWMAN, M. E. J. 2003. Mixing patterns in networks. *Phys. Rev. E 67*, 2, 026126.

PROVOST, F., MELVILLE, P., AND SAAR-TSECHANSKY, M. 2007. Data acquisition and cost-effective predictive modeling: targeting offers for electronic commerce. In *Proceedings of the ACM International Conference on Electronic Commerce*. 389–398.

RATTIGAN, M., MAIER, M., AND JENSEN, D. 2007. Exploiting network structure for active inference in collective classification. In *Proceedings of the ICDM Workshop on Mining Graphs and Complex Structures*. 429–434.

RICHARDSON, M. AND DOMINGOS, P. 2002. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, 61–70.

RICHARDSON, M. AND DOMINGOS, P. 2006. Markov logic networks. *Mach. Learn. 62*, 1-2, 107–136.

ROY, N. AND MCCALLUM, A. 2001. Toward optimal active learning through sampling estimation of error reduction. In *Proceedings of the International Conference on Machine Learning*. 441–448.

SAAR-TSECHANSKY, M. AND PROVOST, F. 2004. Active sampling for class probability estimation and ranking. *Mach. Learn. 54*, 2, 153–178.

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.

SEUNG, H. S., OPPER, M., AND SOMPOLINSKY, H. 1992. Query by committee. In *Proceedings of the ACM Annual Workshop on Computational Learning Theory*. ACM, New York, 287–294.

SHENG, V. S. AND LING, C. X. 2006. Feature value acquisition in testing: a sequential batch test algorithm. In *Proceedings of the International Conference on Machine Learning*. 809–816.

TASKAR, B., ABBEEL, P., AND KOLLER, D. 2002. Discriminative probabilistic models for relational data. In *Proceedings of the Annual Conference on Uncertainty in Artificial Intelligence*. 485–492.

TONG, S. AND KOLLER, D. 2002. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res. 2*, 45–66.

XIANG, R. AND NEVILLE, J. 2008. Pseudolikelihood em for within-network relational learning. In *Proceedings of the IEEE International Conference on Data Mining*. IEEE Computer Society Press, Los Alamitos, CA, 1103–1108.

YEDIDIA, J., FREEMAN, W. T., AND WEISS, Y. 2000. Generalized belief propagation. In *Neural Information Processing Systems*. 689–695.

ZADROZNY, B. AND ELKAN, C. 2001. Learning and making decisions when costs and probabilities are both unknown. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 204–213.

ZHU, X. AND GHAHRAMANI, Z. 2002. Learning from labeled and unlabeled data with label propagation. Tech. Rep. CMU-CALD-02-107, Carnegie Mellon University.

ZHU, X., GHAHRAMANI, Z., AND LAFFERTY, J. D. 2003. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the International Conference on Machine Learning*. 912–919.

Received March 2009; revised July 2009; accepted August 2009