On the Trade-Off Between Iterative Classification and Collective Classification: First Experimental Results

Tayfun Gürel and Kristian Kersting

Machine Learning Lab
Institut für Informatik
University of Freiburg
Georges-Koehler-Allee 79 79110, Freiburg Germany
{guerel|kersting}@informatik.uni-freiburg.de

Abstract. There have been two major approaches for classification of networked (linked) data. *Local* approaches (iterative classification) learn a model locally without considering unlabeled data and apply the model iteratively to classify unlabeled data. *Global* approaches (collective classification), on the other hand, exploit unlabeled data and the links occurring between labeled and unlabeled data for learning. Naturally, global approaches are computationally more demanding than local ones. Moreover, for large data sets, approximate inference has to be performed to make computations feasible.
In the present work, we investigate the benefits of collective classification based on global probabilistic models over local approaches. Our experimental results show that global approaches do not always outperform local approaches with respect to the classification accuracy. More precisely, the results suggest that global approaches considerably outperform local approaches only for low ratios of labeled data.

1 Introduction

Networked data has a key importance since many real world data can be represented in networks of data containing nodes. The most typical example of networked data is the world wide web. Web pages can be modeled as data instances, which are nodes of a directed graph. Hyperlinks give graph characteristics to billions of web pages, hence, gives us the opportunity to represent the world wide web as a huge network. The set of scientific publications exhibits network characteristics, as scientific publications cite each other. Groups of interacting people (social networks), transmission of epidemic diseases are examples of networked data, where the nodes represent human beings. Recently, there has been an increasing interest in mining networked data, see i.e. Lise Getoor's (2003) overview on link mining.

The problem of labeling networked data can be stated as follows:

Given: An undirected graph (V, E), where E is the set of edges and each node in V corresponds to a vector of features $A_1, ..., A_n, C$, where C denotes the class attribute; the values for the A_i are known for all $v \in V$, but class labels for C are only known for a proper subset T of V . We will call a node also a **data instance**.

Find: The class labels for $U := V - T$.

Several approaches to solve the *networked data labeling problem*:

Type 1: The **AVL** approach to tackling this problem is to treat this as a classical learning problem, where the nodes L are the examples and the nodes in L have to be classified (while ignoring E). The traditional AVL approach is, however, incapable of using the network features. (**type 1**: AVL approach)

Type 2: The **iterative classification** scheme (Chakrabarti et. al. 1998, Neville and Jensen 2000, Macskassy and Provost 2003, Lu and Getoor 2003a) employs a traditional AVL learner on the nodes in L but enriches the features $A_1, ..., A_n$ with additional features $F_1, ..., F_m$, which are derived from the neighborhood of the node in E. Examples of F_i include the existence of a neighbor of a particular class c. Only the labeled data is employed in learning, but in contrast to the previous method, the structure of the network is employed. The resulting classifiers is then iteratively applied on the unlabeled nodes. (**type 2**: Iterative Classification)

Type 3: The **collective classification** (Taskar et al. 2001, Taskar et al. 2002, Lu and Getoor 2003b) approach treats the problem as a global optimization problem, in which a particular function, e.g. the maximum likelihood w.r.t. the overall nodes V , is maximized. In this framework, one typically employs the Expectation Maximization (EM) algorithm (Dempster et al.,1977) over the features $A_1, ..., A_n, F_1, ..., F_m$. So, in this framework, both labeled and unlabeled data are employed, as well as the structure of the network. (**type 3** : Collective Classification)

Here, the approaches are listed according to their complexity:

> **Type 1** *approaches employ less information than* **type 2**, *and* **type 2** *less information than* **type 3**

Consequently, learning of **type 3** models can be expected to be computationally more demanding and algorithmically more complex than learning **type 2** and **type 1** approaches. Identifying the cases in which the higher computational costs of **type 3** approaches pay off is an interesting research question.

Our main contribution is such an experimental investigation showing that *the additional complexity of* **type 3** *approaches based on naive Bayes only pays off for low ratios of labeled data.*

This is somewhat surprising as Lu and Getoor (2003b) report that **type 3** approaches based on logistic regression improve the classification/labeling performance of **type 1** and **type 2** approaches for any ratio.

The paper is structured as follows. After reviewing related work, we will introduce a collective classification algorithm, which makes use of structural information as well as that from unlabeled nodes to improve the classification performance. The algorithm is similar to the one employed by Lu and Getoor (2003b). Instead of logistic regression, however, we use a Naive Bayes approach. We will argue that the algorithm can be viewed to learn a global probabilistic model. More precisely, the model can be represented as a Probabilistic Relational Model (Getoor et al, 2001). Before concluding, we will present our experimental results. We compare our algorithm with Naive Bayes as baseline approach (**type 1**) and iterative Naive Bayes classification (Neville and Jensen, 2000) (**type 2**).

2 Related Work

In the last few years, there has been a lot of interesting work on labeling networked data. They can be roughly divided into two approaches, *iterative classification* and *collective classification*. We will now discuss each of them in turn.

Influence propagation by Iterative Classification: Collective inference can be traced back to Chakrabarti et al. (1998). Based on collective inference, they employ a local relaxation labeling algorithm, in a sub-graph around the entity to be classified. Computations of the probability distributions are based on the naive bayes classifier.

Neville and Jensen (2000) present an iterative classification algorithm for relational data. They introduce the concept of dynamic attributes, whose values are subject to change, according to the classification results. For the task of predicting the company type in a relational corporate data set, one example of dynamic attribute might be the dominating type of the companies that the owner of a particular company owns. A naive bayes classifier is employed, which is trained on fully labeled data. The trained classifier is eventually used iteratively to label unseen examples. Neville and Jensen report an improvement of approximately 12 % in the accuracy.

Macskassy and Provost (2003) introduce Relational Neighborhood classifier (RN), which classifies the entities of a graph. RN starts with a partially labeled graph and completes labels by simply computing the weighted counts of each class among the neighbors (weighted with the weight of the edge for each neighbor) and by selecting the class that gives the maximum weighted count. There is also an iterative version of the RN classifier, which repeatedly applies RN classification as its inner loop until convergence.

Collective Classification approaches: Getoor et. al.(2001) develop a probabilistic relational model (PRM) to classify web pages. The PRMs are extensions of Bayesian networks, which were developed for relational databases. PRMs are expressive enough to compute the influence of the labels and the words on the hyperlinked web pages on the label of another web page.

Taskar et. al. (2002) employ relational Markov networks for collective classification tasks. A relational Markov Network defines undirected dependencies and joint distributions for relational databases. When unrolled, a relational Markov network instantiates a Markov network, which defines the dependencies among the entities and their attributes. Similar to the PRMs, learning the Markov networks is based on the expected count of the values for each attribute. Inference is performed with belief propagation as in the PRMs.

Lu and Getoor (2003b) present an EM like algorithm employing logistic regression to make use of link structure for classification of networked data. They enrich the feature space with the ones from the link structure and employ iterative learning of a logistic regression classifier.

3 Collective Classification Using the Expectation Maximization Algorithm

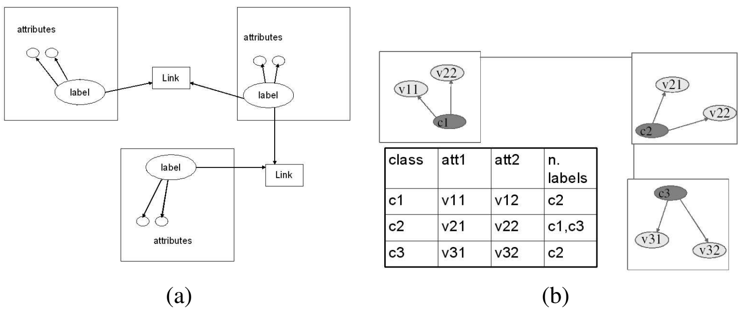
The Expectation Maximization (EM) algorithm is a popular iterative algorithm to learn the parameters of a probabilistic model in case of missing data. The idea of using the EM algorithm for learning from unlabeled data is not new. Nigam et al. (2000) employ the EM algorithm together with the naive bayes classifier for text classification making use of the unlabeled data. In this work, we extend the naive bayes-EM algorithm to the relational case, by introducing additional features from network structure.

3.1 Generation of Second Order Attributes

In order to exploit the network structure, we create **second order attributes**, which hold the information about the labels of the neighboring instances. Since the information about the neighboring labels is not complete due to missing labels, second order attributes may have missing values. We applied two settings for second order attribute generation.

Neighboring Labels Our first setting for second order attribute generation is based on a graphical generative model, which is a Bayesian network. We treat the attributes and the class labels of the instances as random variables and assume that the attributes of an instance are mutually independent, given its class label (naive bayes assumption).

Additionally, we assume the existence of a link random variable between two instances (only if they are linked by an edge). Note that this is only a conceptual random variable, since we know the value of the random variable will always be 'yes' ('yes':if the edge exists and 'no': if the edge does not exist). This random variable cannot take the value 'no', because the link random variables are not included in the Bayesian network for non-existent edges. Note that this is not a complete generative probabilistic model, because we discard the influence of the non-existence of an edge.



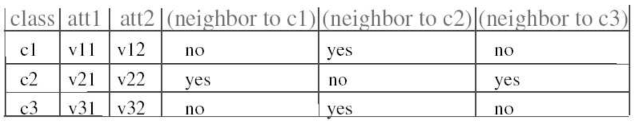
class	att1	att2	n. labels
c1	v11	v12	c2
c2	v21	v22	c1,c3
c3	v31	v32	c2

Fig. 1. (a) The underlying probabilistic model (b) Second Order Attribute Generation: Neighboring Labels

Figure 1 illustrates a mini dataset (b) and the corresponding Bayesian network (a). In the figure a square represents a data instance with its attributes and class label. Edges between squares stand for links.

From the Bayesian network, we can infer that attributes of the instances are independent of the attributes of the other instances and the values of the link random variables are independent of each other, given the class labels. To learn the parameters of this Bayesian network, we can make use of the fact that the the parameters will be the same for each link random variable and for the attributes of the each instance. It is, therefore, possible to learn the parameters for the nodes of the same type together. This idea is analogous to that of PRMs (Getoor et al., 2001). Learning the parameters of the Bayesian network can be achieved by flattening the data into a table, with an additional attribute. We call this attribute "**Neighboring Labels**". We extract all the labels of the neighboring instances and use these labels as the values of the "Neighboring Labels". Since it is possible that one instance has more than one neighboring instance, this second order attribute is a multi-set valued attribute. An example of second order attribute generation with the ”Neighboring Labels” setting is given in Figure 1b. In this example, for the sake of simplicity, we assume that there are 2 attributes and 3 instances. Squares represent the instances, whereas the connections between the squares represent the links. In this example c_i denote the i^{th} class label. The flattened table is also shown in the same figure.

With a complete training set, the class dependent attribute probabilities (model parameters), could be found by counting in this table and they would correspond to the parameters of the Bayesian network. The parameters for the attribute nodes ($p(v_{ak}|c_j)$, where a is an index over attributes, k is an index over values and j is an index over class labels) would correspond the parameters on the first order attributes on the table. The Bayesian network parameters of the Link nodes would correspond the parameters on the second order attribute from the table, i.e. $p(Link =' yes'|c_j, c_k) = p(NeighborClass = c_k|c_j)$, since links are undirected.



class	att1	att2	(neighbor to c1)	(neighbor to c2)	(neighbor to c3)
c1	v11	v12	no	yes	no
c2	v21	v22	yes	no	yes
c3	v31	v32	no	yes	no

Fig. 2. Second Order Attribute Generation: Existence Attributes

Existence Attributes The second setting for second order attribute generation is the generation of the **existence attributes**. These second order attributes hold information about existence of links to instances from each class label. For example, if instance A has link to an instance, class label of which is c_j, then, the value of the attribute ExistsLinkTo-c_j will be yes. Otherwise, the value of this attribute will be no. Doing so for each class label, the number of classes times second order attributes are generated. An example second order attribute generation in this setting is shown in Figure 2. The data is the same as in Figure 1b. $P(there\ is\ a\ link\ to\ class\ c_n|c_j))$.

3.2 Maximum Likelihood Parameters and the EM Algorithm

Our goal is to optimize the parameters of the model globally by finding the maximum likely parameters, given the data. According to Bayes’ law, the likelihood of the parameters can be written as: $P(\theta|D) = P(D|\theta)P(\theta)/P(D)$.

Dropping $P(D)$ from the expression will not change the maximizing parameters since it is not conditional on θ. We assume that the prior probability for each parameter set is equal. Therefore, we can drop $P(\theta)$ from the expression too. Since the logarithm is a monotonic function, the maximum likely parameter set is given as follows: $\theta' = \arg\max_\theta \log(P(D|\theta))$.

In our case, all the other nodes of the Bayesian network are independent of each other if the class labels are given. Therefore, the likelihood of the data can be written as follows (In a Bayesian network, a node is independent from its non-descendants given its parents):

$$\begin{aligned} P(D|\theta) =& \{\prod_{i=1}^{N} P(c(i)|\theta) \prod_{a=1}^{|A|} P(v_{ia}|c(i);\theta)\} \\ &* \{\prod_{l=1}^{|L|} P(c(l_1)|c(l_2);\theta)\} \end{aligned} \tag{1}$$

In the above equation, i is an index over the instances, N is the total number of instances, a is an index over the attributes, and $|A|$ is the number of attributes. $c(l_1)$ and $c(l_2)$ denote the class labels on two sides of a link.

We employ the EM algorithm, since the values of the second order attributes are partially missing. We introduce the hidden variables z_{ij}. i denotes an index over the instances, and j denotes an index over the classes. The true value of z_{ij} is 1 if instance i belongs to class c_j; 0 otherwise. The computation of the expected value z_{ij} corresponds to finding the probability that the instance i belongs to class c_j. With the introduction of the hidden variables, the likelihood in Equation (1), can be rewritten as follows:

$$\begin{aligned} P(D|\mathbf{Z};\theta) =& \{\prod_{i=1}^{N}\prod_{j=1}^{|C|}\prod_{a=0}^{|A|} (P(v_{ia}|c_j;\theta)P(c_j|\theta))^{z_{ij}}\} \\ &* \{\prod_{d_i \in D}\prod_{d_m < d_i}\prod_{j=1}^{|C|}\prod_{n=1}^{|C|} P(Link =' yes'|c_n, c_j;\theta)^{t_{im} z_{ij} z_{mn}}\} \end{aligned} \tag{2}$$

Applying the logarithm yields:

$$\begin{aligned} \log P(D|\mathbf{Z};\theta) =& \{\sum_{i=1}^{N}\sum_{j=1}^{|C|}\sum_{a=0}^{|A|} z_{ij}\log(P(v_{ia}|c_j;\theta)P(c_j|\theta))\} \\ &+ \{\sum_{i=1}^{N}\sum_{m<d}\sum_{j=1}^{|C|}\sum_{n=1}^{|C|} t_{im} z_{ij} z_{mn} \log P(Link =' yes'|c_n, c_j;\theta)\} \end{aligned} \tag{3}$$

In the above equation, i and m are indices over the instances, N is the total number of instances, j is an index over the class labels, c_j is the j^{th} class label, v_{ia} is the value of the a^{th} attribute, a is an index over the attributes. t_{im} is defined as 1 if there is a link between the i^{th} and the m^{th} instances; and 0 else. Note that $P(Link =' yes'|c_n, c_j; \theta) = p(c_n|c_j) = p(c_j|c_n)$. The two latter parameters can be found by counting in the flattened table.

Equation (3) is composed of merely sum of logs and computable in closed form (Dempster et al, 1977). The expectation and the maximization step of the algorithm are as follows (The formulas follow our first setting of attribute generation, which consists of constructing a set valued second order attribute, namely "Neighboring Labels"):

Expectation

$$\begin{aligned} E^{(k+1)}[z_{ij}] &= P(classLabel = c_j | d_i, L, E^{(k)}[\mathbf{Z}]) \\ &= \alpha p(c_j) \prod_a p(v_{ia}|c_j) \prod_{u \in Neighbors(i)} (\sum_{t=1}^{|C|} E^{(k)}[z_{ut}] p(c_t|c_j)) \end{aligned} \tag{4}$$

In the above equation $E^{(k+1)}[z_{ij}]$ denotes the expected value of the hidden variable z_{ij} at step $k+1$. d_i denotes the first order attribute values of the instance i. L denotes the set of links. $E^{(k)}[\mathbf{Z}])$ stands for the expected values of the hidden variables at step k. v_{ia} stands for the value of the a^{th} attribute of instance i. Please note that the probability is computed under naive bayes assumption that the attributes are independent of each other. This assumption is also made for second order attributes as well. $p(NeighborClass = c_t|c_j) = p(c_t|c_j)$ is the probability that an instance from class c_t is situated at one side of a link, given that an instance from c_j is situated on the other side. This corresponds to the probability that the second order attribute value is c_t given that class label is c_j.

Maximization The maximization step corresponds to the learning the model classifier parameters based on the expected values of the hidden variables.

$$p(v_{ak}|c_j) = \frac{\hat{N}(v_{ak}, c_j)}{\hat{N}(c_j)} = \frac{\sum_i^N E[z_{ij}] \delta(v_a - v_{ak})}{\sum_i^N E[z_{ij}]} \tag{5}$$

$$\begin{aligned} p(c_l|c_k) &= \frac{\hat{N}(c_k, c_l)}{\hat{N}(c_k, DontCare)} \\ &= \frac{\sum_{i=1}^N \sum_{m=1}^N (E[z_{ik}]E[z_{ml}]t_{im} + E[z_{il}]E[z_{mk}]t_{im})}{\sum_{j=1}^{|C|} \sum_{i=1}^N \sum_{m=1}^N (E[z_{ik}]E[z_{mj}]t_{im} + E[z_{ij}]E[z_{mk}]t_{im})} \end{aligned} \tag{6}$$

$$p(c_j) = \frac{\hat{N}(c_j)}{N} = \frac{\sum_i^N E[z_{ij}]}{N} \tag{7}$$

The maximization step is nothing more than finding the Naive Bayes parameters by counting. Since the exact counts are missing, the expected counts are used. $\delta(v_a - v_{ak})$ is defined as 1 if $v_a = v_{ak}$; 0 else. t_{im} denotes the existence of a link between the instances i and m. By $\hat{N}(c_k, c_l)$, we point to the expected counts of the links, which have class labels c_k and c_l on its two sides. By $\hat{N}(v_{ak}, c_j)$, we point to the expected counts of the instances, the a^{th} attributes of which have values v_{ak}. N denotes the total number of instances. $\hat{N}(c_j)$ denotes the expected count of the instances, which have class labels c_j.

```
– Input: a set of labeled instances $T$, a set of unlabeled instances $U$ and a set of links $L$.
– Learn a local naive bayes model on $T$ ignoring the links
– for each node $i$ in $U$
    • for each possible class label $c_j$
        ∗ compute $P(label(i) = c_j | Attributes(i))$ using the local naive bayes model
    • end for
– end for
– repeat
    • Learn the Bayesian network parameters (M-Step, Equations (5), (6), (7))
    • for each node $i$ in $U$
        ∗ for each possible class label $c_j$
            · Compute $E[z_{ij}] = P(label(i) = c_j | d_i, L, E^{(k)}[\mathbf{Z}])$ (E-Step, Equation (4))
        ∗ end for
    • end for
– until distributions over possible class labels converge
– for each node $i$ in $U$
    • $label(i) := \arg\max_{c_j} P(label(i) = c_j)$
– end for
– Output: The labels of the nodes in $U$
```

Table 1. The (soft) EM algorithm for collective classification

4 Experimental Results

We compare the performances of **type 1** (naive bayes), **type 2** (iterative naive bayes with second order attributes), **type 3** (collective classification via the EM, see table 1) approaches. For **type 3**, we also implemented a hard version of the

EM algorithm, which finds most likely values of the hidden values instead of expected values at each iteration. We investigate whether **type 3** approach has a significant improvement on **type 2** approach. The **type 2** approach uses the same features (second order features) as the **type 3** approaches, but instead of global optimization, learns from labeled data only and applies the learned model to unlabeled data iteratively. We also compare **type 2** and **type 3** approaches to **type 1** approach.

We ran our experiments on three datasets: Gene (KDD-Cup 2001, http://www.cs.wisc.edu/ dpage/kddcup2001/), Cora (McCallum et al, 2000) and CiteSeer (Lu and Getoor, 2003a) datasets. The Gene dataset contains 1242 examples. The task is to predict the localization of the protein in the cell (among 15 locations), given values of six attributes (some of which may be missing) and given the interaction network between proteins. There are 1806 interactions among proteins. The Cora data set contains 4187 examples. The examples are machine learning papers, which are categorized into seven topics. The Cora dataset contains 6185 citations. The CiteSeer dataset includes around 3600 computer science papers in six categories as well as 7522 citations. For Cora and CiteSeer datasets, we used the document frequency pruned dictionaries (Lu and Getoor, 2003a). The pruned dictionary for Cora has 1400 words and the pruned dictionary for CiteSeer has 3000 words.

For each data set and for each second order attributes setting, we ran the four algorithms for different ratios of labeled data, ranging from 0.1 to 0.8 . Each experiment was repeated 5 times. Each time, the instances are randomly initialized as labeled or unlabeled, according to the probability resulting from labeled data ratio. For example if the 80 % percent of the data should be labeled, the probability of any instance being initially labeled is 0.8. We performed paired t-tests (95% significance level), to assess the significance of outperformances. The results are presented in Figure 3. With our second order attributes, naive bayes (**type 1**) is significantly outperformed by the iterative and collective classification algorithms (**type 2** and **type 3**) significantly. Whether **type 3** algorithms can outperform the **type 2** algorithm does strongly depend on the labeled data ratio. If the unlabeled data ratio is equal or below a threshold (0.4-0.5), **type 3** algorithms are no more significantly better than the **type 2** algorithm (Exception: Cora Dataset with existence attributes where threshold is 0.7).

Lu and Getoor (2003b) did a similar analysis with an EM-like iterative logistic regression algorithm. In contrast to our results for Naive Bayes, Lu and Getoor report that **type 3** algorithm improve the predictive performance of **type 1** and **type 2** approaches for any ratio of labeled data. Explaining the different results is an open and interesting future research question.

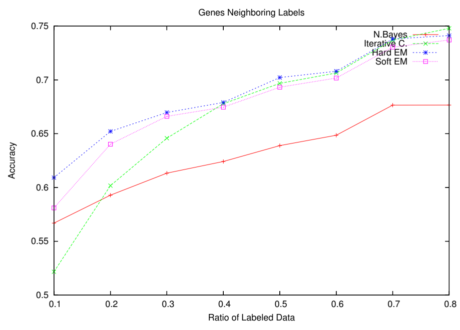
5 Conclusions

We experimentally compared global and local approaches for labeling/classifying networked data. To do so, we devised a simple global algorithm based on Naive Bayes and EM making use of labeled and unlabeled data. Our experimental results suggest that global approaches improve the performance of corresponding local ones only for low ratios of labeled data.

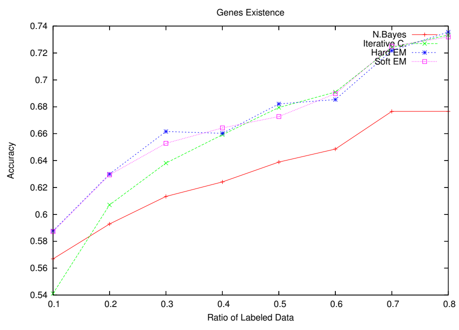
Acknowledgements We would like to thank Luc De Raedt and Lise Getoor for valuable discussions. Further thanks to Lise Getoor and Prithviraj Sen for the CiteSeer and Cora data sets. This work was partially supported by the European Union IST programme, contract no FP6-508861, Applications of Probabilistic Logic Programming 2.

References

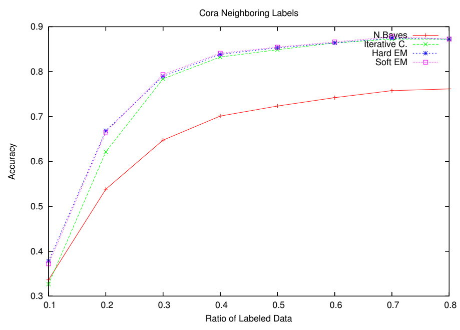
1. Chakrabarti, S., Dom B., Indyk, P.: Enhanced hypertext classification using hyperlinks. Proc. ACM SIGMOD (1998)
2. Dempster, A. P., Laird, N. M., Rubin, D. B.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B. **39** (1977) 1–38
3. Getoor, L., Segal, E., Taskar, B., Koller, D.: Probabilistic Models of Text and Link Structure for Hypertext Classification. IJCAI Workshop on ”Text Learning: Beyond Supervision”. Seattle, WA. (2001)
4. Getoor, L.: Link Mining: A New Data Mining Challenge. SIGKDD Explorations. volume 5, issue 1. (2003)
5. Lu, Q. & Getoor, L.: Link-based Classification. International Conference on Machine Learning. Washington, DC. (2003)
6. Lu, Q. & Getoor, L.: Link-based Classification using Labeled and Unlabeled Data. ICML Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining. Washington, DC. (2003)
7. Macskassy, S. A. & Provost, F.: A Simple Relational Classifier. ICML Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining. Washington DC. (2003)
8. McCallum, A., Nigam, K., Rennie, J., Seymore, K.: Automating the Construction of Internet Portals with Machine Learning. Information Retrieval Journal. **3** (2000) 127–163
9. Neville, J. & Jensen, D.: Iterative Classification in Relational Data. In Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data. (2000)
10. Nigam, K., McCallum, A., Thrun, S., Mitchell T.: Text Classification from Labeled and Unlabeled Documents using EM. Machine Learning. **39(2/3)** (2000) 103–134
11. Taskar, B., Segal, E., Koller, D.: Probabilistic Clustering in Relational Data. Seventeenth International Joint Conference on Artificial Intelligence (IJCAI01). Seattle, Washington. (2001)
12. Taskar, B., Abbeel, P., Koller, D.: Discriminative Probabilistic Models for Relational Data. Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI02). Edmonton, Canada (2002)



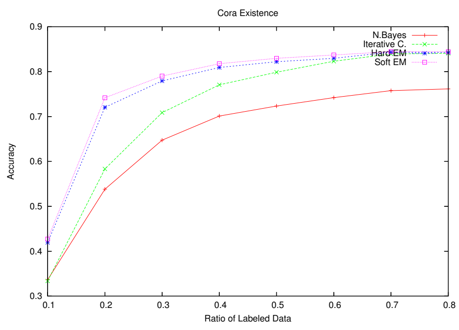
(a)



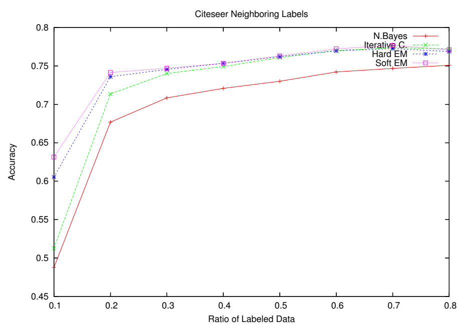
(b)



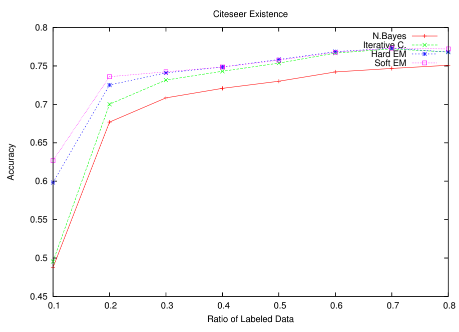
(c)



(d)



(e)



(f)

Fig. 3. Hard EM and (Soft) EM outperform the Iterative Classification **significantly** (t-test, 95%) only below a threshold of labeled data ratio (a) Hard EM threshold $= 0.4$, Soft EM threshold $= 0.4$ (b) Hard EM threshold $= 0.4$, Soft EM threshold $= 0.3$ (c) Hard EM threshold $= 0.3$, Soft EM threshold $= 0.4$ (d) Hard EM threshold $= 0.7$, Soft EM threshold $= 0.7$ (e) Hard EM threshold $= 0.5$, Soft EM threshold $= 0.3$ (f) Hard EM threshold $= 0.6$, Soft EM threshold $= 0.4$