Empirical Comparison of Approximate Inference Algorithms for Networked Data

Prithviraj Sen SEN@CS.UMD.EDU
Lise Getoor GETOOR@CS.UMD.EDU
Department of Computer Science, University of Maryland, College Park, MD 20742.

Abstract

Over the past few years, a number of approximate inference algorithms for networked data have been put forth. We empirically compare the performance of three of the popular algorithms: loopy belief propagation, mean field relaxation labeling and iterative classification. We rate each algorithm in terms of its robustness to noise, both in attribute values and correlations across links. A novel observation from our experiments is that loopy belief propagation faces difficulty when inferring over data with *homophily*, a common type of link correlation observed in relational data.

1. Introduction

Over the past few years, relational classifiers have gained considerable attention. Due to the efforts of various researchers we now have access to numerous approaches (e.g., Lafferty et al., 2001; Taskar et al., 2002; Lu & Getoor, 2003; Taskar et al., 2003; Neville & Jensen, 2004 etc.) each with its own set of advantages. However, inference in networked data is still a challenging issue. The consensus seems to be that *exact inference is a hard problem* in this domain and cannot be achieved without exploiting special properties of the application or data at hand. In part due to this reason, even the earliest efforts in relational classification have shown a tendency to *develop approximate inference algorithms* (e.g., Hummel & Zucker, 1983; Chakrabarti et al., 1998). Due to the collective efforts of the research community we now have a considerably long list of approximate inference algorithms to choose from. In this paper, we aim to empirically compare three of the most popular approximate inference procedures used in relational classification to date.

Even though relational classifiers have gained attention only in the past five to seven years, the general problem of inference for structured output spaces has received attention for a considerably longer period of time from various research communities such as computer vision, spatial statistics and natural language processing, in particular. Thus, it is not surprising that one of the earliest principled approximate inference algorithms, *Relaxation Labeling* (RL) (Hummel & Zucker, 1983), was developed by researchers in computer vision. Due to its simplicity and appeal, RL was a topic of active research for some time and many researchers developed different versions of the basic algorithm (Li et al., 1994). In this paper, we experiment with a particularly simple version of RL referred to as the *Mean Field* (MF) approach (Yedidia et al., 2005; Li et al., 1994).

During the last decade, researchers working with an iterative decoding scheme known as ”Turbo Codes” (Berrou et al., 1993) began to report excellent results with another approximate inference algorithm termed *Loopy Belief Propagation* (LBP) or the *Sum Product* algorithm (Kschischang & Frey, 1998; McEliece et al., 1998; Kschischang et al., 2001) which corresponds to running Pearl's belief propagation algorithm (Pearl, 1988) on networks with loops. It wasn't long before the machine learning community took notice and reported that LBP can return good results under certain conditions (Murphy et al., 1999). LBP has since been justified as a variational method (Yedidia et al., 2000) and the basic algorithm now has numerous extensions (Yedidia et al., 2005). LBP is another approximate inference procedure we include in our suite of experiments.

Recently, various researchers in the machine learning community began reporting reasonable results with another approximate inference algorithm that we will refer to as *Iterative Classification Algorithm* (ICA) (Neville & Jensen, 2000; Lu & Getoor, 2003; Carvalho & Cohen, 2005). Besag (1986) originally proposed this greedy approach as a simple means of performing inference in Markov random fields for spatial statistics and called it *Iterated Conditional Modes* (ICM). ICA, in fact, is more general than ICM since one can use different classifiers in conjunction with

Presented at the ICML Workshop on Open Problems in Statistical Relational Learning, Pittsburgh, PA, 2006. Copyright 2006 by the author(s)/owner(s).

it as each of Neville and Jensen (2000); Lu and Getoor (2003); Carvalho and Cohen (2005) show on various relational datasets. Due to its extreme simplicity, different researchers tend to rediscover this algorithm. In addition, it often returns useful results. ICA is the third inference algorithm we include in our experiments.

Besides the above mentioned inference algorithms, there are many other algorithms that we did not consider for empirical testing such as Yuille (2002), Wainwright et al. (2005) and Minka (2001) or exact inference algorithms like *Belief Propagation* (Pearl, 1988).

The aim of our experiments was to determine how the algorithms perform in the most commonly occurring settings. All our experiments are on synthetically generated random graph data. We used a power-law graph generating algorithm (Bollobas et al., 2003) to create the graphs. Each node in our synthetic datasets is described by a set of attributes which functions as observed evidence. Correlations across links are varied over a range of settings.

The main assumption in relational classification is that the labels of linked nodes are correlated, a phenomenon also referred to as *autocorrelation* (Jensen & Neville, 2002), and that these correlations can be utilized during the classification process. One of our contributions is to show that different types of correlations across links can affect the performance of various inference algorithms. One example of a specific type of correlation is *homophily* (Lazarsfeld & Merton, 1954; McPherson et al., 2001) also referred to as *perfect assortative mixing* (PAM) (Newman, 2003) where nodes with the same label exclusively link with each other. Another type of correlation occurs when nodes prefer to link with nodes of the opposite label, Newman refers to this phenomenon as *perfect disassortative mixing* (PDM). In our experiments, we test the inference algorithms on datasets ranging from PDM to PAM.

We rate each approximate inference algorithm according to its resilience to noise, both in attribute values and correlations across links. Our experiments also tested the algorithms' performance across varying link densities. We found that MF is particularly prone to local minima while LBP was sensitive to PAM. We also found that LBP's convergence does not necessarily indicate good results. ICA seemed to be the most consistent of all the three algorithms returning reasonable accuracies across all settings albeit LBP has the ability to produce slightly better results than ICA under certain settings such as low link density.

Ours is not the only empirical comparison of approximate inference algorithms. Murphy et al. (1999) empirically tested the performance of LBP on some well known directed probabilistic graphical models, e.g., QMR-DT etc., but does not compare LBP with other approximate inference algorithms. Weiss (2001) compares the performance of MF and LBP on some simple toy undirected graphical models. Macskassy and Provost (2005) compare various inference procedures on datasets devoid of any attribute values; in their experiments they provide evidence by providing a subset of the nodes with their known labels.

Undirected graphical models or *Markov networks* (Cowell et al., 1999) have been shown to be an effective way to represent relational classification problems and correlations due to the link structure. All our experiments were performed on Markov networks and we next describe the preliminary notation required to describe the various approximate inference algorithms.

2. Preliminaries

Let $\mathbf{V}$ be a set of discrete random variables, and let $\mathbf{v}$ be an assignment of values to the random variables. A Markov network is described by a graph $G = (\mathbf{V}, E)$ and a set of parameters Ψ. Let $C(G)$ denote a set of (not necessarily maximal) cliques in G. For each $c \in C(G)$, let V_c denote the nodes in the clique. Each clique c has a clique potential $\psi_c(V_c)$ which is a non-negative function on the joint domain of V_c and let $\Psi = \{\psi_c(V_c)\}_{c \in C(G)}$. For classification problems we are often interested in conditional models. Let $\mathbf{X}$ be the set of observed random variables we condition on and let $\mathbf{x}$ denote the observed values of $\mathbf{X}$. Let X_c denote the observed random variables in clique $c \in C(G)$ and let x_c denote the observed values of X_c. Let $\mathbf{Y}$ be the set of target random variables to which we want to assign labels and let $\mathbf{y}$ denote an assignment to $\mathbf{Y}$. Let Y_c denote the set of target random variables in clique $c \in C(G)$ and let y_c denote an assignment to it. A *conditional Markov network* or *conditional random field* is a Markov network (G, Ψ) which defines the distribution $P(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C(G)} \psi_c(x_c, y_c)$ where $Z(\mathbf{x}) = \sum_{\mathbf{y}'} \prod_c \psi_c(x_c, y'_c)$.

Pairwise Markov networks are Markov networks whose set of cliques $C(G)$ consist of the set of nodes in the network and the set of edges. For simplicity, in this paper, all our experiments were performed on pairwise Markov networks.

Conditional Markov networks, as presented above, are not suited for relational classification tasks since they involve clique specific potentials $\psi_c(V_c)$. Taskar et al. (2002) suggest that instead of providing clique specific potentials we can employ a small set of feature functions (usually indicator functions for class labels etc.) and define the potentials in log-space $\log \psi_c(y_c, x_c) = \sum_i w_i f_i(x_c, y_c)$ where f_i is the i^{th} feature function and w_i is a parameter which needs to be estimated.

To perform inference one needs to compute the complete labeling $\text{argmax}_{\mathbf{y}}\ P(\mathbf{y}|\mathbf{x})$ for the given Markov network. Unless the underlying Markov network has special properties (e.g., it is a tree, a sequence or a network with a low treewidth) exact inference may be infeasible. For most relational classification problems, the Markov network might consist not only of thousands of nodes but may also be densely connected. Hence the need for approximate inference algorithms. In the next section we describe the three inference algorithms we will be comparing in this paper.

Estimating the parameters Ψ of the Markov network is possible from fully labeled training data. Taskar et al. (2002) show that Markov networks can be trained using first-order optimization methods like conjugate gradient descent that require the computation of a gradient. Taskar et al. also show that the gradient can be estimated by performing inference on the underlying Markov network of the training data. We used the approximate inference algorithms described in the next section to train the Markov networks.

3. Approximate Inference Algorithms

We compared the performance of three approximate inference algorithms and here we describe each algorithm. For simplicity, we describe each algorithm in terms of pairwise Markov networks. In each of the algorithms, we use α to denote a proportionality constant and $\phi_i(y)$ to denote the exponentiated sum of the feature counts weighted by the parameters of the attribute values of node $Y_i \in \mathbf{Y}$.

3.1. Iterative Classification Algorithm (ICA)

The basic idea behind ICA, as the name suggests, is to perform inference in an iterative fashion, in each iteration visiting each $Y \in \mathbf{Y}$ re-estimating its class conditional probabilities given the labels on its neighbors and its attribute values. Algorithm 1 describes the algorithm we used for our experiments.

Algorithm 1 Iterative Classification Algorithm

```
 1: for each $Y_i \in \mathbf{Y}$ do {Bootstrapping}
 2:     $b_i(y) \leftarrow \alpha\phi_i(y)$ such that $\sum_y b_i(y) = 1$
 3:     $y_i \leftarrow \text{argmax}_y\ b_i(y)$
 4: end for
 5: repeat {perform message passing}
 6:     for each $Y_i \in \mathbf{Y}$ do
 7:         $b_i(y) \leftarrow \alpha\phi_i(y)\exp\{\sum_{(Y_i,Y_j)\in E}\sum_{y'} w_{y,y'}\delta_{y_j}(y')\}$ such that
            $\sum_y b_i(y) = 1$
 8:         $y_i \leftarrow \text{argmax}_y\ b_i(y)$
 9:     end for
10: until all $y_i$ stop changing
```

In Algorithm 1, the inference is initialized by labeling all the nodes by their attribute-only labels. Then we proceed with the iterations of ICA.

In ICA, one has the freedom of choosing the order in which to visit each $Y \in \mathbf{Y}$. We refer the reader to Getoor (2005) where numerous orderings are compared and shown not to have a significant impact on the final result.

3.2. Mean Field Relaxation Labeling (MF)

As we mentioned in Section 1, there are many versions of Relaxation Labeling (RL). Here, we chose to experiment with the Mean Field version (MF). The basic structure of MF is almost identical to ICA performing the inference in an iterative fashion, where, in each iteration we visit each node, relabeling it depending on the neighborhood. The main difference lies in the way the class conditional probabilities are updated. Instead of using the neighborhood class labels (like ICA), MF uses the class conditional probabilities of the neighborhood to update the current node's probabilities. Algorithm 2 describes the MF algorithm we used for our experiments.

Algorithm 2 Mean Field

```
1: for each $Y_i \in \mathbf{Y}$ do {initialize message}
2:     $b_i(y) \leftarrow 1$
3: end for
4: repeat {perform message passing}
5:     for each $Y_i \in \mathbf{Y}$ do
6:         $b_i(y) \leftarrow \alpha\phi_i(y)\exp\{\sum_{(Y_i,Y_j)\in E}\sum_{y'} w_{y,y'} b_j(y')\}$ such that
           $\sum_y b_i(y) = 1$
7:     end for
8: until all $b_i(y)$ stop changing
```

Notice that MF is simply the "soft" version of ICA. We refer the reader to Weiss (2001) for details about the justification of MF as a variational method.

3.3. Loopy Belief Propagation (LBP)

Yedidia et al. (2000) show that LBP can be justified as a variational method. To see the connection between LBP and MF one needs to dig a little deeper. Since performing inference using the correct distribution imposed by the Markov network is too hard, variational methods use a much simpler trial distribution to approximate the Markov network distribution. Usually, the trial distribution is such that once it is estimated then we can simply read off the node specific class conditional probabilities. Yedidia et al. (2005) show that the difference between MF and LBP lie in their choice of trial distributions. LBP has more variables in its trial distribution and can thus provide a better approximation to the Markov network distribution. We review the LBP algorithm in Algorithm 3 and refer the reader to Yedidia et al. for more details. In Algorithm 3, $\mathcal{N}(Y)$ denotes the neighborhood of Y.

4. Synthetic Data Generation

Commonly available real-world networks exhibit properties like preferential attachment with degrees following

Algorithm 3 Loopy Belief Propagation

```
1: for each $Y_i \in \mathbf{Y}$ do {initialize messages}
2:   for each $Y_j \in \mathbf{Y}$ such that $(Y_i, Y_j) \in E$ do
3:     $m_{i \to j}(y) \leftarrow 1$
4:   end for
5: end for
6: repeat {perform message passing}
7:   for each $Y_i \in \mathbf{Y}$ do
8:     for each $Y_j \in \mathbf{Y}$ such that $(Y_i, Y_j) \in E$ do
9:       $m_{i \to j}(y) \leftarrow \alpha \sum_{y'} \phi_i(y') \exp(w_{y,y'}) \prod_{Y_k \in \mathcal{N}(Y_i) \setminus Y_j} m_{k \to i}(y')$
         such that $\sum_y m_{i \to j}(y) = 1$
10:    end for
11:  end for
12: until all $m_{i \to j}(y)$ stop showing any change
13: for each $Y_i \in \mathbf{Y}$ do {compute beliefs}
14:   $b_i(y) \leftarrow \alpha \phi_i(y) \prod_{(Y_i, Y_j) \in E} m_{j \to i}(y)$ such that $\sum_y b_i(y) = 1$
15: end for
```

power-law distributions and correlations amongst the labels across links. Since our aim is to find out how approximate inference algorithms perform on such networks, we chose to model our synthetic data generation algorithm (Algorithm 4) along the lines of the evolutionary network model described in Bollobas et al. (2003).

4.1. Growing the Data Graph

The synthetic data generator (Algorithm 4) "grows" a graph from an empty set of nodes. The number of nodes in the final graph is controlled by the parameter n. α is a parameter which controls the number of links in the graph. Roughly, the final graph should contain $\frac{n}{1-\alpha}$ links. For all our experiments, we set $n = 300$. For our experiments, α is an important parameter and we were interested in finding out how the various inference algorithms perform with varying link density.

4.2. Introducing Correlations in the Link Structure

For simplicity, we generate binary class data (label $\in \{0,1\}$) using our synthetic data generator. Algorithm 4 proceeds through iterations and in each iteration it either connects a newly created node to the graph or connects two existing nodes in the graph. Each time Algorithm 4 creates an edge, it makes a call to Algorithm 5. Algorithm 5 implements a rudimentary form of *preferential attachment* where a node can choose which nodes to link based on their labels. This introduces correlations amongst the labels across links. The strength of these correlations is controlled by the parameter ρ. Each node can link to nodes of its own class with probability ρ. With probability $1-\rho$, a node can choose to link to a node of the other class. In addition, nodes with higher out-degree have a higher chance of getting linked to. This introduces the power-law degree distribution commonly observed in most real-world networks. We refer the interested reader to Bollobas et al. (2003) for

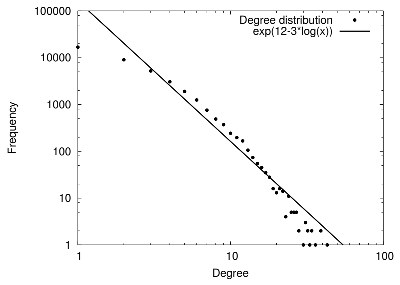

Figure 1. Degree distribution of a graph consisting of 40000 nodes and 53313 edges created by our synthetic data generator plotted in log-scale.

more details regarding this aspect of our synthetic data generation algorithm. In Figure 1, we plot the degree distribution of a graph consisting of 40000 nodes and 53313 edges created by our synthetic data generator. In our experiments, we vary ρ to study the effects of varying correlations across links on the various inference algorithms.

Algorithm 4 Synthetic data generator

```
SynthGraph($n$, $\alpha$, $\rho$, $\omega$)
1: $i \leftarrow 0$
2: $G \leftarrow \emptyset$
3: while $i < n$ do
4:   sample $r \in [0,1]$ uniformly at random
5:   if $r <= \alpha$ then
6:     $v \leftarrow$ select any node uniformly at random from $G$
7:     connectNode($v$, $G$, $\rho$)
8:   else
9:     add a new node $v$ to $G$
10:    choose $v.label$ from $\{0,1\}$ uniformly at random
11:    connectNode($v$, $G$, $\rho$)
12:    $i \leftarrow i + 1$
13:  end if
14: end while
15: for i = 1 to n do
16:   $v \leftarrow i^{th}$ node in $G$
17:   genAttributes($v$, $\omega$)
18: end for
19: return $G$
```

4.3. Attribute Generation

After generating the graph, we generate attributes for each node (`genAttributes`). The total number of attributes is controlled by the parameter *vocabSize* and each node can have a maximum of *numObs* attributes. In order to generate a simple skewed attribute distribution, we use the following simple model. For each node, we sample attributes from noisy class-specific binomial distributions. The noise in the attributes is controlled by the parameter ω. With probability ω, we sample an attribute uniformly from the set $\{0, \ldots, vocabSize - 1\}$. With probability $1-\omega$, we sample an attribute from the distribution $Binomial(p = 1/3, vocabSize - 1)$ if the node belongs to class 0 and

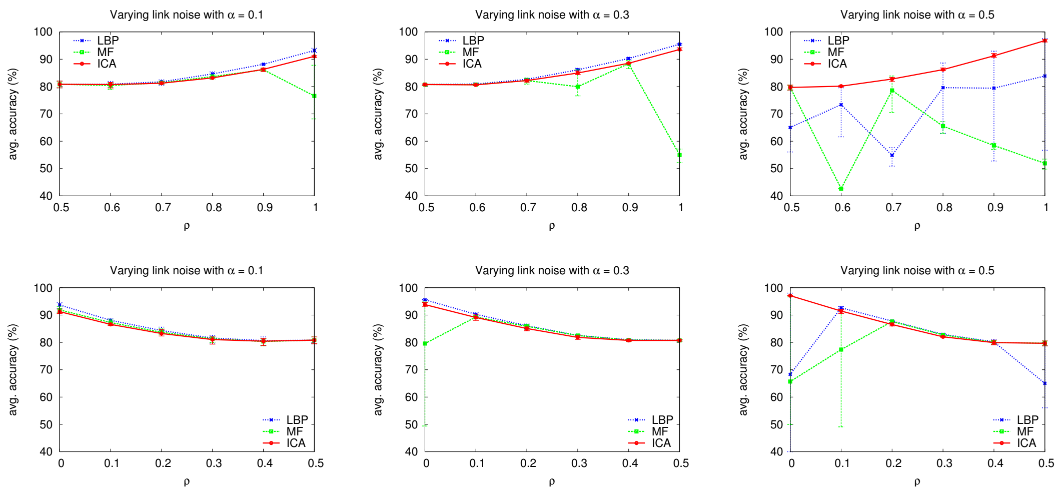

Figure 2. Effects of varying correlations across links: In all of these experiments we varied ρ and set $\omega = 0.3$. In the top row, we provide the results of varying ρ from 0.5 (no link correlations) to 1 (PAM). In the bottom row, we provide the results of varying ρ from 0.5 (no link correlations) to 0 (PDM).

$Binomial(p = 2/3, vocabSize - 1)$ if the node belongs to class 1. For each node, we perform *numObs* such samplings. For all our experiments we set $vocabSize = 5$ and $numObs = 4$. These parameter settings were chosen to bring out the differences amongst the various approximate inference algorithms and changing the settings produces expected changes in the results, e.g., increasing the value of *vocabSize* will introduce more discriminative power in the attribute values, causing all three inference algorithms to produce the same results. ω is another parameter that we vary to study the effects of varying attribute noise.

Algorithm 5 Generating an edge in the synthetic data graph

```
connectNode($v$, $G$, $\rho$)
1: sample $r \in [0,1]$ uniformly at random
2: if $r \leq \rho$ then
3:     $c_n \leftarrow v.label$
4: else
5:     $c_n \leftarrow (v.label + 1) \bmod 2$
6: end if
7: $w \leftarrow$ select a node from $G$ with $w.label = c_n$ and probability
   of selection proportional to its out-degree
8: introduce an edge from $v$ to $w$
```

5. Experiments

Our experiments were aimed to find out how the various inference algorithms perform in the presence of different underlying structure and different sources of noise.

5.1. Experimental Setup

For each experiment, we generated three datasets using our data generator and performed three-fold cross validation. We report average accuracies (with errorbars).

For each classifier, we assumed a ”shrinkage” prior and compute the MAP estimate of the parameters. More precisely, we assumed that different parameters are a priori independent and define $p(w_i) = \lambda w_i^2$. We tried a range of regularization constants for each classifier and found that $\lambda = 10$ returned the best results. Taskar et al. (2002) report using a regularization constant of the same magnitude $\lambda \approx 5.5$.

5.2. Effects of varying Correlations across Links

In our first set of experiments, we study the effects of varying correlations across links. Recall that ρ controls how nodes choose neighbors during the synthetic data generation. More specifically, at $\rho = 0.5$ nodes link to neighbours randomly, at $\rho = 1$ nodes link to neighbours with similiar class labels (perfect assortative mixing or PAM) and at $\rho = 0$ nodes link to neighbours with the opposite class label (perfect disassortative mixing or PDM). In all these experiments we kept ω constant at 0.3.

We divide this set of experiments into two parts, in the first part we vary ρ from 0.5 to 1.0 and in the second part, we vary ρ from 0.5 to 0. Since we are also interested in the effects of varying the link density (α), for each experiment, we produce three plots, each with a different setting for α. Recall that the number of links is governed approximately as $\frac{n}{1-\alpha}$, thus the number of links increase as we increase α.

The top row of plots in Figure 2 shows the results of varying ρ from 0.5 (no correlations) to 1 (PAM) for three different settings of α. At $\alpha = 0.1$, we have sparse data graphs and

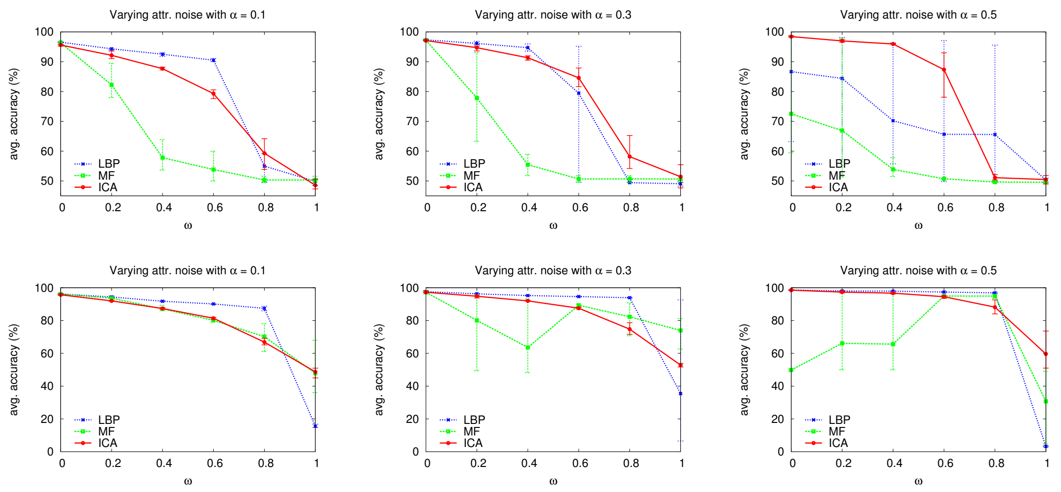

Figure 3. Effects of varying attribute noise: In all of these experiments we varied ω. In the top row, we provide the results of varying ω from 0 (no attribute noise) to 1 (random attribute values) at $\rho = 1$ (PAM). In the bottom row, we provide the results of varying ω from 0 to 1 at $\rho = 0$ (PDM).

as we increase ρ, we introduce more and more correlations across links, thus allowing all three inference algorithms to improve performance. At $\alpha = 0.3$ (the second plot in the top row in Figure 2), a similar trend is also observed. It may not be clear from the plots, but for ICA and LBP, performance at $\alpha = 0.3$ is better than $\alpha = 0.1$. This is expected since more links should provide more correlations to exploit. For example, at $\alpha = 0.1$ and $\rho = 1$, ICA returns an average accuracy of 91% and LBP returns 93.2%, whereas the corresponding numbers at $\alpha = 0.3$ are 93.5% and 95.4% respectively.

At $\alpha = 0.1$ and $\alpha = 0.3$, MF returns respectable performance closely matching ICA's results (and sometimes even improving on them) on all settings of ρ except for $\rho = 1$. This is contrary to what one would expect since $\rho = 1$ corresponds to PAM and there are significant correlations across links to exploit. This is even more surprising given the fact that ICA, the "hard" labeling version of MF, does well at this setting of ρ almost matching LBP's results. We attribute this result to MF's tendency of getting stuck at local minima (Weiss, 2001). Our experiments seem to suggest that very high correlations across links may actually increase MF's tendency to getting stuck at local minima. We discuss this issue more closely in Section 5.4.

At $\alpha = 0.5$ (the last plot in the top row of Figure 2), both MF and LBP return erratic and poor results whereas ICA still returns reasonable results. Variational methods like LBP and MF are known to suffer from the problem of short, closed "loops" (Yedidia et al., 2005). Whenever there are a number of tightly clustered nodes that repeatedly exchange messages with each other, LBP and MF tend to provide poor approximations and thus fail to provide good results. At $\alpha = 0.5$, the large number of links provides a number of closed loops thus causing problems for LBP and MF. Note that contrary to what Murphy et al. (1999) suggest, convergence of LBP is *not* a good indication of LBP's accuracy. In almost all cases in Figure 2, LBP converged in a few iterations (≤ 20) but none of its results at $\alpha = 0.5$ (third plot in the top row in Figure 2) are very good in terms of accuracy.

In the bottom row in Figure 2, we vary ρ from 0 (PDM) to 0.5 (nodes linking randomly). The results are similar to the previous case. At $\alpha = 0.1$ and $\alpha = 0.3$, both ICA and LBP improve results as more correlations across links are introduced with results being better at $\alpha = 0.3$ than at $\alpha = 0.1$ due to the larger number of links. MF, just as before, returns poor results at $\rho = 0.0$ and $\alpha = 0.3$ when there are significant PDM correlations across links. At $\alpha = 0.5$, once again, both LBP and MF face problems with the high link density whereas ICA returns good results showing resilience to link density.

5.3. Effects of varying Noise in Attribute Values

In our second set of experiments, we study the effects of varying attribute noise on the different approximate inference algorithms. Recall that ω controls the attribute noise in the generated datasets. More specifically, at $\omega = 0$ the attribute values are selected from class-specific distributions devoid of noise and as we increase ω, we introduce some randomness in the attribute generation procedure.

Once again, we divide the set of experiments into two parts. In the first part, we set $\rho = 1$ thus studying the effect of varying attribute noise in the presence of PAM. In the sec-

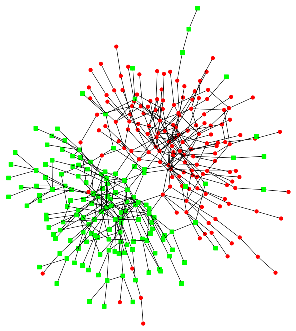

Figure 4. Synthetic data generated by setting $\rho = 1$ (PAM) consisting of 300 nodes and 418 edges. Colors denote labels.

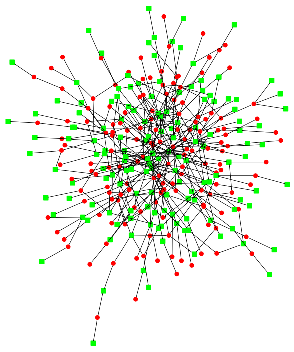

Figure 5. Synthetic data generated by setting $\rho = 0$ (PDM) consisting of 300 nodes and 421 edges. Colors denote labels.

ond part, we set $\rho = 0$ studying the effects of varying ω in the presence of PDM. For each setting we produce three plots, one for each setting of α thus showing the effects of attribute noise at different link densities.

In the first row in Figure 3, we show the results of varying ω from 0 to 1 with ρ set to 1. At $\alpha = 0.1$, when the data is sparse, all three inference algorithms exploit the correlations across links when the attribute values are devoid of noise ($\omega = 0$). As ω increases, more and more noise gets introduced into the attribute values and the performance of all three inference algorithms decrease. Similarly, at $\alpha = 0.3$, due to the increase in the number of links, the inference algorithms perform better than at $\alpha = 0.1$ but only when ω is close to 0. As the plots show, at $\omega \geq 0.8$, all three algorithms perform poorly. This means that if the attribute values are too noisy then no amount of correlations across links is going to help since we do not have any evidence to bootstrap the inference procedure. At $\alpha = 0.5$, the increased link density causes LBP and MF to return poor, erratic results whereas ICA performs well.

In all three plots in the top row of Figure 3, MF tends to perform poorly even at the slightest of noise in the attribute values. In all three plots MF's decreasing performance is the sharpest amongst all three inference procedures.

In the bottom row of Figure 3, we set ρ to 0 (PDM) and varied ω from 0 to 1. In these plots, except for $\alpha = 0.1$, all runs of MF produced very poor results showing its tendency to get caught in local minima. In contrast, ICA and LBP produced good results except when the noise in the attribute values was high ($\omega = 1$). LBP even performed well at $\alpha = 0.5$. ICA produced results close to LBP's but LBP performed slightly better than ICA in most cases.

5.4. PAM vs. PDM: the Effect of different Correlations across Links on Inference Algorithms

In both Figure 2 and Figure 3, LBP seems to perform better in the presence of PDM than in the presence of PAM. For example, compare the two plots in the last column in Figure 2. In the top plot (PAM with noise), LBP shows extremely erratic results while in the right plot (PDM with noise), LBP almost matches ICA's results (even improving slightly at times). There is a similar story in Figure 3. These results seem to suggest that LBP is better at exploiting PDM correlations than exploiting PAM correlations. To find out why, we plotted two of our synthetically generated datasets in Figure 4 and Figure 5.

The two data graphs seem to suggest that, in the case of PAM (Figure 4), the data generator forms clusters where each cluster contains nodes of the same class label and within these clusters we have densely connected loops while in the case of PDM (Figure 5) the links are more spread out. Thus, for LBP, it is easier to infer over datasets with PDM. Note that both data graphs contain roughly the same number of edges. Also, this observation suggests that as we introduce more correlations across links we get more and more closed loops. This may be the reason for the poor performace of MF at $\rho = 1$ and $\alpha = 0.1, 0.3$ in Figure 2.

6. Conclusion

We empirically compared the performance of three popular approximate inference algorithms: loopy belief propagation, mean field relaxation labeling and iterative classification algorithm. ICA is simply the hard labeling version of MF while LBP is a more sophisticated cousin of MF. Our experiments confirm that LBP has a problem dealing with closed "loops" and that MF has severe issues with local minima. ICA tends to perform quite well on random graph data, closely matching LBP's best performances which is surprising. One thing that we observed was, quite often, LBP may converge but to incorrect values. A completely unexpected observation in our experiments was that the type of link correlation (PAM or PDM) may affect the performance of LBP. This will need further corroboration.

References

Berrou, C., Glavieux, A., & Thitimajshima, P. (1993). Near Shannon limit error-correcting coding and decoding: Turbo codes. *Proc. of IEEE Intl. Communications Conf.*.

Besag, J. (1986). On the statistical analysis of dirty pictures. *J. of the Royal Stat. Society*.

Bollobas, B., Borgs, C., Chayes, J., & Riordan, O. (2003). Directed scale-free graphs. *Proc. of the ACM-SIAM Symp. on Disc. Alg.*.

Carvalho, V., & Cohen, W. W. (2005). On the collective classification of email speech acts. *Special Interest Group on Inf. Retrieval*.

Chakrabarti, S., Dom, B., & Indyk, P. (1998). Enhanced hypertext categorization using hyperlinks. *Intl. Conf. on Management of Data*.

Cowell, R. G., Dawid, A. P., Lauritzen, S. L., & Spiegelhalter, D. J. (1999). *Probabilistic networks and expert systems*. Springer, New York.

Getoor, L. (2005). *Link-based classification*. Adv. Methods for Knowledge Discovery from Complex Data, Springer.

Hummel, R., & Zucker, S. (1983). On the foundations of relaxation labeling processes. *IEEE Trans. on Patt. Anal. and Mach. Intel.*.

Jensen, D., & Neville, J. (2002). Linkage and autocorrelation cause feature selection bias in relational learning. *Proc. of the Intl. Conf. on Machine Learning*.

Kschischang, F. R., & Frey, B. J. (1998). Iterative decoding of compound codes by probability progation in graphical models. *IEEE Journal on Selected Areas in Communication*.

Kschischang, F. R., Frey, B. J., & Loeliger, H. A. (2001). Factor graphs and the sum-product algorithm. *IEEE Trans. on Inf. Theory*.

Lafferty, J. D., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proc. of the Intl. Conf. on Machine Learning*.

Lazarsfeld, P. F., & Merton, R. K. (1954). *Friendship as a social process: a substantive and methodological analysis*. Freedom and Control in Modern Society, M. Berger (ed.), New York: Van Nostrand.

Li, S., Wang, H., & Petrou, M. (1994). Relaxation labeling of markov random fields. *Proc. of Intl. Conf. Pattern Recognition*.

Lu, Q., & Getoor, L. (2003). Link based classification. *Proc. of the Intl. Conf. on Machine Learning*.

Macskassy, S., & Provost, F. (2005). NetKit-SRL: A toolkit for network learning and inference. *North American Association for Computational Social and Organizational Science*.

McEliece, R. J., MacKay, D. J. C., & Cheng, J. F. (1998). Turbo decoding as an instance of Pearl's belief propagation algorithm. *IEEE Journal on Selected Areas in Communication*.

McPherson, J. M., Smith-Lovin, L., & Cook, J. M. (2001). Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology*.

Minka, T. (2001). Expectation propagation for approximate bayesian inference. *Proc. of the Annual Conf. on Uncertainty in Artificial Intelligence*.

Murphy, K., Weiss, Y., & Jordan, M. I. (1999). Loopy belief propagation for approximate inference: An empirical study. *Proc. of the Annual Conf. on Uncertainty in Artificial Intelligence*.

Neville, J., & Jensen, D. (2000). Iterative classification in relational data. *AAAI Workshop on Learning Statistical Models from Relational Data, L. Getoor and D. Jensen (ed.)*.

Neville, J., & Jensen, D. (2004). Dependency networks for relational data. *Proc. of the IEEE Intl. Conf. on Data Mining*.

Newman, M. E. J. (2003). Mixing patterns in networks. *Phys. Rev.*

Pearl, J. (1988). Probabilistic reasoning in intelligent systems. *Morgan Kaufmann, San Fansisco*.

Taskar, B., Abbeel, P., & Koller, D. (2002). Discriminative probabilistic models for relational data. *Proc. of the Annual Conf. on Uncertainty in Artificial Intelligence*.

Taskar, B., Guestrin, C., & Koller, D. (2003). Max-margin markov networks. *Neural Inf. Processing Syst.*.

Wainwright, M. J., Jaakkola, T. S., & Willsky, A. S. (2005). Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches. *IEEE Trans. on Inf. Theory*.

Weiss, Y. (2001). Comparing the mean field method and belief propagation for approximate inference in MRFs. *Adv. Mean Field Methods, Saad and Opper (ed), MIT Press*.

Yedidia, J., Freeman, W., & Weiss, Y. (2005). Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. on Inf. Theory*.

Yedidia, J., Freeman, W. T., & Weiss, Y. (2000). Generalized belief propagation. *Neural Inf. Processing Syst.*.

Yuille, A. L. (2002). CCCP algorithms to minimize the bethe and kikuchi free energies: Convergent alternatives to belief propagation. *Neural Inf. Processing Syst.*.