Overcoming Relational Learning Biases to Accurately Predict Preferences in Large Scale Networks

Joseph J. Pfeiffer III¹, Jennifer Neville¹, Paul N. Bennett²
¹Purdue University, ²Microsoft Research
{jpfeiffer,neville}@purdue.edu, paul.n.bennett@microsoft.com

ABSTRACT

Many individuals on social networking sites provide traits about themselves, such as interests or demographics. Social networking sites can use this information to provide better content to match their users’ interests, such as recommending scheduled events or various relevant products. These tasks require accurate probability estimates to determine the correct answer to return. Relational machine learning (RML) is an excellent framework for these problems as it jointly models the user labels given their attributes and the relational structure. Further, semi-supervised learning methods could enable RML methods to exploit the large amount of unlabeled data in networks.

However, existing RML approaches have limitations that prevent their application in large scale domains. First, semi-supervised methods for RML do not fully utilize all the unlabeled instances in the network. Second, the collective inference procedures necessary to jointly infer the missing labels are generally viewed as too expensive to apply in large scale domains. In this work, we address each of these limitations. We analyze the effect of full semi-supervised RML and find that collective inference methods can introduce considerable bias into predictions. We correct this by implementing a maximum entropy constraint on the inference step, forcing the predictions to have the same distribution as the observed labels. Next, we outline a massively scalable variational inference algorithm for large scale relational network domains. We extend this inference algorithm to incorporate the maximum entropy constraint, proving that it only requires a constant amount of overhead while remaining massively parallel. We demonstrate our method’s improvement over a variety of baselines on seven real world datasets, including large scale networks with over five million edges.

Categories: H.2.8 [Database Management]: Database Applications - Data Mining
Keywords: Relational Machine Learning; Maximum Entropy Inference; Large Scale Networks

1. INTRODUCTION

Many individuals on social networking sites provide information about their preferences and behaviors (e.g., profile and activity information). Social networking sites can use this information to better target relevant content to their users. For example, for scheduled events (e.g., concerts), the social networking site can determine whether the event is relevant to a user’s interest and, if so, recommend it to the user. After some initial users have indicated whether they are/are not attending, the service can predict other interested users based on the relational information in the network (i.e., links among users). These types of prediction tasks occur in a variety of scenarios; for example, a site may be interested in estimating the number of users that will like a news article or buy a product, and users may be interested in estimated answers to relational summarization queries, such as “do my friends prefer rock or rap?” Methods that can make accurate predictions about unobserved user preferences are critical for estimation of user-level behaviors.

Common approaches to these types of tasks—where the goal is to accurately predict class probabilities for the set of unlabeled users in the network, given a small subset of labeled users—include conventional machine learning methods for independent and identically distributed (IID) data (e.g., logistic regression), or simple inference-only relational models such as label propagation (LP) [29, 10]. However, these approaches are limited in the sense that they only utilize either intrinsic information (e.g., attributes) or external sources (e.g., neighboring labels), and they do not learn how to combine them together.

Relational machine learning (RML) approaches focus on learning the relative importance of the relational information to the intrinsic information. Traditionally, the parameters for a local conditional model are maximized by solely learning from a labeled subgraph (Figure 1a), where an observed vertex’s label is conditioned on the observed neighbors’ labels and intrinsic attributes. RML then “stitches together” a full joint model by collectively inferring all unlabeled instances. Semi-supervised learning (SSL) RML methods have been developed for partially-observed single network domains where the goal is within-network learning and inference [18, 12]. However, RML SSL methods have limited applicability for sparsely-labeled, large-scale networks due to their restricted performance gains and poor scalability.

More precisely, although IID learning algorithms for SSL usually treat the unlabeled data as weighted probablistic samples, the same cannot be said for the general RML SSL approaches. Methods such as relational expectation maxi-
Figure 1: (a) Pseudolikelihood ($G_L$), (b) Composite likelihood over the full graph $G$, where predicted labels for unlabeled (dashed) vertices are only considered as features of labeled vertices during learning and (dashed) links among unlabeled vertices are only used during collective inference. (c) Pseudolikelihood over the full graph $G$, where all vertices/edges are used for learning.

mization (relational EM, e.g., [18]) partly extend the view of traditional RML learning by utilizing the predictions of the unlabeled examples, incorporating these solely as attributes to the labeled examples (Figure 1b). Note that in this example, the predictions for the dashed vertices are only considered as features during learning, not as labels. This is in contrast to IID SSL methods, which treat inferred predictions as labels that are used to relearn the parameters (see e.g., [13]) (Figure 1c). In this case, learning corresponds to a weighted maximization problem, where the weights of the unlabeled corpus are equal to the previously inferred probabilities. Prior work on relational EM has generally not used this information, largely due to reductions in performance (although special exceptions exist, e.g., [12]).

In addition, although various approximations for learning RML models are generally the same order complexity as IID models, inference with RML models is generally viewed as a limiting factor for large scale implementations. RML methods require the use of collective inference to jointly infer label probabilities throughout large networks, and prior work has rarely been applied on networks with more than 10s of thousands of examples (see e.g., [14, 12]). This is in contrast to both IID learning and simple LF models that have previously been applied to large scale data and employ simple parallel inference algorithms.

In this work, we extend relational SSL approaches to address these two issues and apply them on sparsely-labeled, large-scale networks. In the first component of this work, we discuss how previous relational EM approaches can lead to extreme biases during inference—in particular, relearning from the inferred predictions in relational domains generally collapses all predictions to a singular prediction (e.g., all predict negatively labeled with high probability). To solve this, we augment the inference step of the EM algorithm to include a maximum entropy constraint. Our method (MaxEntInf) adjusts the label predictions at every step of collective inference so they adhere to maximum entropy constraints; namely, we force the predicted label proportions (i.e., percentage predicted positive vs. negative) to align with the proportions observed in the training set. Note the difference from typical maximum entropy approaches that augment the learning step of the algorithm; in this work, we adjust the inference step. Moreover, our approach provides a more general correction than [12], which requires a special model form and regularizer. MaxEntInf can be easily combined with any chosen RML conditional distribution, to keep predictions from collapsing to a singular value, and thus enables the use of more general SSL techniques.

In the second component of our work, we demonstrate that the collective inference step used by RML algorithms can easily be massively parallelized. In particular, we show that through asynchronous updates to variational inference methods, we can achieve linear speedup in terms of the number of cores. This speedup is largely attained due to the avoidance of synchronous updates to high degree vertices (common in real world networks in spite of their sparsity), which would effectively stop the parallelism. Our MaxEntInf algorithm is simple to extend to this massively parallel case and we prove that it requires only a constant overhead for both sequential and parallel algorithms (since it requires a bounded constant number of data samples to compute the correction).

The contributions of this work include the following:

- Identification of a bias with RML SSL learning algorithms in sparsely labeled networks.
- Development of maximum entropy inference (MaxEntInf) to correct the bias.
- Parallelized inference for RML and MaxEntInf.
- Proven error bounds for inference approximations, including both sequential and parallel cases. In particular, MaxEntInf correction can be employed with constant overhead.

We demonstrate the accuracy and scale of our correction and parallel algorithm on seven real world datasets. In particular, we find that our SSL methods using MaxEntInf outperform a variety of competing state-of-the-art baselines, both independent learners and simple relational-only models. We can use the accurate probabilities predicted by MaxEntInf for a variety of problems, including the relational summarization task described above. Notably, we apply our methods networks with over five million edges, demonstrating it scales to networks orders of magnitude larger than prior implementations.

2. RELATIONAL MACHINE LEARNING

Define a graph $G = (V, E)$ to represent our social network, with $v_i \in V$ corresponding to the individuals (vertices) and $E \subseteq V \times V$ indicating the friendships (edges). Let $X, Y$ define the sets of attributes and labels. Every $v_i \in V$ has a corresponding set of attributes $x_i \in X$ and a class label $y_i \in Y$.

In a social network, a subset of items $V_L$ are presumed labeled. Our task is to jointly predict the remaining unknown labels ($Y_U$) using the known labels $Y_L$, attributes
and graph \( G \). Within RML, we can express this statement as jointly inferring the unknown labels given the available information \( P(Y_U \mid Y_L, X, G) \). RML utilizes a local conditional model \( C \) with corresponding parameters \( \Theta_C \), to learn and infer labels within the network. A wide array of relational local conditional models \( C \) exist, and we can view most as a relational extension to common independent models (e.g., relational naive Bayes or relational logistic regression). As many instances are unavailalbe, traditional RML learns a function from the labeled subgraph \( G_L \) where all of the neighbors to all of the instances are known.

\[
\hat{\Theta}_C = \arg \max_{\Theta_C} P(Y_L \mid X_L, G_L, \Theta_C)
\]  

(1)

To make predictions, these parameters are then applied to jointly predict the unlabeled instances on the entire network (not just \( G_L \)):

\[
P(Y_U \mid Y_L, X, \hat{\Theta}_C)
\]

To reduce clutter, in the remainder of this paper we drop references to \( X \): it is fixed and always conditioned on, regardless of the learning representation. RML utilizes conditional independence to simplify the above expressions. Let \( Y_{MB(v)} \) indicate the labels of the Markov blanket of a vertex \( v \), or the friends of an individual \( v \), on a graph \( G \). The full joint distributions are broken into local conditional distributions of the form:

\[
P(y_i \mid Y_{MB(v_i)}, \Theta_C)
\]

By maximizing the pseudolikelihood of the labeled subgraph \( G_L \) we can learn the local model parameters \( \Theta_C \) in a scalable fashion:

\[
\hat{\Theta}_C \approx \arg \max_{\Theta_C} \sum_{v_i \in V_L} \log P(y_i \mid Y_{MBL(v_i)}, G_L, \Theta_C)
\]

(2)

Unlike independent models, we must perform collective classification for inference, due to the dependencies between the unlabeled instances in the network. To this end, the local conditional probabilities are combined with a global conditional model with corresponding parameters \( \Theta \), to estimate the joint distribution over the unlabeled vertices, i.e:

\[
P(Y_U \mid Y_L, G) \approx Q(Y_U) = \prod_{v_i \in V_U} Q_i(y)
\]

where each component \( Q_i(y) \) is iteratively updated in a coordinate ascent algorithm:

\[
Q_i(y) \propto \exp \left\{ \mathbb{E}_{Y_U, \ast} f(y \mid Y_{MB(v_i)}, G, \Theta_C) \right\}
\]

where \( f(\cdot) \) is the unnormalized energy function. Throughout this work, let \( Y_U \) indicate the current joint probability estimates of the unlabeled instances that are iteratively updated via variational inference.

### 2.1 Semi-Supervised Learning

Many relational learning tasks involve learning and inference within a single large, partially-labeled network. In this case, a natural extension from the above RML methods utilizes the unlabeled data to make better predictions within the network. The most common form utilizes expectation maximization (EM; Algorithm 1). EM iteratively updates the parameter estimates by utilizing the expected values of the unlabeled examples to relearn the parameters and can be divided into two basic steps.

**Algorithm 1 Relational-EM(\( Y_L, X, G_L, C \))**

1. \( \hat{\Theta}_C = \text{InitialParameters}(Y_L, X, G_L) \)
2. **while** More Iterations or Not Converged **do**
3. \( P(Y_U) = \text{UpdateInferences}(Y_L, X, G, \hat{\Theta}_C^{-1}) \)
4. \( \hat{\Theta}_C = \text{UpdateParameters}(Y_L, P(Y_U), X, G_L) \)
5. **end while**
6. \( P(Y_U) = \text{FinalizeInferences}(Y_L, X, G, \hat{\Theta}_C) \)

**E-Step:** evaluate \( P(Y_U \mid Y_L, G, \hat{\Theta}_C^{-1}) \)

\[
\arg \max_{\Theta_C} \sum_{Y_U \in \mathcal{Y}_U} P(Y_U \mid Y_L, G, \Theta_C^{-1}) \log P(Y_U, Y_L \mid G, \Theta_C)
\]

(3)

**M-Step:** learn \( \Theta_C \)

\[
\arg \max_{\Theta_C} \sum_{Y_U \in \mathcal{Y}_U} P(Y_U \mid Y_L, G, \Theta_C^{-1}) \log P(Y_U, Y_L \mid G, \Theta_C)
\]

We can compute the E-Step (Eq. 3) via RML collective classification methods; but, as with other RML optimizations, the M-Step (Eq. 4) is intractable to compute directly. Existing relational EM methods [18] simplify the expression by optimizing the composite likelihood (Eq. 6) instead:

**E-Step:** evaluate \( P(Y_U \mid Y_L, G, \hat{\Theta}_C^{-1}) \)

\[
\arg \max_{\Theta_C} \sum_{Y_U \in \mathcal{Y}_U} \sum_{v_i \in V_L} P(y_i \mid Y_{MB(v_i)}, \Theta_C) \log P(y_i \mid Y_{MB(v_i)}, \Theta_C)
\]

(5)

**M-Step:** learn \( \Theta_C \)

\[
\arg \max_{\Theta_C} \sum_{Y_U \in \mathcal{Y}_U} \sum_{v_i \in V_L} \log P(y_i \mid Y_{MB(v_i)}, \Theta_C)
\]

(6)

By utilizing the additional relational information provided by the unlabeled instances, relational EM methods generally outperform traditional RML.

### 3. Likelihood Approximations

As discussed in the previous section, computing the full joint likelihood (Eq. 1) is not scalable to large datasets. Hence, RML maximizes the pseudolikelihood over the labeled graph \( G_L \) (Eq. 2), while relational EM methods maximize the composite likelihood on the graph \( G \) (Eq. 6). Note that although the composite likelihood over the graph \( G \) is similar to the pseudolikelihood, it is distinct as it only sums over the log conditional distributions of the labeled instances.

We graphically illustrate the differences between the learning approaches in Figure 1. Figure 1a shows the traditional RML maximization problem, which uses the full pseudolikelihood of the labeled graph \( G_L \). Relational EM methods use additional information from the predicted label values. In particular, the neighboring probabilities are incorporated as attributes for the label maximization step. This is shown in Figure 1b, where the solid outlined probabilities are treated as labeled instances for maximization and the dashed instances are only used as attributes. Since this is only a partial pseudolikelihood, we refer to these learning algorithms as composite likelihood EM (CL-EM) methods.

Figure 1c shows the full pseudolikelihood maximization, where every solid instance (including the unlabeled items) is used to update the parameter values. This corresponds to the critical component of EM that has been shown to work well for independent data/IID learners—incorporating probabilistic samples of the unlabeled instances into the training set allows the learners to observe new correlations between the various attributes and labels that were not present in the original labeled set. We formalize the generalized relational pseudolikelihood EM (PL-EM) approach as:
4. MAXIMUM ENTROPY INFERENCE

In this section, we introduce our method to correct for the biases experienced by relational classifiers during their inference step; more generally, this will allow us to improve relational EM by utilizing the full pseudolikelihood over all unlabeled data during an EM process.

Note that given a labeled sample $V_L \subseteq V$, it is simple to compute the proportion of observed label types $P(y)$ (e.g., $P(\cdot, +)$). We aim to satisfy the following maximum entropy constraint:

**Proposition 1.** The proportion of unlabeled items with predicted value $y$ should equal the proportion of labeled items with value $y$.

This forms the basis of our maximum entropy inference (MaxEntInf) approach, which will we will use to augment standard collective inference algorithms. We use variational mean field (VMF) inference as our example inference procedure, but the results are more general.

Recall that VMF assumes an approximating distribution $Q(Y_U)$, such that $P(Y_U | Y_L, G, \Theta_C) \approx Q(Y_U)$. For a possible $Y_U \in Y_U$, VMF assumes a fully factorized form $Q(Y_U) = \prod_{i \in Y_U} Q_i(y_i)$, where (in a slight abuse of notation) we have a factorized probability of vertex $v_i$ having label $y$. After each round of VMF, every $Q_i(y_i) \in Q(Y_U)$ corresponds to the probability of an instance having a particular label. For example, for every unlabeled instance with $Q_i(+) \geq 0.5$ the corresponding predicted label $\hat{y}_i$ is +. MaxEntInf forces the proportion of unlabeled instances with $Q_i(+) \geq 0.5$ to be exactly $P(+)$. In more formal terms, MaxEntInf constrains the first moment of the predicted population to match the first moment of the observed population.

Our method will focus on a linear shift around an offset, which (without considerable checks) could result in probabilities lying outside $[0, 1]$—thus directly working with the probabilities themselves is problematic. To this end, we perform a transform of the current probabilities (i.e., every $Q_i(\cdot)$) using the logit function (i.e., $\sigma^{-1}(x) = \log(x/(1-x))$):

$$z_i = \sigma^{-1}(Q_i(+)) = \log\left(\frac{Q_i(+)}{1 - Q_i(+)}\right)$$

The values $z_i \in Z$ take values in the range $[-\infty, +\infty]$, meaning that a linear transform within this space can then be transformed back into probability space through the logistic function (i.e., $\sigma(x) = (1 + e^{-x})^{-1}$).

Next let $z(r)$ indicate the $r^{th}$ ranked value of $Z$ (i.e., index $r$ after $Z$ are sorted). Let $\phi$ be our linear pivot—the index that will split the sorted range into two proportions, one approximately equal to $P(\cdot)$ and the other approximately equal to $P(\cdot)$ (i.e., $P(\cdot) \cdot |V_U|$). Formally, we have:

$$\phi = \arg \min_r \left( \frac{\sum_{i \leq r} |U| \cdot [u \leq r] - P(-)}{|V_U|^2} \right)^2$$

Lastly, for all $z_i \in Z$ we subtract off the corresponding pivot value $z(\phi)$. The result is then transformed back to the probability space to define $Q_i(y)$:

$$Q_i(+) = \sigma(z_i(y) - z(\phi)) \quad Q_i(-) = 1 - Q_i(+)$$

In particular, note that our transformation of $z(\phi)$ is assigned $Q_i(\phi)(y) = \sigma(z(\phi) - z(\phi)) = \sigma(0) = .5$, splitting the data into the two desired proportions to enforce the maximum entropy constraint. Further, the transformation is...
Algorithm 2 MaxEntInf\((G, V_S, Y_U, C)\)

1: \(P(-) = \text{NegativeProportion}(Y_L)\)
2: \(\text{while} \text{ Not Converged or More Iterations do} \)
3: \(Z = []\)
4: \(\text{for every } v_i \in V_S \text{ do} \)
5: \(\text{update variational } Q_i(y)\)
6: \(Z.\text{insert}(\logit(Q_i(+)))\)
7: \(\text{end for} \)
8: \(Z.\text{sort()}\)
9: \(\phi = P(-) \cdot |Z|\)
10: \(\text{for every } v_i \in V_S \text{ do} \)
11: \(Q_i(+) = \text{logistic}(\logit(Q_i(+)) - Z[\phi])\)
12: \(Q_i(-) = 1 - Q_i(+)\)
13: \(Y_U.\text{update}(Q_i(y))\)
14: \(\text{end for}\)
15: \(\text{end while}\)

Algorithm 3 Parallel-MaxEntInf\((G, Y_L, V_U, T, C)\)

1: \([V_U^1, \ldots, V_U^T] = \text{RandomSplit}(V_U, T)\)
2: \(Y_U = \text{SharedMemManager}(Y_L)\)
3: \(\text{for } t \in 1, \ldots, T \text{ do} \)
4: \(\text{spawn } R_t := \text{MaxEntInf}(G, Y_L, V_U, Y_U, C)\)
5: \(\text{end for} \)
6: \(\text{for } t \in 1, \ldots, T \text{ do} \)
7: \(\text{join thread } R_t \text{ upon completion}\)
8: \(\text{end for}\)
9: \(\text{return } Y_U\)

For example, if we set \(\epsilon = .05\) to be the amount of error in the index and \(\delta = .05\) as the probability bound, then to ensure \(95\%\) probability of success \((1 - \delta)\), there only needs to be 28 samples in \(V_S\). Thus, as \(|V_U|\) grows \(|V_S|\) remains fixed, meaning our correction has a constant overhead.

Corollary 1 (Sequential Constant Overhead). For a specified \(\epsilon\) and \(\delta\), an approximation to the proposed sequential algorithm can be performed with constant overhead.

Proof. From Lemma 1, we need only sample \(|V_S| = O(1)\) vertices from \(V_U\) to estimate the offset index \(\phi\). The sampling can be performed in constant time, and sorting and selection is therefore also in constant time. Although updating the probabilities requires \(O(|V_U|)\), the original variational inference algorithm required \(O(|V_U| + |E|)\) time, thus our approach does not increase the order complexity.

The fact that the approximation only requires a constant overhead makes it quite powerful for the sequential algorithm and big data problems in general. In the next section, we discuss parallelizing the method and prove that we retain a constant overhead in this scenario as well.

5. INFERENCE ON LARGE SCALE DATA

In this section, we discuss our scalable approach to VMF inference in parallel. This inference approach will allow us to apply relational machine learning at a scale not previously accomplished, and at the same time handle the MaxEntInf correction necessary for PL-EM.

To start, assume we have a set of \(T\) cores, with shared memory (or memory manager) syncing \(Y_U\) between the cores. We propose solving the joint inference process through an asynchronous and lock-free parallel VMF algorithm [3]. In particular, the unlabeled data is split into \(T\) segments, which are distributed amongst the \(T\) clients. Each client receives its corresponding portion of unlabeled data \(V_U^t\) and is tasked with updating the corresponding \(Q(Y_U^t) \subseteq Q(Y_U)\). Along the way, the client estimates its own \(\phi^t\), computing its own offsets in the logit space, and calibrates its own MaxEntInf correction to the portion of the unlabeled data it is assigned.

Each client uses the memory manager \(Y_U\), which is updated periodically with new label estimates as provided by the other clients. After a client finishes updating its corresponding segment of data, it pushes the newly estimated \(Q_i(y)\) to the memory manager (maintaining \(Y_U\)) for distribution amongst the other clients. When updating \(Q_i(y)\) on a particular client, it is assumed that for every neighboring \(v_j\) some form of \(Q_i(y)\) exists in \(Y_U\), although it may not be the most recent update. As high degree vertices are neighbors to most of the graph, synchronized updates would

Approximating MaxEntInf with Constant Sample Sizes

We can improve the runtime of the above sequential algorithm by sampling from the vector of logit values. In particular, we can prove that with a high confidence \((1 - \delta)\), the chosen offset has provably small error \((\epsilon)\). Importantly, the size of sample does not depend on the data size; rather, it only depends on the amount of error and confidence we wish to have. Define \(V_S \subseteq V_U\), \(\phi^s = P(-) \cdot |V_S|\) and \(Z^s = \{z_i | z_i \in Z \cup v_i \in V_S\}\). Then, we desire the following:

\[
P\left(z_i^*(\phi^s) \in z_i(\phi^s) \right) \geq 1 - \delta
\]

That is, of the full distribution \(V_U\), the \(z_i^*(\phi^s)\) we choose in the subsample \(V_S\) is no more than \(\epsilon\) away from the \(z_i(\phi^s)\) in the full data. This error can be bounded using the Lemma 7 of Manku et al. [11]:

Lemma 1 (Lemma 7 of [11]). Let \(V_S \subseteq V_U\) be a uniformly random subset from the unlabeled vertices, \(\phi\) be the index of our offset, \(\epsilon\) be the amount of error in the chosen index we will allow, and \(\delta\) be the probability bound. To satisfy \(z_i^*(\phi^s) \in z_i(\phi^s)\) with \(1 - \delta\) probability, we must have:

\[
|V_S| \geq \sqrt{\frac{1}{2\epsilon^2} \log \left(\frac{2}{\delta}\right)}
\]

Thus, we require a constant number of samples from \(V_U\).
Corollary 2 (Parallel Constant Overhead). Let $V_{1}, \ldots, V_{t} \subseteq V_U$ be disjoint uniformly random subsets of the unlabeled network vertices. For a specified $\epsilon$ and $\delta$, an approximation to the parallel algorithm proposed can be performed with constant overhead.

Proof. From Theorem 1, we need only sample $|V_S| = O(1)$ vertices from $V_U$ to estimate the offset index $\phi'$. Again, the sampling can be performed in constant time, sorting and selection is therefore also in constant time, meaning that updating the estimates is again done in $O(|V_U| + |E|)$ time.

6. EXPERIMENTS

In this section, we compare our proposed PL-EM framework against a variety of competing state-of-the-art methods. We test each method on seven real world datasets, three of which are an order of magnitude larger than any known prior application of RML.

6.1 Models

To control for variation due to knowledge representation, we compared models based on logistic regression, including independent logistic regression and relational methods that use logistic regression for the local conditional distribution in collective classification. For the relational approaches, three additional variables are incorporated into the conditional distribution: the proportion of positive neighbors, the proportion of negative neighbors, and the degree of the vertex. The parameter learning is done via iteratively reweighted least squares [4], where the least squares solution is solved using the full/skinny streaming QR matrix factorization [3].

Logistic Regression [LR]: This is the independent logistic regression model. It does not consider any relational features, using only the vertex features to predict the label.

Logistic Regression EM [LR (EM)]: The independent logistic regression approach coupled with EM.

Relational Logistic Regression [RLR]: Logistic regression that incorporates relational features (positive proportions, negative proportions and degree). This method
### Dataset Comparison

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$N_v$</th>
<th>$N_e$</th>
<th>$W$</th>
<th>$\rho$</th>
<th>$P(\text{+})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facebook</td>
<td>5,906</td>
<td>73,374</td>
<td>2</td>
<td>0.174</td>
<td>0.320</td>
</tr>
<tr>
<td>IMDB</td>
<td>7,934</td>
<td>122,230</td>
<td>28</td>
<td>0.207</td>
<td>0.164</td>
</tr>
<tr>
<td>DVD</td>
<td>16,118</td>
<td>75,596</td>
<td>28</td>
<td>0.208</td>
<td>0.210</td>
</tr>
<tr>
<td>Music</td>
<td>56,891</td>
<td>272,544</td>
<td>26</td>
<td>0.153</td>
<td>0.078</td>
</tr>
<tr>
<td>Comm.</td>
<td>881,187</td>
<td>5,302,712</td>
<td>50</td>
<td>0.710</td>
<td>0.599</td>
</tr>
<tr>
<td>Computers</td>
<td>881,187</td>
<td>5,302,712</td>
<td>50</td>
<td>0.815</td>
<td>0.169</td>
</tr>
<tr>
<td>Organic</td>
<td>881,187</td>
<td>5,302,712</td>
<td>50</td>
<td>0.486</td>
<td>0.021</td>
</tr>
</tbody>
</table>

Figure 3: Datasets compared. From left: dataset name, number of vertices, number of edges, number of attributes, label correlation, proportion positive.

This does not perform EM and only the initial parameters are used for prediction. Predictions are not made collectively.

**Label Propagation (LP):** This is a standard algorithm for inference in relational networks \[\{2\}, \{10\}\]. It does not learn a dependence on attributes and relational information; rather, the algorithm assumes high correlation and iteratively predicts label probabilities by averaging the current estimates of the relational neighbors. This iterative process repeats until convergence.

**Composite Likelihood EM (CL-EM):** This is the traditional semi-supervised relational EM algorithm that maximizes the composite likelihood \[\{18\}\]. To allow for a comparison, we utilize our parallelized collective inference algorithm for efficiency. However, this method does not utilize the MaxEntInf correction proposed. It performs 10 rounds of variational inference for collective inference. As CL-EM is known to be unstable \[\{14\}\], we smooth the parameters at each iteration. More specifically, we estimate $\Theta_{t+1} = \alpha_t \Theta_{t+1}^{CE} + (1 - \alpha_t) \Theta_{t+1}^{-1}$, where $\alpha_t = \exp\{-0.125 \cdot t\}$. Further, for this method we report the average error between 10 and 11 rounds of EM.

**Naive Pseudolikelihood EM (PL-EM (Naive)):** This method naively applies a semi-supervised relational EM that maximizes the pseudolikelihood rather than the composite likelihood. We again implement our parallelized collective inference algorithm for efficiency, but again omit the proposed MaxEntInf correction. It performs 10 rounds of variational inference for collective inference and, since the PL-EM is more stable than CL-EM, 10 rounds of EM.

**MaxEntInf Pseudolikelihood EM (PL-EM (MaxEntInf)):** This is our proposed semi-supervised relational EM method that uses pseudolikelihood combined with the MaxEntInf approach to correct for relational biases. As with PL-EM (Naive), this method utilizes 10 rounds of variational inference for collective inference, 10 rounds of EM, and maximizes the full PL. However, this approach utilizes our proposed inference correction during each round of variational inference.

### 6.2 Datasets

We compare each of the methods on seven real-world networks, gathered from various sources. Each network only includes vertices with degree greater than zero, and excludes the rest. A full listing of the statistics are given in Figure 3.

#### Smaller Datasets

The first four datasets are small in comparison to the last three. However, each provides a different type of network on which to compare the algorithms; further, they provide a means to evaluate scalability of our parallel inference.

**Facebook:** This is a snapshot of the Purdue University Facebook network. We include users who have listed their (a) political views, (b) religious views and (c) gender. The resulting network contains 5,906 vertices and 73,394 edges. We predict the political views, with the other two variables as features, resulting in a label correlation of 0.174 and positive proportion 0.32. This positive proportion is the largest observed in any dataset.

**IMDB:** This is a movie dataset release by the Internet Movie Database (www.imdb.com). The task is to predict whether a movie will have a gross revenue of $50 million (or greater). As features, we utilize the 19 provided movie genres: for each genre we define an indicator variable for whether the movie falls into the associated genre (these are not necessarily disjoint). In addition, we incorporate the user rating of the movie through 9 boolean indicator variables: each variable indicates whether the average movie rating is greater than the corresponding index. We connect movies through their producers: two movies that share two (or more) producers are linked. The resulting network has 7,934 vertices and 122,230 edges, with label correlation 0.207 and positive proportion 0.164.

**DVD:** This is a subset of the Amazon dataset gathered by \[\{8\}\], with items in the DVD classification. The prediction task is to determine whether an item has an Amazon salesrank < 7500. The attributes are the associated 24 genres that Amazon provides, as well as four boolean variables indicating whether the average number of stars is greater than the associated index. The edges are created through DVD copurchases, with an edge indicating that Amazon believes two items are frequently purchased together. The resulting network has 16,118 vertices and 75,596 edges, with a label correlation of 0.208 and positive proportion 0.21.

**Music:** This is another subset of the Amazon dataset gathered by \[\{8\}\], with items in the Music classification. As before, the prediction task is to determine whether an item has an Amazon salesrank < 7500. The attributes are the associated 22 styles of music that Amazon provides, as well as four boolean variables indicating whether the average number of stars is greater than the associated index. The resulting network has 56,891 vertices and 272,544 edges, with a label correlation of 0.153 and positive proportion 0.078. This is the order of the largest datasets on which RML methods have previously been applied.

#### Larger Datasets

Our large scale network datasets are constructed from the publicly available NBER patents datasets (network structure \[\{9\}\], labelings \[\{6\}\]). For every patent that was published from 1990-2000, we queried the corresponding text from http://patft.uspto.gov stripping out the claims and description for each patent. We removed English stop words \[\{1\}\] and took the top 50 most frequently occurring words. We weighted each document’s words using TF-IDF \[\{7\}\], and each document feature vector was length normalized. The network has 881,187 vertices (patents) and 5,302,712 edges (citations between patents). We constructed three different classification tasks by considering the filing categories associated with each patent \[\{6\}\].

\[1\]http://www.nber.org/patents/subcategories.txt
Figure 4: Results on the four smaller datasets. PL-EM with MaxEntInf outperforms each method.

Figure 5: IMDB: Distribution of predictions at varying labeled percentages. Note that at a high labeled percentage RLR, CL-EM, and Naive PL-EM continue to be poorly calibrated.

Communications: In this task, we considered whether patents were filed in “Primary Category 2, Subcategory 21” or not. The patents in this category are communications patents, involving computer communication infrastructure and technologies. Since this is a subcategory, the label has has considerable skew, with 0.059 positive proportion. However, label correlation is relatively high at 0.71.

Computers: In this task, we considered whether patents were filed in “Primary Category 2” or not. The patents in this category are related in some way to computers. Since it is a relatively large category, the positive proportion in 0.169. It has extremely high label correlation of 0.815.

Organic: In this task, we considered whether patents were filed in “Primary Category 1, Subcategory 14” or not. This category comprises chemical patents that relate to organic compounds. This is the most skewed dataset, with a positive proportion of 0.021. Label correlation is 0.486.

6.3 Methodology

For each dataset, we compare all methods. We repeat the experiments 100 times for the smaller datasets and 20 times for the larger datasets. Our error statistic is the Balanced Absolute Error (BAE) and we report the mean of the trials. The BAE measures the absolute error of a classifier $C$, but normalizes the error across the classes:

$$err_C(y) = \frac{\sum_{v_i \in V_U} P_C(y_i \neq y) \mathbb{I}[y_i = y]}{\sum_{v_i \in V_U} \mathbb{I}[y_i = y]}$$

$$BAE_C = \frac{\sum_{y \in \mathcal{Y}} err_C(y)}{|\mathcal{Y}|}$$

This measure averages the balanced accuracy for all unlabeled instances. For the smaller datasets, we examine the BAE across a range of labeling percentages (0.05-0.9), while on the larger datasets we report accuracies on the more interesting sparser labeling percentages (0.001-0.1). Note that the extremely sparse labelings have only 880 instances out of nearly 900,000 labeled. All tests are paired across the various methods (i.e., each is given the same set of labeled instances).

Our tests were performed on a MacPro with two 2.66GHz 6-Core Intel Xeon processors, capable of 24 possible hyper-threads, with 48GB of RAM. The parallelized algorithms utilized all possible hyper threads, except for during the speedup tests.

6.4 Results

In Figure 4 we report the performance of the varying methods as the percentage of labeled data increases for each of the small datasets. In every instance, PL-EM with MaxEntInf outperforms all of the competing methods. Further, we find that vertex features alone are learned fairly accurately from a low label percentage, with little improvement as more data is gathered. This results in LR (EM) performing on par with LR, and each of these methods are outperformed by the relational methods as the proportion of labeled data increases. By incorporating both relational information and vertex information, RLR makes an initial gain over LP by utilizing the vertex information, then continues to improve at the same rate as LP. These gains are accentuated in the Amazon datasets, where the additional degree information leads to considerable gain over LR and LP. This is due to the salesrank of an item being heavily correlated to the degree ($\approx -0.26$) making the degree highly predictive. For each of these smaller datasets, PL-EM with MaxEntInf improves over the baselines.

We contrast the difference between the Naive PL-EM and PL-EM with MaxEntInf. In particular, the Naive application of PL-EM is nearly always outperformed by the more restrictive CL-EM, particularly for sparsely labeled domains. PL-EM (MaxEntInf) also slightly outperforms PL-EM (Naive) even at higher label percentages. This is due to Naive PL-EM continuing not to calibrate at the higher label percentages. We illustrate this in Figure 5. At the lower label percentage, Naive PL-EM strays far from the prior, as expected, while PL-EM (MaxEntInf) goes through the correct point. For the higher label percentage, PL-EM (Naive) has improved its estimates, but remains further from the
inference, within 20 seconds. This shows that the collective inference necessary for relational machine learning is no longer a significant burden.

6.5 Relational Summarization

As stated in the introduction, having more accurate probability estimates allows us to improve on a variety of tasks outside of simple prediction. So, we additionally compare on the alternative problem of relational summarization. For this we measure the \textit{Friend BAE}, a measure that balances the error of individuals’ friends for making predictions:

$$ferr_c(y,v_i) = \sum_{v_j \in (MB(v_i) \cap V)} Pc(y_j \neq y|y = y)$$

$$\text{FriendBAE}_c(y) = \frac{\sum_{v_i \in V} \sum_{y \in Y} ferr_c(y)}{\sum_{v_i \in V} \sum_{y \in V} Pc(y)}$$

These results are reported in Figure 8 on the DVD and Music datasets from Amazon, as well as the Communication and Computers large scale datasets. As with the original BAE, PL-EM with MaxEntInf outperforms all competing methods. Interestingly, PL-EM performs considerably better in the space on the DVD and Music datasets. This implies that Amazon generally has high degree instances connect with other high degree instances, making the prediction task easier due to the degree variable. In contrast, such effect does not occur in the Patents dataset, implying that the error is largely dispersed independent of degree. Overall, the PL-EM with MaxEntInf is able to outperform other estimators on a variety of tasks.

7. RELATED WORK AND DISCUSSION

Our work advances the field of relational machine learning (RML) in two notable directions. First, we demonstrated that the error from the pseudolikelihood maximization learning approximation can be overcome by correcting the inference step of the algorithm. This approach allows any relational conditional distribution to be corrected on the fly solely by a small correction to the inference step, as well as allowing for the more general PL-EM algorithm to be used in conjunction with the chosen conditional. Second, we demonstrated that by using asynchronous variational mean field inference we can trivially parallelize the joint inference
algorithm, allowing for fast computations of the unlabeled probabilities. As part of this, we demonstrated that we can also parallelize the correction and we provided bounds on the error from this parallelization. The correction itself is similar to one proposed by [17] for IID EM learning, although we propose applying it in a considerably different domain and demonstrate it overcoming extreme inference biases. We also provided the error bounds for the sampled approach, proofs for the constant sampling overhead and demonstrated success in parallelizing the subsamples on large domains.

Seminal related work centers around relational Markov networks (RMNs) [16] and their first-order logic generalization Markov logic networks (MLNs) [15]. Both formalisms lie within the exponential family, meaning optimizing their corresponding MLEs will maximize their entropies. However, for large scale and complex network data these are also typically learned by maximizing the pseudolikelihood. The MLE approximation provably over-propagates error for RMNs [19], similar to the extreme prediction biases observed in this work for the logistic regression conditionals. Hence, parameters learned from RMNs or MLNs likely suffer similar problems as the logistic regression formulation. This raises interesting theoretical questions on what characteristics a conditional form must have in order to avoid the over propagation error, which we leave for future work.

The most related work to ours is that of McDowell & Aha [12]. Their work first noted the differences between learning from the labeled vertices and the full network during the ‘M’-step. However, their solution required the use of a special regularizer during the optimization step, and made use of a specific form of conditional. In contrast, our work pairs with any black box conditional distribution that provides label probabilities. Further, we demonstrated the power of parallelizing our inference step, allowing for our correction and inference method to scale to data orders of magnitude above previous relational algorithms.

Lastly, recent work on stochastic variants of the CL-EM algorithm has produced methods for relational stochastic EM and relational data augmentation [13]. These methods aggregate over a range of possible parameter values, resulting in significantly more stable estimates. However, each of these methods requires a maximization after a single round of Gibbs sampling. In [14], the authors proposed 1000 maximizations. In contrast, our MaxEntInf approach requires only 10 maximizations, and our estimation considers the full pseudolikelihood case. Coupled with our parallel algorithm, our approach is both more efficient and more general.

8. CONCLUSIONS

In this work, we proposed a novel maximum entropy constraint for inference during RML. Implementing this constraint is straightforward, allowing it to be used in conjunction with any relational learner. We proved the method has a constant overhead, making it ideally suited for big data problems. Additionally, we applied asynchronous variational mean field algorithms with success to relational inference problems. The maximum entropy inference correction is also ideally suited for our parallel implementation; as with the sequential case, we proved that it can be implemented with constant overhead. We demonstrated our approach on 7 real world network domains, showing that it outperformed a variety of baselines and competing methods. Further, we showed our parallel corrected inference procedure can be performed in under 20 seconds on networks with more than five million edges—an order of magnitude larger than prior works.

The methods presented here open multiple avenues for future work. First, the sampling approximation proofs are not limited to providing estimates on a per core basis: in many real world dataset we may wish to partition our data to reduce traffic overhead across the links while preserving an unbiased corrective estimate. However, these partitions may naturally introduce a bias on each core (e.g., geographic partitions). The included proofs provide evidence that the inference correction needs only a constant overhead between the cores, potentially greatly reducing the latency between the servers. Second, the proposed maximum entropy inference correction provides encouraging evidence that corrective inference methods can significantly improve relational machine learning methods, but is currently limited to the binary classification case. This approach should be expanded to include a larger number of possible label values. Third, these results raise interesting questions for which forms of conditionals are prone to bias and over propagation error effects. More theoretical work is necessary to determine when a relational classifier might suffer from this bias and whether there are classifiers that are robust to these errors.

Acknowledgements

We thank David Gleich for his help with optimizing the maximization problem for the learning algorithms and Iman Alodah for her help with parts of the data processing. This research is supported by NSF under contract number(s) IIS-1149789 and IIS-1302172.
REFERENCES


