Estimating Aggregate Properties In Relational Networks With Unobserved Data

Varun Embar*  
UC Santa Cruz  
vembar@ucsc.edu  

Sriram Srinivasan*  
UC Santa Cruz  
ssriniv9@ucsc.edu  

Lise Getoor  
UC Santa Cruz  
getoor@soe.ucsc.edu  

Abstract
Aggregate network properties such as cluster cohesion and the number of bridge nodes can be used to glean insights about a network's community structure, spread of influence and the resilience of the network to faults. Efficiently computing network properties when the network is fully observed has received significant attention (Wasserman and Faust 1994; Cook and Holder 2006), however the problem of computing aggregate network properties when there is missing data attributes has received little attention. Computing these properties for networks with missing attributes involves performing inference over the network. Statistical relational learning (SRL) and graph neural networks (GNNs) are two classes of machine learning approaches well suited for inferring missing attributes in a graph. In this paper, we study the effectiveness of these approaches in estimating aggregate properties on networks with missing attributes. We compare two SRL approaches and three GNNs. For these approaches we estimate these properties using point estimates such as MAP and mean. For SRL-based approaches that can infer a joint distribution over the missing attributes, we also estimate these properties as an expectation over the distribution. To compute the expectation tractably for probabilistic soft logic, one of the SRL approaches that we study, we introduce a novel sampling framework. In the experimental evaluation, using three benchmark datasets, we show that SRL-based approaches tend to outperform GNN-based approaches both in computing aggregate properties and predictive accuracy. Specifically, we show that estimating the aggregate properties as an expectation over the joint distribution outperforms point estimates.

Introduction
Large relational networks are ubiquitous, arising naturally across several domains such as social media (e.g., friendship and follower networks), computational biology (e.g., protein interaction networks) and IoT (e.g., sensor networks). Structural properties, such as bridge nodes and graph clustering coefficients, are used to analyze the network for tasks such as influence maximization, community structure and spread of information. While many such structural properties have been proposed (Scott 1988; Wasserman and Faust 1994; Cook and Holder 2006; Rajaraman and Ullman 2011), along with efficient algorithms to estimate them (Shi et al. 2015; Liu et al. 2018; Wu et al. 2014; Qu et al. 2014; Dunne and Shneiderman 2013), the task of computing these properties in the presence of missing information, such as node labels, has not received much attention.

In such networks, we need to combine the tasks of estimating these properties with inference of missing information. Here we examine two categories of network inference approaches: statistical relational learning (SRL) and graph neural networks (GNNs). SRL approaches (Getoor and Taskar 2007) are one class of machine learning approaches which are well suited to making inferences in multi-relational networks. Most SRL approaches provide a way of specifying a declarative probabilistic model and efficient inference and learning algorithms (Richardson and Domingos 2006; Bach et al. 2017; De Raedt and Kersting 2008; Friedman et al. 1999; Neville and Jensen 2007). In this paper, we investigate two such approaches, Markov logic networks (MLNs) (Richardson and Domingos 2006) and probabilistic soft logic (PSL) (Bach et al. 2017). Both approaches specify a model using weighted logical rules, and use the model to define a joint distribution over the missing information. We can use the MAP or mean of this distribution to infer missing values, and estimate the properties, or we can compute the expectation of these properties over the joint distribution.

GNNs are another class of machine learning approaches based on neural networks that can infer missing information in relational networks (Gilmer et al. 2017; Hamilton, Ying, and Leskovec 2017; Kipf and Welling 2017; Velickovic et al. 2018; Qu, Bengio, and Tang 2019). These approaches learn non-linear representations of nodes in the network and use these node representations to infer missing information. Graph convolution networks (GCN) (Kipf and Welling 2017) and Graph attention networks (GAT) (Velickovic et al. 2018) are two such popular approaches. Once the node representations are learned, these approaches independently infer
the missing information for each node. Graph Markov neural networks (GMMN) (Qu, Bengio, and Tang 2019) is a recently proposed neural network-based approach that models dependencies in the missing information along with node representations. We can use these models to infer the missing values and estimate the properties. We cannot compute the expectation for these approaches, as the final softmax layer infers a distribution for each node independently, and not the joint distribution over all nodes.

In this work, we study the effectiveness of SRL and GNN based approaches in computing aggregate structural properties of networks with missing information. For the MAP and mean estimates of SRL approaches and for GNN based approaches, we use the point estimates to infer the missing information and then compute the aggregate property. For SRL approaches, we also compute the properties as an expectation over the joint distribution.

Further, we introduce a novel sampling approach for computing expectation in PSL. We propose an efficient Gibbs sampling based approach, ABGibbs, to generate samples from the PSL joint distribution. We use the generated samples to compute the expectation of aggregates tractably using Monte Carlo approximation. ABGibbs identifies highly correlated random variables (RVs), called association blocks, and performs block sampling on these blocks. This overcomes the poor performance of a naive Gibbs sampler due to high correlation between RVs.

The contributions of our paper include: 1) We define several practical aggregate properties of a network; 2) We propose a novel Metropolis-within-Gibbs sampling framework, ABGibbs, to generate samples from the joint distribution of PSL; 3) We analyze the effectiveness of three popular graph neural networks (GCN, GAT, GMMN) and two SRL methods (PSL and MLN) in computing the proposed aggregate properties; 4) Through experiments on three benchmark datasets, we show that computing aggregate properties as an expectation outperforms point estimate, and the runtime experiments show that the proposed ABGibbs approach for PSL is up to 3 times faster than MLN sampling approach;

Background

In this section, we first review the field of Statistical Relational Learning (SRL), including MLNs and PSL, followed by three graph neural network approaches (GNNs), GCN, GAT and GMMN.

Statistical Relational Learning

SRL methods combine probabilistic reasoning with knowledge representations that capture the structure in relational data. SRL frameworks define a joint probability distribution over the set of all possible networks using a declarative probabilistic model. An SRL model $M$ is defined by a set of first order formula $F_i$, associated with weights $w_i$. Intuitively, the weight of a formula indicates how likely it is that the formula is true in the world.

Given the domain for the variables in the formulas, SRL approaches generate a set of ground formulas, where the variables are replaced with the values in the domain. The atoms in the formula, where the variables are replaced with the values, are called ground atoms. These approaches then induce an undirected graphical model over the set of ground atoms, where each ground atom is modeled as a random variable (RV). The cliques in the graphical model correspond to the ground formulas.

Given an assignment to a set of ground atoms $X$, the probability distribution over the remaining unobserved ground atoms $Y$ is given by:

$$ P(Y = y | X = x; w) = \frac{1}{Z} \exp \left( \sum_{i=1}^{m} w_i \phi_i(x, y) \right) $$

where $\phi_i(x, y)$ are the potential functions corresponding to the ground formulas and $Z$ is the normalization constant.

Markov Logic Networks (MLNs): MLNs (Richardson and Domingos 2006) are a notable SRL framework where the ground atoms are modeled as binary RVs. The potential functions are defined using boolean satisfiability, and take the value 1 if the ground formula is satisfied, and 0 otherwise.

Probabilistic Soft Logic: PSL (Bach et al. 2017) is another recently introduced SRL framework. Unlike MLNs, the ground atoms in PSL are continuous and defined over the range $[0, 1]$, and the weights $w_i$ are restricted to $\mathbb{R}^+$. For the potential functions, PSL uses a continuous relaxation of boolean logic which results in hinge functions instead of boolean satisfiability. These differences make MAP inference in PSL convex.

Graph neural networks

Another line of research for inferring missing information in networks is based on the recent progress of graph neural networks. These techniques learn a non-linear representation for each node using neural networks and use these representations to infer missing attributes independently.

Graph Convolution Networks (GCNs): GCNs (Kipf and Welling 2017) iteratively update the representation of each node by combining each node’s representation with its neighbors’ representation. The propagation rule to update the hidden representation of a node is given by:

$$ H^{(l+1)} = \sigma \left( \tilde{D}^{-0.5} \tilde{A} \tilde{D}^{-0.5} H^{(l)} W^{(l)} \right) $$

where $H^{(l)}$ denotes the representation at layer $l$, $\tilde{D}$ represents the degree matrix, $\tilde{A}$ represents the adjacency matrix with self-loop, and $W$ represents the weights. $\sigma$ denotes an activation function, such as the ReLU. The final representations are fed into a linear softmax layer classifier for label prediction.

Graph Attention Networks (GATs): GATs (Veličković et al. 2018) are similar to GCNs, where node representations are updated iteratively by combining the representation of each node with its neighbors. However, instead of using a graph Laplacian, GATs use self-attention while combining representations. This allows the model to assign different weights to each of its neighbors’
probabilistic aggregate estimation

SRL techniques such as PSL and MLNs model missing node attributes as RVs and define a joint probability distribution over them. This makes the aggregate properties a function of RVs and we can compute the expected value of these functions over the distribution. However, computing the expectation analytically may not always be possible due to the intractability of the integration in the expectation.

One way to overcome this problem is through the use of Monte Carlo methods to approximate the expectation by drawing samples from the distribution. The expectation can be approximated as follows:

\[ E_f = \frac{1}{S} \sum_{j=1}^{S} f(G, a_{v_i}^o, a_{v_i}^{u(j)}) \]

where \( a_{v_i}^{u(j)} \) are samples drawn from the distribution \( p(a_{v_i}^u | G, a_{v_i}^o) \).

Gibbs sampling (Gilks, Richardson, and Spiegelhalter 1995) is a type of MCMC sampling approach that generates samples from the joint distribution by iteratively sampling from the conditional distribution of each RV keeping the remaining RVs fixed. For MLNs, approaches such as MC-SAT have been proposed (Poon and Domingos 2006) that combine MCMC and satisfiability, and are shown to greatly outperform Gibbs sampling.

However, using Gibbs sampling for PSL has two main challenges. First, unlike MLNs, where the conditional distributions follow a binomial distribution and is easy to sample, it is non-trivial to generate samples from the conditional distributions of PSL. The conditional distribution for a RV \( y_i \) conditioned on all other variables \( X, Y_{-i} \) in PSL is given by:

\[ p(y_i | X, Y_{-i}) \propto \exp \{ \sum_{r=1}^{N_i} \phi_r (y_i, X, Y_{-i}) \} \]

where \( N_i \) is the number of groundings that variable \( y_i \) participates in. The above distribution neither corresponds to a standard named distribution nor has a form amenable to techniques such as inversion sampling. Second, Gibbs sampling has poor convergence rates when the RVs are highly correlated. Identifying such high probability regions is challenging.

Unlike previously existing hit-and-run based sampling approach (Broecheler and Getoor 2010), we propose a simple and effective Gibbs sampling based approach to handle both these challenges. Our proposed sampling approach, ABGibbs, overcomes the first challenge of sampling from the conditional by incorporating a single step of a Metropolis-Hastings algorithm within the Gibbs sampler (also called Metropolis-within-Gibbs (Gilks, Richardson, and Spiegelhalter 1995)). For each RV \( y_i \), we first sample a new value \( y_i' \) from a symmetric proposal distribution \( q(y_i) \) and compute the acceptance ratio \( \alpha \) given by:

\[ \alpha = \frac{\exp \{ \sum_{r=1}^{N_i} \phi_r (y_i, X, Y_{-i}) \}}{\exp \{ \sum_{r=1}^{N_i} \phi_r (y_i', X, Y_{-i}) \}} \]
We then accept the new value $y_t'$, as a sample from the conditional, with a probability proportional to the acceptance ratio $\alpha$.

Highly weighted PSL rules with multiple unobserved terms result in highly correlated RVs. In general, highly weighted rules with more than two unobserved atoms can result in a large fraction of RVs becoming highly correlated, making it very hard to sample from the distribution. However, in the case of rules with up to two unobserved atoms, we propose a novel strategy to identify and cluster the correlated variables. We refer to the groups that we construct as associated blocks. The primary idea of ABGibbs is to combine association blocks and Metropolis-within-Gibbs technique to generate samples efficiently from the joint distribution of a PSL model. The algorithm for generating samples is shown in Algorithm 1. Our approach first identifies association blocks using the algorithm shown in Algorithm 2. We identify pairwise associations between ground atoms from ground rules with weights greater than a threshold $\lambda_c$. We also keep track of the region where these potential functions are minimized. This corresponds to a region of high probability. We identify ground atom pairs where the region of high probability is below a threshold $\theta$. Finally, we merge these pairs into blocks such that all associated pairs lie in the same block.

Algorithm 1 ABGibbs sampler for PSL

**Input:** Set of $N$ ground rules $R$, # of iterations $T$, burn-in period $b$, weight threshold $\lambda_t$, range threshold $\theta$.

**Output:** Set of samples $\mathcal{S}$

# Identify the associated blocks using the ABlock algorithm
$C \leftarrow \text{ABlocks}(R, \lambda_t, \theta)$

# Initialize $Y^{(0)}$ to MAP state
$Y^{(0)} \leftarrow \text{argmax}_Y p(Y|X)$

for $t$ from 1 to $T$ do

# Sample values for each block $c \in C$
for $c^{(t)} \in C^{(t)}$ do

$c' \sim \text{BlockSample}(c, R^+, R^-)$

$\alpha = \frac{\exp(\sum_{r=1}^{N} w_r \phi_r(c', X, Y^{(t+1)}))}{\exp(\sum_{r=1}^{N} w_r \phi_r(Y^{(t)}), X, Y^{(t+1)}))}$

$u \sim U(0, 1)$

if $u \leq \alpha$ then

$Y_c^{(t+1)} = c'$

else

$Y_c^{(t+1)} = Y_c^{(t)}$

end if

end for

# Consider samples after burn-in period $b$
if $t > b$ then

$\mathcal{S} = \mathcal{S} \cup Y^{(t)}$

end if

end for

Return $\mathcal{S}$

Having identified the association blocks, we generate samples from the distribution using a blocked Metropolis-in-Gibbs sampler. We initialize the RVs to the MAP state.

Algorithm 2 ABlock: Identifying blocks of associated RVs

**Input:** Set of $N$ ground rules $G$, weight threshold $\lambda_t$, range threshold $\theta$.

**Output:** Blocks of associated RVs (RVs) $C$

**Initialize:** Hashmaps of associated RVs (RVs) $C$

for $r \in 1 \to N$ do

# For rules with high weights
if $\lambda_r > \lambda_t$ then

# Update the bounds
if $r$ is of the form $a - b \leq c$ then

$R^-(a, b).\max = \min\{R^-(a, b).\max, c\}$

else if $r$ is of the form $a - b \geq c$ then

$R^-(a, b).\min = \max\{R^-(a, b).\min, c\}$

else if $r$ is of the form $a + b \leq c$ then

$R^+(a, b).\max = \min\{R^+(a, b).\max, c\}$

else if $r$ is of the form $a + b \geq c$ then

$R^+(a, b).\min = \max\{R^+(a, b).\min, c\}$

end if

end if

end for

# Identify clusters from pairwise associations
for $(a, b) \in R^+ \cup R^-$ do

if $R^+(a, b).\max - R^+(a, b).\min \leq \theta$ or $R^-(a, b).\max - R^-(a, b).\min \leq \theta$ then

Merge blocks containing $a, b$ and update $C$

end if

end for

Add remaining RVs as singleton clusters to $C$

Return set of blocks $C$

We then iteratively sample new values for each association block $B_k$ after a burn-in period $b$. The proposed sampling approach for each block is given in Algorithm 3. We first randomly choose a variable $y_t \in B_k$ and sample a value from $U(0, 1)$. We then update the region of high probability for all RVs in $B_k$ based on the sampled value for $y_t$. We randomly choose a variable $y_{t+1} \in B_k$ such that $y_{t+1} \neq y_1, \ldots, y_t$, and sample a value from the region of high probability with probability $\beta$ and sample a value from $Unif(0, 1)$ with probability $1 - \beta$. We again update the region of high probability for all variables in $B_k$ with $y_{t+1}$. This process is performed iteratively for all the variables in the block. Finally, once we have sampled a value for all the variables in a block, we accept or reject the samples for all $y_t \in B_k$ with probability $\alpha$. A single sample from the joint distribution is complete when every block has been sampled once.

Aggregate properties

Next we discuss the several aggregate queries studied in this paper that are useful in analyzing community structure and spread of influence in social networks. We illustrate using a citation network given by $G = (V, E, a_v)$, where $V = \{v_1, \ldots, v_n\}$ correspond to documents, and $E = \{e_{uv} | u, w \in V\}$ corresponds to a citation link from documents $u$ to $w$. For each node $v$, the set of node attributes
Algorithm 3 BlockSample: Sampling variables in a block

Input: A block of RVs c, R⁺, R⁻
Output: Sample s for variables in c
s = Ø
Pick a variable yᵢ from c at random
yᵢ ~ U(0, 1)
s.add(yᵢ)
while yⱼ ∈ c \ s and associated to some variable in s do
  Update range [u, v] for yⱼ based on s, R⁺, and R⁻
  b ∼ [0, 1]
  if b ≤ β then
    yⱼ ∼ U(u, v)
  else
    yⱼ ∼ U(0, 1)
s.add(yⱼ)
end if
end while
Return s

is given by $a_c$. The node attribute corresponding to the document category is denoted by $c_i \in \{0, \ldots, \kappa\}$ where $\kappa$ is the number of categories. On the above network, we define five different queries Q1 to Q5. Q1 and Q2 are inspired from cluster analysis (Tan, Steinbach, and Kumar 2006) and Q3, Q4, and Q5 are related to bridge nodes in social networks (Musial and Juszczyszyn 2009).

[Q1]: Category Cohesion: This property is defined as the count of document pairs $(vᵢ, vⱼ)$ that have a citation link and belong to the same category $cᵢ = cⱼ$, i.e.,

$$Q1 = \sum_{eᵢⱼ \in E} \mathbb{1}(cᵢ = cⱼ)$$

where $\mathbb{1}$ is an indicator function with value one when the condition is satisfied. Network where categories are isolated have a high value.

[Q2]: Category Separation: This property is defined as the count of document pairs $(vᵢ, vⱼ)$ that have a citation link between them and belong to different category $cᵢ \neq cⱼ$, i.e.,

$$Q2 = |E| - Q1$$

Networks that have related categories have a large value for this property.

[Q3]: Diversity of Influence: This property is defined as the number of documents $vᵢ$ that are connected to documents belonging to more than half the categories, i.e.,

$$Q3 = \sum_{i=1}^{n} \mathbb{1}\left(\left|\{cⱼ | \forall j = 1, \ldots, n (eᵢⱼ \in E \land cᵢ \neq cⱼ)\}\right| \geq \frac{\kappa}{2}\right)$$

where $\{|\ldots\}$ indicates number of elements in the set. Networks containing documents that have influenced many categories have a large value for this property.

[Q4]: Exterior Documents: This property is defined by the number of documents $vᵢ$ that have more than half the neighbors belonging to categories other than the documents category $cᵢ$, i.e.,

$$Q4 = \sum_{i=1}^{n} \mathbb{1}\left(\left(\sum_{j=1}^{n} eᵢⱼ \mathbb{1}(cᵢ \neq cⱼ)\right) > \frac{\sum_{j=1}^{n} eᵢⱼ}{2}\right)$$

Unlike Q3, Q4 measures the number of documents that have more reach in a different category than their own. Networks where the categories are not well separated typically have a large value for this property.

[Q5]: Interior Documents: This property is defined as the number of documents $vᵢ$ that have more that half of its neighbors belonging to the same category as the document $cᵢ$, i.e.,

$$Q5 = \sum_{i=1}^{n} \mathbb{1}\left(\left(\sum_{j=1}^{n} eᵢⱼ \mathbb{1}(cᵢ = cⱼ)\right) > \frac{\sum_{j=1}^{n} eᵢⱼ}{2}\right)$$

Networks with large category clusters typically have a large value for this property.

Empirical Evaluation

In this section we analyze the performance of SRL and GNN-based approaches on the queries proposed in the previous section. We also compare the predictive and runtime performance of these approaches.

Experimental Setup and Datasets

We consider three benchmark citation datasets for relational learning: Cora, Pubmed and Citeseer (Sen et al. 2008). The statistics for these datasets are given in Table 1. Each node in the network corresponds to a document, and has attributes describing the words occurring in a document represented as a bag-of-words, and an attribute that represents the category of the document. Links between documents represent citations. We assume all the words and citations are observed, while the categories are only partially observed. We follow the same splits as (Yang, Cohen, and Salakhutdinov 2016) and the number of observed node labels is given in Table 2.

Table 1: Statistics for the three datasets: Cora, Pubmed and Citeseer.

| Dataset   | #Categories | #Nodes | #Edges | #Cats | #Obs Labels |
|-----------|-------------|--------|--------|-------|-------------|
| Cora      | 7           | 2,308  | 5,429  | 1433  | 640         |
| Pubmed    | 3           | 15,917 | 44,139 | 580   | 550         |
| Citeseer  | 6           | 3,327  | 4,732  | 3703  | 620         |
Table 2: Relative errors ($\delta$) for different queries on the three datasets. PSL-SAMPLES and MLN-SAMPLES have lower error. These approaches compute the expectation of the properties over the distribution.

| Methods   | Q1 - $\delta$ | Q2 - $\delta$ | Q3 - $\delta$ | Q4 - $\delta$ | Q5 - $\delta$ | $\delta$ |
|-----------|---------------|---------------|---------------|---------------|---------------|---------|
| PSL-MAP   | 0.047         | 0.205         | 0.165         | 0.1                      | 0.115         | 0.115   |
| MLN-MAP   | 0.032         | 0.046         | 0.412         | 0.436         | 0.242         | 0.234   |
| MLN-MAP   | 0.021         | 0.090         | 0.027         | 0.054         | 0.041         | 0.047   |
| MLN-MAP   | 0.038         | 0.163         | 0.009         | 0.174         | 0.068         | 0.090   |
| GCN       | 0.046         | 0.207         | 0.137         | 0.671         | 0.34           | 0.28    |
| GCN       | 0.073         | 0.318         | 0.356         | 0.097         | 0.355         | 0.362   |
| GCN       | 0.041         | 0.305         | 0.114         | 0.352         | 0.322         | 0.322   |
| PSL-SAMPLES | 0.014      | 0.061         | 0.050         | 0.083         | 0.031         | 0.041   |
| MLN-SAMPLES | 0.045         | 0.161         | 0.034         | 0.173         | 0.068         | 0.097   |
| PSL-SAMPLES | 0.175         | 0.527         | 0.467         | 0.57          | 0.272         | 0.445   |
| MLN-MAP   | 0.207         | 0.648         | 0.594         | 0.794         | 0.392         | 0.527   |
| PSL-MAP   | 0.134         | 0.403         | 0.544         | 0.551         | 0.253         | 0.377   |
| MLN-MAP   | 0.137         | 0.731         | 0.792         | 0.691         | 0.315         | 0.584   |
| GCN       | 0.211         | 0.637         | 0.712         | 0.813         | 0.396         | 0.553   |
| GA T      | 0.248         | 0.747         | 0.9           | 0.881         | 0.416         | 0.639   |
| GMNN      | 0.253         | 0.774         | 0.881         | 0.906         | 0.437         | 0.655   |
| PSL-SAMPLES | 0.133         | 0.412         | 0.539         | 0.499         | 0.256         | 0.364   |
| MLN-SAMPLES | 0.244         | 0.736         | 0.935         | 0.691         | 0.315         | 0.585   |

The different SRL based approaches that we consider are:

**MLN-MAP**: This is the mode of the distribution defined by the MLN model over the unobserved node labels. We use the MaxWalkSAT algorithm implemented in the Tuffy framework [Niu et al. 2011].

**MLN-MEAN**: This is mean of the distribution defined by the MLN model over the unobserved node labels. We generate 1100 samples using the MC-SAT algorithm, discard the first 100 samples as burn-in samples and use the 1000 samples to approximate the mean.

**MLN-SAMPLES**: Here we estimate the properties as an expectation over the distribution defined by the MLN model. We generate samples similar to MLN-MEAN, but randomly choose 100 samples from the 1000 (to ensure minimal correlation) and use Monte Carlo approximation to compute aggregate property expectation.

**PSL-MAP**: This is the mode of the distribution defined by the PSL model over the unobserved node labels. We use the ADMM algorithm implemented in the PSL framework [Bach et al. 2017] to obtain labels.

**PSL-SAMPLES**: Here we estimate the properties as an expectation over the distribution defined by the PSL model. Like MLN-SAMPLE we generate 100 samples and use our proposed Monte Carlo approximation to compute aggregate property expectation.

**GNN based approaches**: The approaches use the node representations to infer node labels. These models use 20 observed node labels from each category to train the model and use the remaining 500 node labels for performing early-stopping. For all three approaches we use the code and hyperparameters provided by the authors of the respective paper. The different GNN based approaches that we consider are:

**GCN**: This approach returns a point estimate computed by the graph convolutional network [Kipf and Welling 2017].

**GAT**: This approach returns a point estimate computed by the graph attention network [Veličković et al. 2018].

**GMNN**: This approach also returns a point estimate computed by the Markov neural network introduced recently [Qu, Bengio, and Tang 2019].

We evaluate the performance on all the queries (Q1 to Q5) using the relative error ($\delta$) as a metric. The relative error $\delta$ is computed using: $\delta = \frac{|P - T|}{T}$ where $T$ is the true value of the query and $P$ is the estimated value. We evaluate the overall performance of a method by computing the mean over all the $\delta$s denoted by $\bar{\delta}$. We also report the predictive performance of these methods, by computing the categorical accuracy (Acc) on all the unobserved nodes. Further, for runtime comparison, we measure the total time taken for each of these approaches.
Among the neural network based approaches we observe that GCN performs better than GAT and GMNN. This is again in contrast to the accuracy, where GCN performs poorly, and GMNN has the higher Acc.

Among the queries, we observe that Q1 and Q5 have lower error compared to the other queries for all the methods. Both Q1 and Q5, estimate node pairs that are adjacent and have the same category. These are easier to estimate as these nodes typically lie at the center of a category clusters. As a result, the node attributes for these nodes and their adjacent nodes are similar. Since all the approaches propagate the similarity between the node neighbors, the models have a lower error on these queries. We computed the accuracy for nodes that participate in these queries and found that most of the methods had an accuracy of over 90%.

Queries Q2, Q3, and Q4 estimate nodes that have neighbors with different categories. These are nodes that lie in the boundary of category clusters and whose categories are harder to infer. We observed a reduction in the accuracy to about 60% for nodes that participate in these queries. Approaches such as GMNN have very large relative error for these queries, resulting in overall poor performance.

### Runtime Comparisons

| Methods     | Cora Time (sec) | Pubmed Time (sec) | Citeseer Time (sec) |
|-------------|-----------------|-------------------|---------------------|
| PSL-MAP     | 14              | 124               | 77                  |
| PSL-MEAN    | 103             | 638               | 124                 |
| MLN-MEAN    | 270             | 1947              | 166                 |
| MLN-MAP     | 65              | 368               | 36                  |
| GCN         | 24              | 59                | 29                  |
| GAT         | 124             | 138               | 124                 |
| GMNN        | 30              | 17                | 8                   |
| PSL-SAMPLES | 105             | 638               | 124                 |
| MLN-SAMPLES | 270             | 1947              | 166                 |

Table 4: Table showing runtimes for each of the approaches on the three datasets.

In Table 4 we show runtimes for each of the approaches. We observe that approaches that compute a point estimate are significantly faster compared to sample-based approaches. This is expected as point estimates are computed using efficient optimization approaches. Not surprisingly, sampling-based approaches are not as efficient, however their runtimes are still reasonable. The runtimes for MEAN and SAMPLES are the same, as we need to generate samples from the distribution for each of these approaches. Further, among PSL and MLN samplers, we observe that PSL is faster by a factor of two for Cora and three for Pubmed.

### Conclusion and Future Work

In this paper, we studied the task of estimating aggregate properties for networks with unobserved data. By comparing three graph neural networks and two probabilistic approaches on benchmark datasets, we show computing the expectation of aggregate properties over the distribution of unobserved random variables reduces the relative error. We have also proposed a blocked Gibbs sampling framework for...
PSL, that identifies pairwise correlated RVs and block samples them. We also showed the overall effectiveness of our approach through experiments. An interesting future direction is to combine graph neural network approaches with SRL models that can learn node representations and also infer a joint distribution over the unobserved data. Extending this analysis for networks with missing edges and nodes is another line of future work.

Acknowledgements

This work was partially supported by the National Science Foundation grants CCF-1740850 and IIS-1703331, AFRL and the Defense Advanced Research Projects Agency. Golnoosh Farnadi is supported by postdoctoral scholarships from IVADO through the Canada First Research Excellence Fund (CFREF) grant.

References

[Bach et al. 2017] Bach, S. H.; Broecheler, M.; Huang, B.; and Getoor, L. 2017. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research* 18:1–67.

[Broecheler and Getoor 2010] Broecheler, M., and Getoor, L. 2010. Computing marginal distributions over continuous markov networks for statistical relational learning. In *Advances in Neural Information Processing Systems*.

[Cook and Holder 2006] Cook, D. J., and Holder, L. B. 2006. *Mining graph data*. John Wiley & Sons, Inc.

[De Raedt and Kersting 2008] De Raedt, L., and Kersting, K. 2008. Probabilistic inductive logic programming. In *Probabilistic Inductive Logic Programming*.

[Dunne and Shneiderman 2013] Dunne, C., and Shneiderman, B. 2013. Motif simplification: improving network visualization readability with fan, connector, and clique glyphs. In *Human Factors in Computing Systems*.

[Friedman et al. 1999] Friedman, N.; Getoor, L.; Koller, D.; and Pfeffer, A. 1999. Learning probabilistic relational models. In *International Joint Conference on Artificial Intelligence*.

[Getoor and Taskar 2007] Getoor, L., and Taskar, B. 2007. *Introduction to Statistical Relational Learning*. The MIT Press.

[Gilks, Richardson, and Spiegelhalter 1995] Gilks, W. R.; Richardson, S.; and Spiegelhalter, D. 1995. *Markov chain Monte Carlo in practice*. Chapman and Hall/CRC.

[Gilmer et al. 2017] Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *International Conference on Machine Learning*.

[Hamilton, Ying, and Leskovec 2017] Hamilton, W.; Ying, Z.; and Leskovec, J. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*.

[Kipf and Welling 2017] Kipf, T. N., and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

[Liu et al. 2018] Liu, Y.; Safavi, T.; Dighe, A.; and Koutra, D. 2018. Graph summarization methods and applications: A survey. *Computing Surveys* 51:62.

[Musial and Juszczyszyn 2009] Musial, K., and Juszczyszyn, K. 2009. Properties of bridge nodes in social networks. In *Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems*.

[Neville and Jensen 2007] Neville, J., and Jensen, D. 2007. Relational dependency networks. *Journal of Machine Learning Research* 8:653–692.

[Niu et al. 2011] Niu, F.; Ré, C.; Doan, A.; and Shavlik, J. 2011. Tuffy: Scaling up statistical inference in markov logic networks using rdbms. In *Proceedings of the VLDB* 4(6).

[Poon and Domingos 2006] Poon, H., and Domingos, P. 2006. Sound and efficient inference with probabilistic and deterministic dependencies. In *National Conference on Artificial Intelligence*.

[Qu, Bengio, and Tang 2019] Qu, M.; Bengio, Y.; and Tang, J. 2019. Gmmn: Graph markov neural networks. In *International Conference on Machine Learning*.

[Qu et al. 2014] Qu, Q.; Liu, S.; Jensen, C. S.; Zhu, F.; and Faloutsos, C. 2014. Interestingness-driven diffusion process summarization in dynamic networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*.

[Rajaraman and Ullman 2011] Rajaraman, A., and Ullman, J. D. 2011. *Mining of massive datasets*.

[Richardson and Domingos 2006] Richardson, M., and Domingos, P. 2006. Markov logic networks. *Machine learning* 62:107–136.

[Scott 1988] Scott, J. 1988. Social network analysis. *Sociology* 22:109–127.

[Sen et al. 2008] Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Gallagher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI Magazine* 29:93–106.

[Shi et al. 2015] Shi, L.; Tong, H.; Tang, J.; and Lin, C. 2015. Vegas: Visual influence graph summarization on citation networks. *Knowledge and Data Engineering* 27:3417–3431.

[Tan, Steinbach, and Kumar 2006] Tan, P.-N.; Steinbach, M.; and Kumar, V. 2006. *Introduction to Data Mining*. Pearson Education.

[Veličković et al. 2018] Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph attention networks. *International Conference on Learning Representations*.

[Wasserman and Faust 1994] Wasserman, S., and Faust, K. 1994. *Social Network Analysis: Methods and Applications*. Cambridge University Press.

[Wu et al. 2014] Wu, Y.; Zhong, Z.; Xiong, W.; and Jing, N. 2014. Graph summarization for attributed graphs. In *International Conference on Information Science, Electronics and Electrical Engineering*.
[Yang, Cohen, and Salakhutdinov 2016] Yang, Z.; Cohen, W. W.; and Salakhutdinov, R. 2016. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning*. 