Graph Markov Neural Networks

Presenter: Jack Lanchantin

University of Virginia
https://qdata.github.io/deep2Read/

201906

Semi-supervised object classification

- Given graph $G = (V, E, \mathrm{x}_V)$, where V is a set of objects, E is a set of edges between objects, and x_V stands for the attributes of all the objects
- Given the labels y_L of a few labeled objects $L \subset V$, the goal is to predict the labels y_U for the remaining unlabeled objects $U = V \setminus L$.
- This problem has been extensively studied in the literature of both statistical relation learning (SRL) and graph neural networks (GNN). Essentially, both types of methods aim to model the distribution of object labels conditioned on the object attributes and the graph structure, i.e. $p(\mathrm{y}_V | \mathrm{x}_V, E)$

Statistical Relational Learning

- Most SRL methods model $p(\mathrm{y}_V|\mathrm{x}_V)$ with conditional random fields, which employ the following formulation:

$$p(\mathrm{y}_V|\mathrm{x}_V) = \frac{1}{Z(\mathrm{x}_V)} \prod_{(i,j)\in E} \psi_{i,j}(\mathrm{y}_i, \mathrm{y}_j, \mathrm{x}_V). \tag{1}$$

  where (i, j) is an edge in the graph G, and $\psi_{i,j}(\mathrm{y}_i, \mathrm{y}_j, \mathrm{x}_V)$ is the potential score defined on the edge (parameterized NN)
- Predicting the labels for unlabeled objects becomes an inference problem (inferring the posterior label distribution of the unlabeled objects $p(\mathrm{y}_U|\mathrm{y}_L, \mathrm{x}_V)$)
- Exact inference is usually infeasible

GNN

- GNN methods ignore the dependency of object labels and instead learn effective object representations for label prediction. The joint distribution of labels is fully factorized as:

$$p(\mathrm{y}_V|\mathrm{x}_V) = \prod_{n \in V} p(\mathrm{y}_n|\mathrm{x}_V). \tag{2}$$

- GNNs infer the label distribution $p(\mathrm{y}_n|\mathrm{x}_V)$ for each object n independently. For each object n, GNNs predict the label in the following way:

$$\mathrm{h} = g(\mathrm{x}_V, E) \quad p(\mathrm{y}_n|\mathrm{x}_V) = \mathrm{y}_n|\mathrm{softmax}(W\mathrm{h}_n),$$

where $\mathrm{h} \in \mathbb{R}^{|V| \times d}$ is the representations of all the objects, $W \in \mathbb{R}^{K \times d}$ is a linear transformation, with d as the dimension and K as the number of label classes

Graph Markov Neural Network (GMNN)

- The goal of GMNN is to combine the advantages of both SRL methods and GNNs, such that we can learn useful objective representations for predicting object labels, as well as model the dependency between object labels.
- GMNNs model the joint distribution of object labels conditioned on object attributes, $p(\mathrm{y}_V|\mathrm{x}_V)$, by using a CRF, which is optimized with a pseudolikelihood variational EM framework

Outline

GMNN Model

- Following SRL methods, we use a CRF field as in Eq. (1) to model the joint distribution of object labels conditioned on object attributes, i.e. $p_\phi(\mathrm{y}_V|\mathrm{x}_V)$, where the potential is defined over each edge, and ϕ is the model parameters
- We learn the model parameters ϕ by maximizing the log-likelihood function of the observed object labels:

$$\log p_\phi(\mathrm{y}_L|\mathrm{x}_V)$$

Minimizing the GMNN Log-Likelihood Function

- Directly maximizing the log-likelihood function is difficult, since many object labels are unobserved
- Instead optimize the evidence lower bound (ELBO) of the log-likelihood function:

$$\begin{aligned} &\log p_\phi(\mathrm{y}_L|\mathrm{x}_V) \geq \\ &\mathbb{E}_{q_\theta(\mathrm{y}_U|\mathrm{x}_V)}[\log p_\phi(\mathrm{y}_L, \mathrm{y}_U|\mathrm{x}_V) - \log q_\theta(\mathrm{y}_U|\mathrm{x}_V)], \end{aligned} \tag{3}$$

  where $q_\theta(\mathrm{y}_U|\mathrm{x}_V)$ can be any distributions over y_U, and the equation holds when $q_\theta(\mathrm{y}_U|\mathrm{x}_V) = p_\phi(\mathrm{y}_U|\mathrm{y}_L, \mathrm{x}_V)$
- According to the variational EM algorithm, such a lower bound can be optimized by alternating between a variational E-step and an M-step.

Pseudolikelihood Variational EM

$$\mathbb{E}_{q_\theta(\mathrm{y}_U|\mathrm{x}_V)}[\log p_\phi(\mathrm{y}_L, \mathrm{y}_U|\mathrm{x}_V) - \log q_\theta(\mathrm{y}_U|\mathrm{x}_V)]$$

- In the variational E-step (a.k.a., inference procedure), the goal is to fix p_ϕ and update the variational distribution $q_\theta(\mathrm{y}_U|\mathrm{x}_V)$ to approximate the true posterior distribution $p_\phi(\mathrm{y}_U|\mathrm{y}_L, \mathrm{x}_V)$.
- In the M-step (a.k.a., learning procedure), we fix q_θ and update p_ϕ to maximize the likelihood function ℓ:

$$\ell(\phi) = \mathbb{E}_{q_\theta(\mathrm{y}_U|\mathrm{x}_V)}[\log p_\phi(\mathrm{y}_L, \mathrm{y}_U|\mathrm{x}_V)]. \tag{4}$$

Pseudolikelihood Variational EM

- However, directly optimizing the likelihood function can be difficult, as we have to deal with the partition function in p_ϕ. To avoid computing the partition function, we instead optimize the pseudolikelihood function below:

$$\begin{aligned} \ell_{PL}(\phi) &\triangleq \mathbb{E}_{q_\theta(\mathrm{y}_U|\mathrm{x}_V)}[\sum_{n\in V} \log p_\phi(\mathrm{y}_n|\mathrm{y}_{V\setminus n}, \mathrm{x}_V)] \\ &= \mathbb{E}_{q_\theta(\mathrm{y}_U|\mathrm{x}_V)}[\sum_{n\in V} \log p_\phi(\mathrm{y}_n|\mathrm{y}_{\mathrm{NB(n)}}, \mathrm{x}_V)], \end{aligned} \tag{5}$$

where $\mathrm{NB}(n)$ is the neighbor set of n, and the equation is based on the independence properties of $p_\phi(\mathrm{y}_V|\mathrm{x}_V)$

Outline

Inference (E-step)

- The inference step aims to compute the posterior distribution $p_\phi(\mathrm{y}_U|\mathrm{y}_L, \mathrm{x}_V)$. Due to the complicated relational structures between object labels, exact inference is intractable.
- Therefore, we approximate it with another variational distribution $q_\theta(\mathrm{y}_U|\mathrm{x}_V)$ using the mean-field method:

$$q_\theta(\mathrm{y}_U|\mathrm{x}_V) = \prod_{n \in U} q_\theta(\mathrm{y}_n|\mathrm{x}_V). \tag{6}$$

where n is the index of unlabeled objects. In the variational distribution, all object labels are assumed to be independent.

Inference (E-step)

- To model the distribution of each object label in q_θ, we parameterize $q_\theta(\mathrm{y}_n|\mathrm{x}_V)$ with a graph neural network (GNN), which learns object representations for label prediction:

$$q_\theta(\mathrm{y}_n|\mathrm{x}_V) = \mathrm{P}(\mathrm{y}_n|\mathrm{softmax}(W_\theta \mathrm{h}_{\theta,n})). \tag{7}$$

Inference (E-step)

- With the mean-field formulation, the optimal distribution $q_\theta(\mathrm{y}_n|\mathrm{x}_V)$ satisfies the following condition:

$$\begin{aligned}&\log q_\theta(\mathrm{y}_n|\mathrm{x}_V) = \\ &\mathbb{E}_{q_\theta(\mathrm{y}_{\mathrm{NB}(n)\cap U}|\mathrm{x}_V)}[\log p_\phi(\mathrm{y}_n|\mathrm{y}_{\mathrm{NB}(n)}, \mathrm{x}_V)] + \text{const.}\end{aligned} \tag{8}$$

  where the right side of the condition involves expectation with respect to q_θ.

- To further simplify the condition, we estimate the expectation by using a sample drawn from $q_\theta(\mathrm{y}_{\mathrm{NB}(n)\cap U}|\mathrm{x}_V)$, resulting in:

$$\begin{aligned}&\mathbb{E}_{q_\theta(\mathrm{y}_{\mathrm{NB}(n)\cap U}|\mathrm{x}_V)}[\log p_\phi(\mathrm{y}_n|\mathrm{y}_{\mathrm{NB}(n)}, \mathrm{x}_V)] \\ &\simeq \log p_\phi(\mathrm{y}_n|\hat{\mathrm{y}}_{\mathrm{NB}(n)}, \mathrm{x}_V).\end{aligned} \tag{9}$$

Inference (E-step)

- In the above formula, $\hat{\mathrm{y}}_{\mathrm{NB}(n)} = \{\hat{\mathrm{y}}_k\}_{k \in \mathrm{NB}(n)}$ is defined as follows: for each unlabeled neighbor k of object n, we sample $\hat{\mathrm{y}}_k \sim q_\theta(\mathrm{y}_k | \mathrm{x}_V)$, and for each labeled neighbor k of object n, $\hat{\mathrm{y}}_k$ is set as the ground-truth label.
- In practice, we find that using one sample from $q_\theta(\mathrm{y}_{\mathrm{NB}(n) \cap U} | \mathrm{x}_V)$ yields comparable results with multiple samples
- Based on Eq. (8) and (9), the optimal $q_\theta(\mathrm{y}_n | \mathrm{x}_V)$ satisfies:

$$q_\theta(\mathrm{y}_n | \mathrm{x}_V) \approx p_\phi(\mathrm{y}_n | \hat{\mathrm{y}}_{\mathrm{NB}(n)}, \mathrm{x}_V), \tag{10}$$

Inference (E-step)

- To learn the optimal $q_\theta(\mathrm{y}_n|\mathrm{x}_V)$, we start with using the current value of θ to compute $p_\phi(\mathrm{y}_n|\hat{\mathrm{y}}_{\mathrm{NB}(n)}, \mathrm{x}_V)$.
- Then the value of $p_\phi(\mathrm{y}_n|\hat{\mathrm{y}}_{\mathrm{NB}(n)}, \mathrm{x}_V)$ is fixed as target, and we update θ to minimize the reverse KL divergence between $q_\theta(\mathrm{y}_n|\mathrm{x}_V)$ and the target $p_\phi(\mathrm{y}_n|\hat{\mathrm{y}}_{\mathrm{NB}(n)}, \mathrm{x}_V)$, yielding the objective function below:

$$O_{\theta,U} = \sum_{n \in U} \mathbb{E}_{p_\phi(\mathrm{y}_n|\hat{\mathrm{y}}_{\mathrm{NB}(n)}, \mathrm{x}_V)}[\log q_\theta(\mathrm{y}_n|\mathrm{x}_V)]. \tag{11}$$

Inference (E-step)

- q_θ can be also trained by predicting the labels for the labeled objects.
- Therefore, we also let q_θ maximize the following supervised objective function:

$$O_{\theta,L} = \sum_{n \in L} \log q_\theta(\mathrm{y}_n | \mathrm{x}_V). \tag{12}$$

  where y_n is the ground-truth label of n.
- By adding Eq. (11) and (12), we obtain the overall objective for optimizing θ:

$$O_\theta = O_{\theta,U} + O_{\theta,L}. \tag{13}$$

Outline

E-step to M-Step

$$\mathbb{E}_{q_\theta(\mathrm{y}_U|\mathrm{x}_V)}[\log p_\phi(\mathrm{y}_L, \mathrm{y}_U|\mathrm{x}_V) - \log q_\theta(\mathrm{y}_U|\mathrm{x}_V)],$$

- In the variational E-step (a.k.a., inference procedure), the goal is to fix p_ϕ and update the variational distribution $q_\theta(\mathrm{y}_U|\mathrm{x}_V)$ to approximate the true posterior distribution $p_\phi(\mathrm{y}_U|\mathrm{y}_L, \mathrm{x}_V)$.
- In the M-step (a.k.a., learning procedure), we fix q_θ and update p_ϕ to maximize the likelihood function below:

$$\ell(\phi) = \mathbb{E}_{q_\theta(\mathrm{y}_U|\mathrm{x}_V)}[\log p_\phi(\mathrm{y}_L, \mathrm{y}_U|\mathrm{x}_V)].$$

Learning (M-step)

- Maximize $\ell_{PL}(\phi)$:

$$\ell_{PL}(\phi) \triangleq \mathbb{E}_{q_\theta(\mathrm{y}_U|\mathrm{x}_V)}[\sum_{n\in V} \log p_\phi(\mathrm{y}_n|\mathrm{y}_{\mathrm{NB(n)}}, \mathrm{x}_V)], \tag{14}$$

- Only the conditional distribution $p_\phi(\mathrm{y}_n|\mathrm{y}_{\mathrm{NB}(n)}, \mathrm{x}_V)$ is required for p_ϕ in both the inference and learning steps (Eq. (11) and (5))
- Therefore, instead of defining the joint distribution of object labels $p_\phi(\mathrm{y}_V|\mathrm{x}_V)$ by specifying the potential function, we can simply focus on modeling the conditional distribution.

Learning (M-step)

- Here, we parameterize the conditional distribution $p_\phi(\mathrm{y}_n|\mathrm{y}_{\mathrm{NB}(n)}, \mathrm{x}_V)$ with another non-linear graph neural network model (GNN) because of its effectiveness:

$$p_\phi(\mathrm{y}_n|\mathrm{y}_{\mathrm{NB}(n)}, \mathrm{x}_V) = \mathrm{P}(\mathrm{y}_n|\mathrm{softmax}(W_\phi \mathrm{h}_{\phi,n})). \quad (15)$$

where the object representation $\mathrm{h}_{\phi,n}$ is learned by GNN_ϕ

Learning (M-step)

- When defining $p_\phi(y_n | y_{\mathrm{NB}(n)}, x_V)$, GNN_ϕ only uses the object labels $y_{\mathrm{NB}(n)}$ surrounding the object n as features, but GNN_ϕ is flexible to incorporate other features.
- For example, we can take both the surrounding object labels $y_{\mathrm{NB}(n)}$ and surrounding attributes $x_{\mathrm{NB}(n)}$ as features in GNN_ϕ (as discussed in Experiments)

Learning (M-step)

- When optimizing p_ϕ to maximize Eq. (5), we estimate the expectation in Eq. (5) by drawing a sample from $q_\theta(\mathrm{y}_U|\mathrm{x}_V)$.
- More specifically, if n is an unlabeled object, then we sample $\hat{\mathrm{y}}_n \sim q_\theta(\mathrm{y}_n|\mathrm{x}_V)$, and otherwise we set $\hat{\mathrm{y}}_n$ as the ground-truth label.
- ϕ can be optimized by maximizing the following objective function:

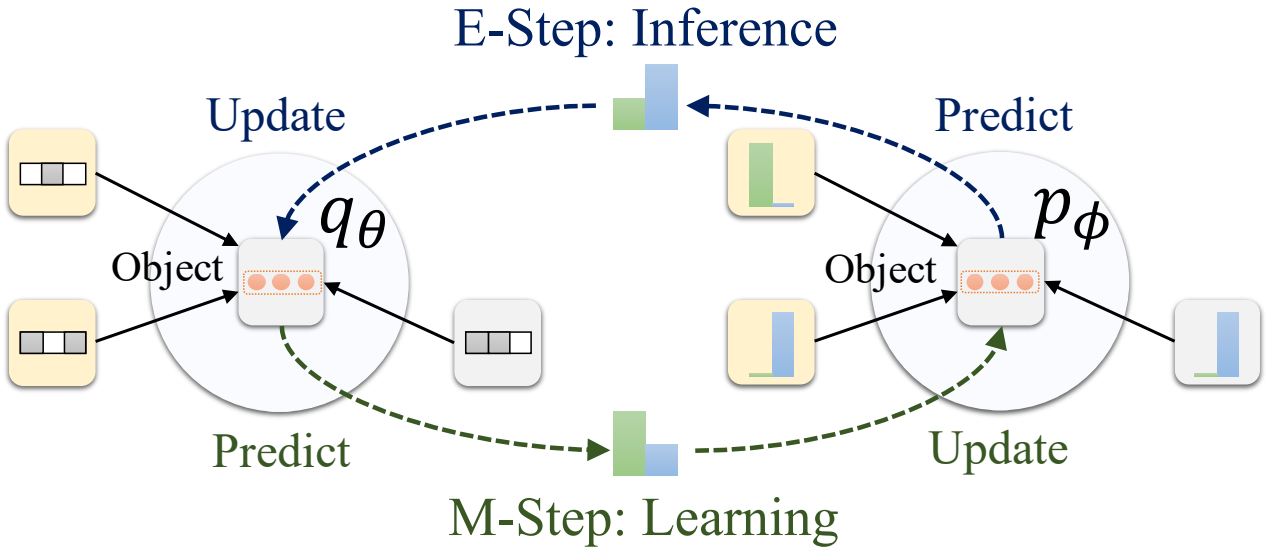
$$O_\phi = \sum_{n \in V} \log p_\phi(\hat{\mathrm{y}}_n|\hat{\mathrm{y}}_{\mathrm{NB(n)}}, \mathrm{x}_V). \tag{16}$$

Outline

Optimization

- First pre-train the inference model q_θ with the labeled objects.
- Then we alternatively optimize p_ϕ and q_θ until convergence.
- Afterwards, both p_ϕ and q_θ can be employed to infer the labels of unlabeled objects.
  - In practice, we find that q_θ consistently outperforms p_ϕ, and thus we use q_θ to infer object labels by default

Optimization



- q_θ uses the attributes of its surrounding objects to learn its representation, and further predicts the label
- In contrast, p_ϕ uses the *labels* of the surrounding objects as features. If a neighbor is unlabeled, use label sampled from q_θ
- In the E-step, p_ϕ predicts the label for the central object, which is then treated as target to update q_θ
- In the M-step, q_θ predicts the label for the central object, which serves as the target data to update p_ϕ

Optimization

Algorithm 1 Optimization Algorithm

Input: A graph G, some labeled objects $(L, \mathbf{y}_L)$.
Output: Object labels $\mathbf{y}_U$ for unlabeled objects U.
Pre-train q_θ with $\mathbf{y}_L$ according to Eq. (12).
while not converge **do**
  ⊡ **M-Step: Learning Procedure**
  Annotate unlabeled objects with q_θ.
  Denote the sampled labels as $\hat{\mathbf{y}}_U$.
  Set $\hat{\mathbf{y}}_V = (\mathbf{y}_L, \hat{\mathbf{y}}_U)$ and update p_ϕ with Eq. (15).
  ⊡ **E-Step: Inference Procedure**
  Annotate unlabeled objects with p_ϕ and $\hat{\mathbf{y}}_V$.
  Denote the predicted label distribution as $p_\phi(\mathbf{y}_U)$.
  Update q_θ with Eq. (11), (12) based on $p_\phi(\mathbf{y}_U), \mathbf{y}_L$.
end while
Classify each unlabeled object n based on $q_\theta(\mathbf{y}_n|\mathbf{x}_V)$.

Results

Table 2. Results of object classification (%). [*] means the results are taken from the corresponding papers.

Category	Algorithm	Cora	Citeseer	Pubmed
SSL	LP	74.2	56.3	71.6
SRL	PRM	77.0	63.4	68.3
	RMN	71.3	68.0	70.7
	MLN	74.6	68.0	75.3
GNN	Planetoid *	75.7	64.7	77.2
	GCN *	81.5	70.3	79.0
	GAT *	83.0	72.5	79.0
GMNN	W/o Attr. in p_ϕ	83.4	**73.1**	81.4
	With Attr. in p_ϕ	**83.7**	72.9	**81.8**