GMNN: Graph Markov Neural Networks

Meng Qu[1,2], Yoshua Bengio[1,2,4], Jian Tang[1,3,4]

[1]Quebec AI Institute (Mila)

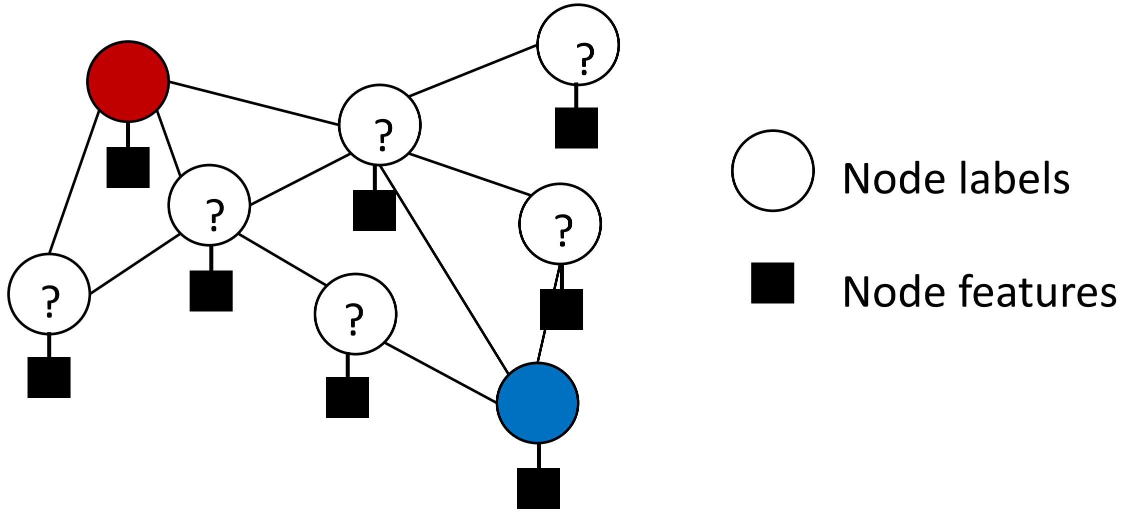
[2]University of Montreal

[3]HEC Montreal

[4]Canadian Institute for Advanced Research (CIFAR)

Semi-supervised Node Classification

- Given a graph $G = (V, E, \mathbf{x}_{\mathrm{V}})$
  - $V = V_L \cup V_U$: nodes
  - E: edges
  - $\mathbf{x}_{\mathrm{V}}$: node features



- Give some labeled nodes V_L, we want to infer the labels of the rest of nodes V_U
- Many other tasks on graphs can be formulated as node classification
  - E.g., link classification

Related Work: Statistical Relational Learning

- Model the joint distribution of the node labels given the node features, i.e., $p(\mathbf{y}_{\mathrm{V}}|\mathbf{x}_{\mathrm{V}})$, with conditional random fields

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

- Pros
  - Capable of modeling the dependency between the node labels
- Cons
  - Some manually defined potential functions
  - Limited model capacity
  - Difficult inference due to the complicated graph structures

Related Work: Graph Neural Networks

- Learn effective node representations by non-linear feature propagations
  - Graph convolutional Networks (Kipf et al. 2016)
  - Graph attention networks (Veličković et al. 2017)
  - Neural message passing (Gilmer et al. 2017)
- Pros
  - Learning effective node representations
  - High model compacity through multiple non-linear graph convolutional layers
- Cons
  - Ignoring the dependency between node labels

GMNN: Graph Markov Neural Networks

- Towards combining statistical relational learning and graph neural networks
  - Learning effective node representations
  - Modeling the label dependencies of nodes
- Model the joint distribution of node labels $\mathbf{y}_V$ conditioned on node features $\mathbf{x}_V$, i.e., $p_\phi(\mathbf{y}_V|\mathbf{x}_V)$
- Can be effectively optimized through pseudolikelihood Variational-EM

Two Graph Neural Networks co-train with Each Other

- Two GNNs:
  - p_ϕ: learning network, modeling the label dependency by **non-linear label propagation**
  - q_θ: inference network, learning the node representations by **non-linear feature propagation**
- q_θ infers the labels of unlabeled nodes trained with supervision from p_ϕ and labeled nodes
- p_ϕ is trained with a fully labeled graph, where the unlabeled nodes are labeled by q_θ

Experimental Results

- State-of-the-art performance in multiple tasks

Table: Semi-supervised Node Classification

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**

Table: Link Classification

Category	Algorithm	Bitcoin Alpha	Bitcoin OTC
SSL	LP	59.68	65.58
SRL	PRM	58.59	64.37
	RMN	59.56	65.59
	MLN	60.87	65.62
GNN	DeepWalk	62.71	63.20
	GCN	64.00	65.69
GMNN	W/o Attr. in p_ϕ	65.59	66.62
	With Attr. in p_ϕ	**65.86**	**66.83**

Table: Unsupervised Node Representation Learning

Category	Algorithm	Cora	Citeseer	Pubmed
GNN	DeepWalk *	67.2	43.2	65.3
	DGI *	82.3	**71.8**	76.8
GMNN	With only q_θ .	78.1	68.0	79.3
	With q_θ and p_ϕ	**82.8**	71.5	**81.6**

Code available at:

https://github.com/DeepGraphLearning/GMNN

Come to our Poster at #7

Jun 11th 06:30-09:00 PM @ Pacific Ballroom