# GMNN: Graph Markov Neural Network

Cheng Peng

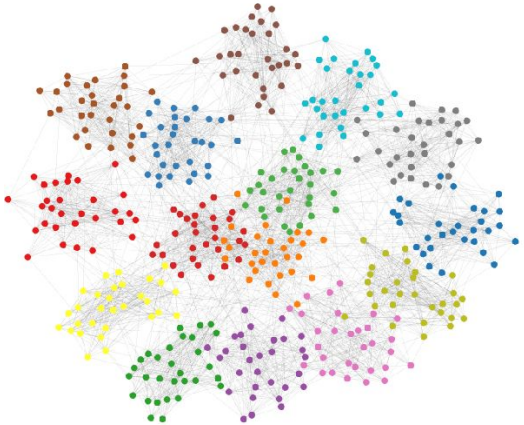
Chenlei Song

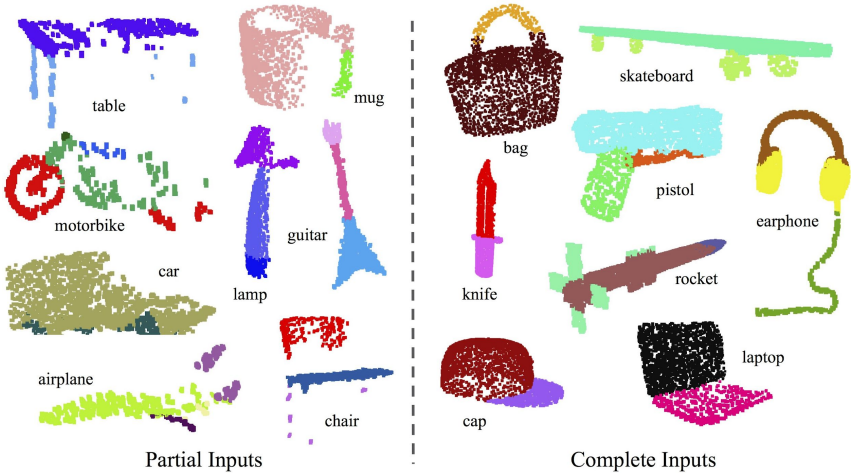
Qiyue Yao

Shijia Hu

# Motivation

- Relational Data Modeling
- Semi-Supervised Object Classification
- Link Problem





# Problem Definition

Consider the problem of semi-supervised object classification:

$$G = (V, E, \mathbf{x}_V)$$

The goal is to predict the labels $\mathbf{y}_U$ for unlabeled objects, given some of the labeled objects $L \subset V$.

More generally, one wants to find $p_\phi(\mathbf{y}_U | \mathbf{y}_L, \mathbf{x}_V)$

# Traditional Methods

- Statistical Relational Learning (SRL)
- Graph Neural Networks (GNN)

# Statistical Relational Learning

- Recap: Conditional Random Field -- Models label dependency w/ conditional distribution

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

- Example: Part-of-Speech tagging with Linear Chain CRF using feature functions
- Problem:
  - Potential functions relies on handcrafted feature functions
  - Intractable posterior inference

# Graph Neural Networks

- Convolutional Neural Networks (CNN) operates on regular Euclidean 2D grid and 1D sequence.
- Non-Euclidean data:
  - Consumer-product interaction in e-commerce system.
  - Molecule bioactivity
  - Citation network
  - ...

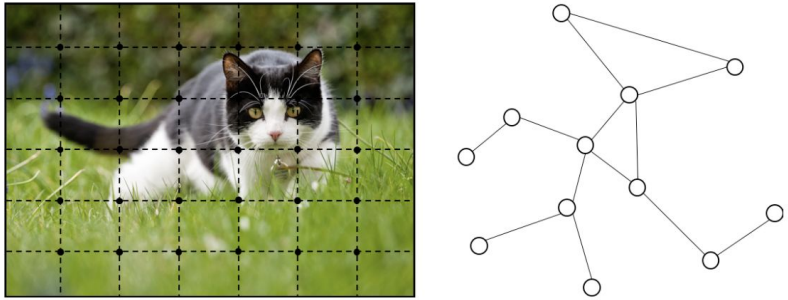
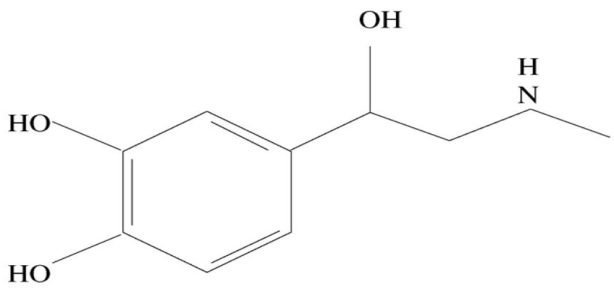


Fig. 1. Left: image in Euclidean space. Right: graph in non-Euclidean space

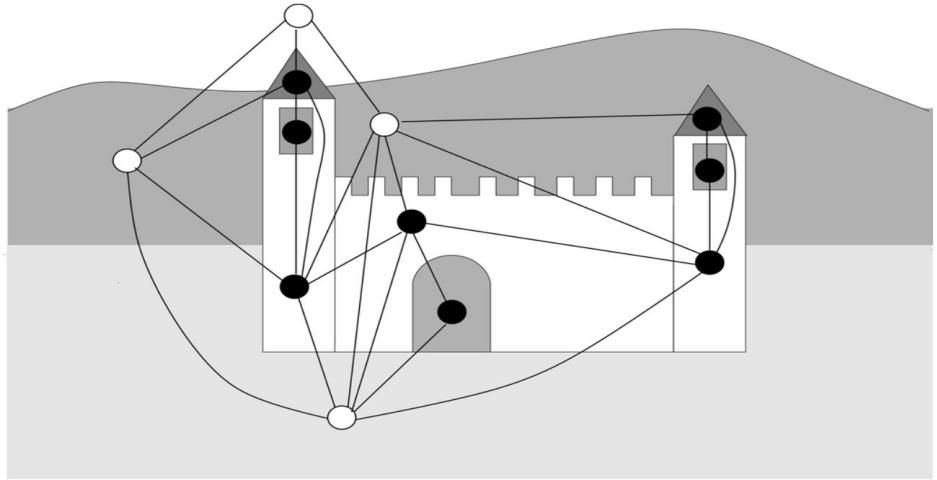
# GNN explained

- Learn a state embedding for each node on the graph.
- State embeddings are used to produce an output, such as a node label in our scenario.
- A parametric function (local transition function) updates the node state according to the input neighborhood.
- A local output function will describe how the output is produced.

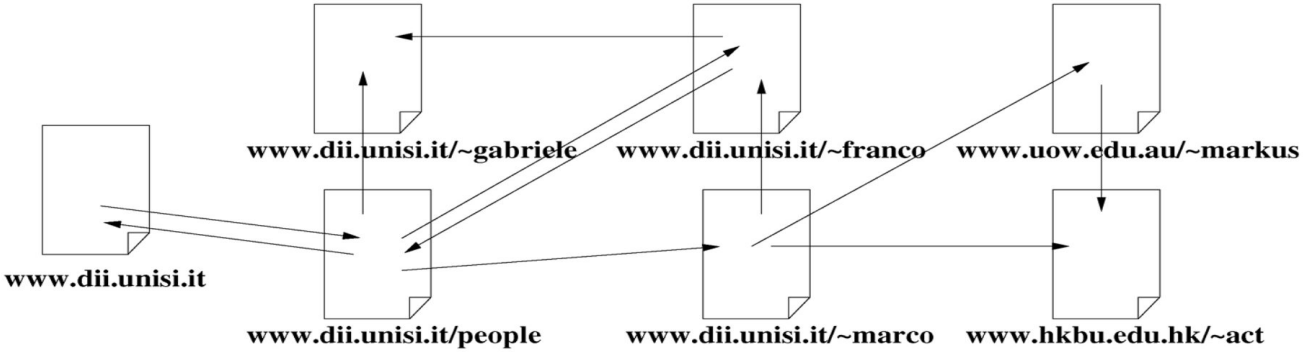
Whiteboard time!


OH
H
N
HO
HO

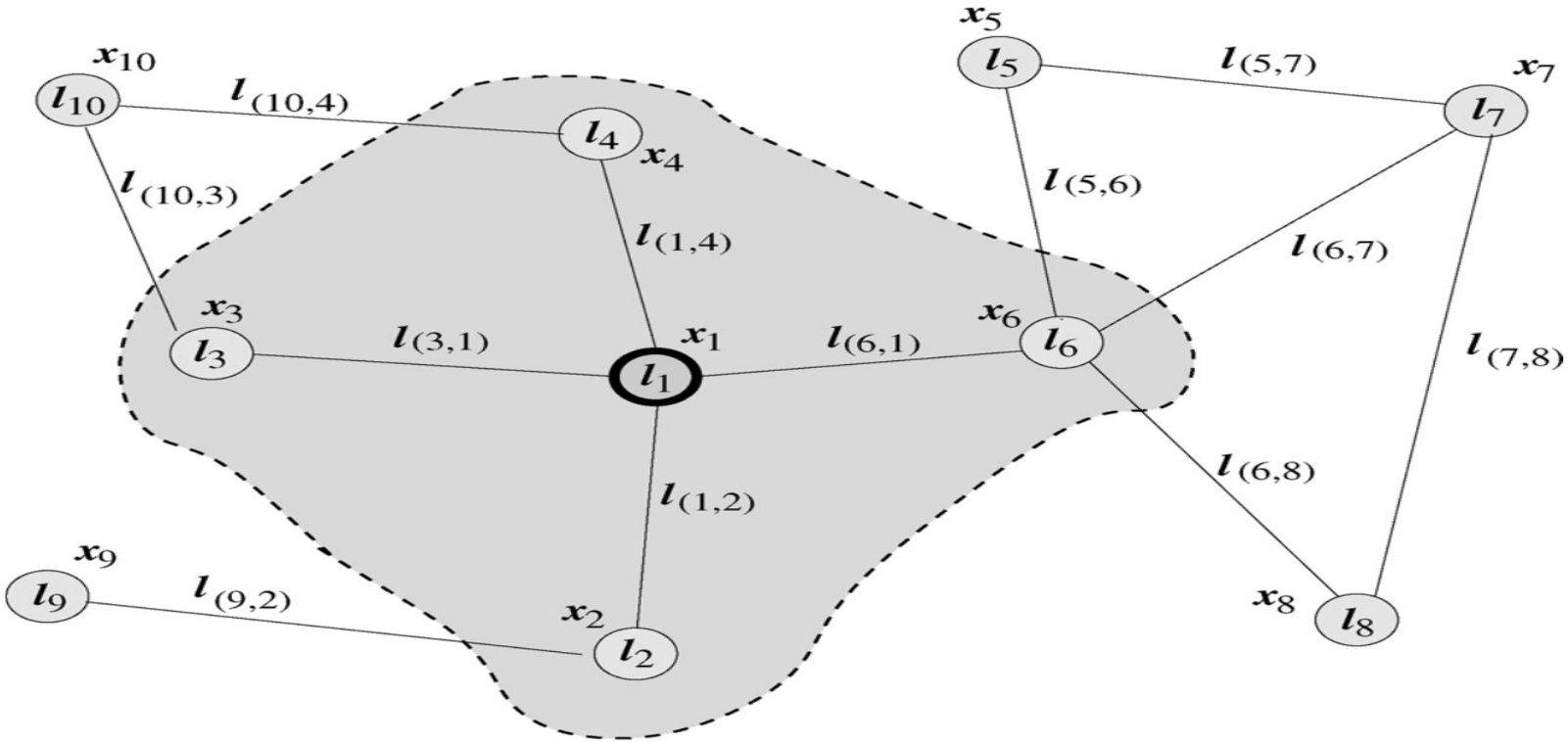
(a)



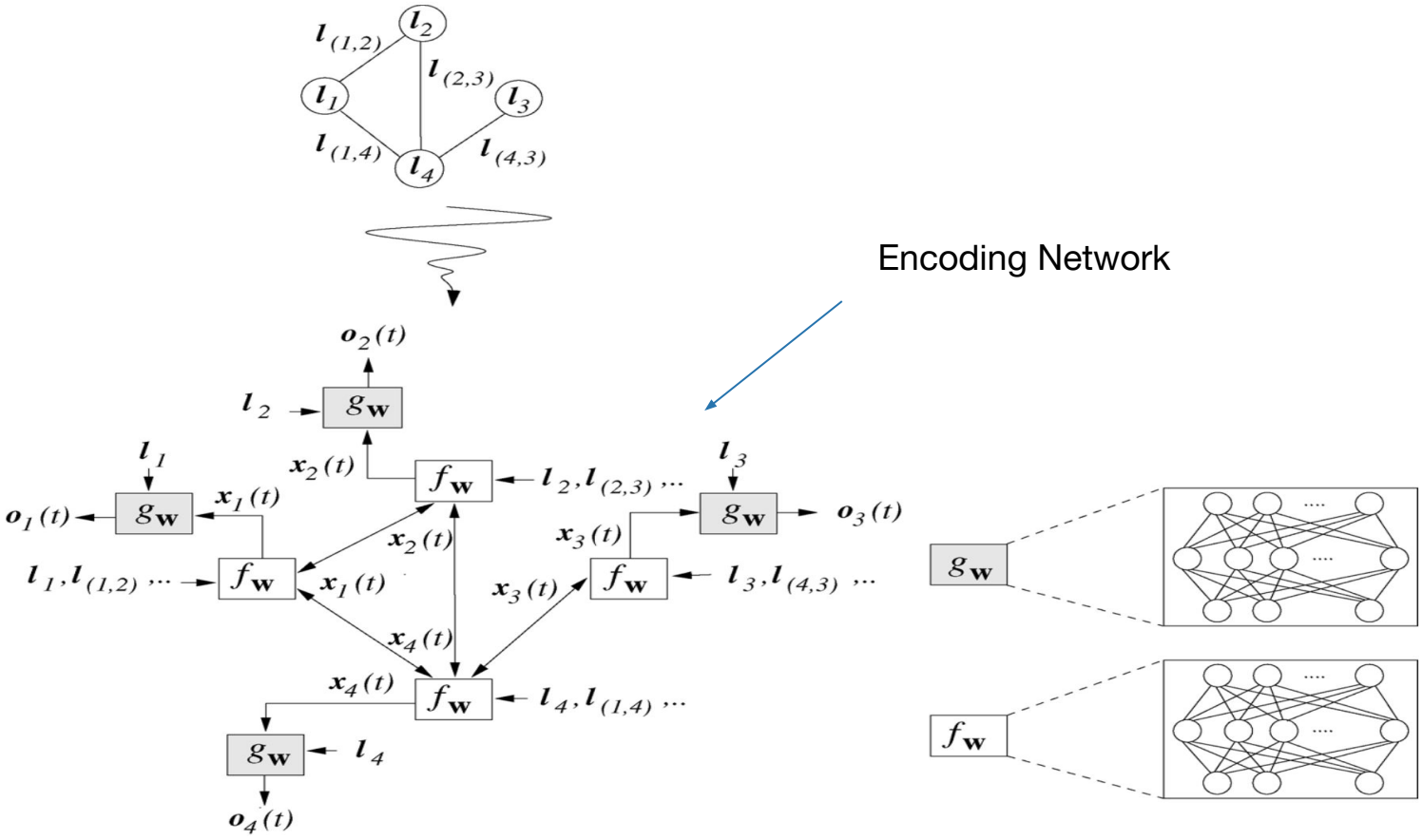
(b)

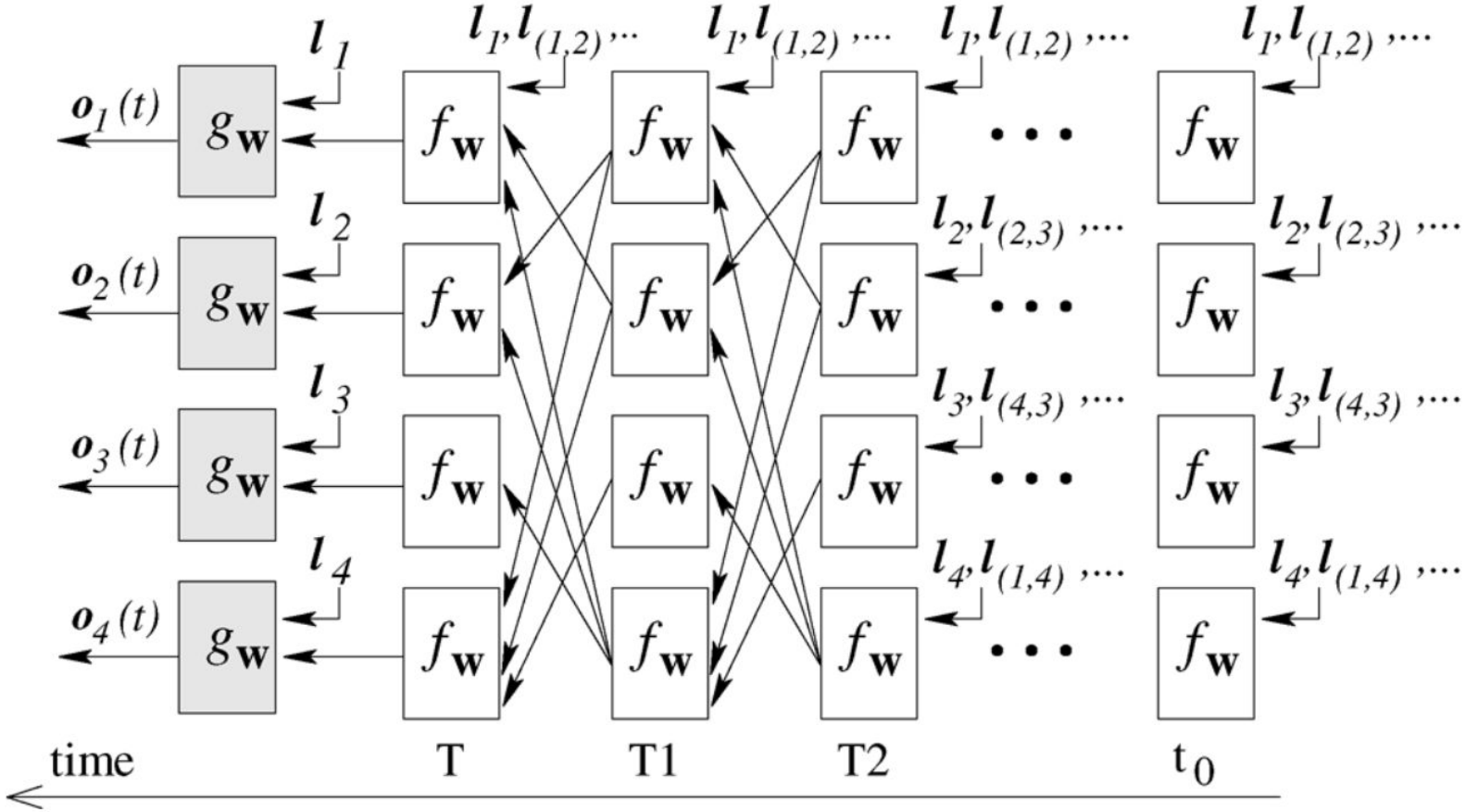

www.dii.unisi.it/~gabriele
www.dii.unisi.it/~franco
www.uow.edu.au/~markus
www.dii.unisi.it
www.dii.unisi.it/people
www.dii.unisi.it/~marco
www.hkbu.edu.hk/~act

(c)




$$x_1 = f_w(\underbrace{l_1, l_{(1,2)}, l_{(3,1)}, l_{(1,4)}, l_{(6,1)}}_{l_{co[1]}}, \underbrace{x_2, x_3, x_4, x_6}_{x_{ne[1]}}, \underbrace{l_2, l_3, l_4, l_6}_{l_{ne[n]}})$$


Encoding Network
$l_{(1,2)}$
$l_{(2,3)}$
$l_{(1,4)}$
$l_{(4,3)}$
$l_1$
$l_2$
$l_3$
$l_4$
$o_2(t)$
$o_1(t)$
$o_3(t)$
$o_4(t)$
$x_1(t)$
$x_2(t)$
$x_3(t)$
$x_4(t)$
$g_{\mathbf{w}}$
$f_{\mathbf{w}}$
$l_2, l_{(2,3)} \ldots$
$l_1, l_{(1,2)} \ldots$
$l_3, l_{(4,3)} \ldots$
$l_4, l_{(1,4)} \ldots$


$o_1(t)$
$o_2(t)$
$o_3(t)$
$o_4(t)$
$g_{\mathbf{w}}$
$l_1$
$l_2$
$l_3$
$l_4$
$l_1, l_{(1,2)}, \ldots$
$l_1, l_{(1,2)}, \ldots$
$l_1, l_{(1,2)}, \ldots$
$l_1, l_{(1,2)}, \ldots$
$l_2, l_{(2,3)}, \ldots$
$l_2, l_{(2,3)}, \ldots$
$l_3, l_{(4,3)}, \ldots$
$l_3, l_{(4,3)}, \ldots$
$l_4, l_{(1,4)}, \ldots$
$l_4, l_{(1,4)}, \ldots$
$f_{\mathbf{w}}$
time
T
T1
T2
$t_0$

$$p(\mathbf{y}_V|\mathbf{x}_V) = \prod_{n \in V} p(\mathbf{y}_n|\mathbf{x}_V)$$

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

# GMNN: Graph Markov Neural Networks

**Model:** joint distribution of object labels conditioned on object attributes

$$p_\phi(\mathbf{y}_V | \mathbf{x}_V)$$

**Learn**: model parameters Φ by maximizing the log-likelihood function (of observed object labels)

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

# Recall: Evidence Lower Bound (ELBO)

**Jensen's Inequality:** if $f$ is concave, $f(\mathrm{E}[X]) \geq \mathrm{E}[f(X)]$

$$
\begin{aligned}
\log p(x) &= \log \int_z p(x,z) \\
&= \log \int_z p(x,z) \frac{q(z)}{q(z)} \\
&= \log \left( \mathrm{E}_q \left[ \frac{p(x,Z)}{q(z)} \right] \right) \\
&\geq \mathrm{E}_q[\log p(x,Z)] - \mathrm{E}_q[\log q(Z)].
\end{aligned}
$$

Optimize ELBO of $\log p_\phi(\mathbf{y}_L | \mathbf{x}_V) \geq$

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

# Optimize: Pseudolikelihood Variational EM

**E-step (inference):** fix $p_\phi$ update variational distribution $q_\theta(\mathbf{y}_U|\mathbf{x}_V)$

**M-step (learning):** fix $q_\theta$ update $p_\phi$ to maximize the likelihood function

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

**Pseudolikelihood**

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

$$= \mathbb{E}_{q_\theta(\mathbf{y}_U|\mathbf{x}_V)}[\sum_{n\in V} \log p_\phi(\mathbf{y}_n|\boxed{\mathbf{y}_{\mathrm{NB(n)}}}, \mathbf{x}_V)]$$

# Inference (E-step)

**E-Step: a.k.a inference procedure — compute posterior distribution**

$$p_\phi(\mathbf{y}_U | \mathbf{y}_L, \mathbf{x}_V)$$

**instead, we use** $q_\theta(Y_U | X_V)$ **to approximate the real distribution**

**— fix $\phi$ and optimize $q_\theta$**

# Inference (E-step)

**mean-field method:**

$$q_\theta(\mathbf{y}_U|\mathbf{x}_V) = \prod_{n \in U} q_\theta(\mathbf{y}_n|\mathbf{x}_V)$$

**— all object labels are assumed to be independent**

# Inference (E-step)

**we use a GNN to parameterize $q_\theta$**

$$q_\theta(\mathbf{y}_n|\mathbf{x}_V) = \mathrm{Cat}(\mathbf{y}_n|\mathrm{softmax}(W_\theta \mathbf{h}_{\theta,n}))$$

- **$q_\theta$ is formulated as a categorical distribution**
- **$h_{\theta,n}$ is learned by a GNN with $X_V$ as features and $\theta$ as parameters**

# Inference (E-step)

**under mean-field method, the optimal distribution is:**

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

**— proof …**

# Inference (E-step)

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

– we have $q_\theta$ in right side. still intractable…

**take a sample to estimate the expectation**

# Inference (E-step)

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

**for unlabeled neighbor:** $\hat{\mathbf{y}}_k \sim q_\theta(\mathbf{y}_k|\mathbf{x}_V)$

**for labeled neighbor:** $\hat{y}_k$ **set as the ground-truth label**

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

# Inference (E-step)

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

**now, we only need to optimize $q_\theta$ to approximate the right side**

**— minimize the reverse KL divergence**

$$-KL(P_\phi(y_n \,|\, \hat{y}_{NB(n)}, X_V) \,||\, q_\theta(y_n \,|\, X_V))$$

# Inference (E-step)

**we get the objective function:**

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

**We can also use labeled objects to train $q_\theta$**

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

– So, the overall objective function for optimizing

$$O_\theta = O_{\theta,U} + O_{\theta,L}$$

# Learning (M-step)

**Log pseudolikelihood:**

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

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

**Parameterize with another non-linear GNN:**

$$p_\phi(\mathbf{y}_n|\mathbf{y}_{\mathrm{NB}(n)}, \mathbf{x}_V) = \mathrm{Cat}(\mathbf{y}_n|\mathrm{softmax}(W_\phi \mathbf{h}_{\phi,n}))$$

# Learning (M-step)

Estimate expectation by drawing a sample from $q_\theta(\mathbf{y}_U|\mathbf{x}_V)$

$\hat{\mathbf{y}}_n$

n unobserved: $\hat{\mathbf{y}}_n \sim \boxed{q_\theta(\mathbf{y}_n|\mathbf{x}_V)}$

n observed: *Ground-Truth label*

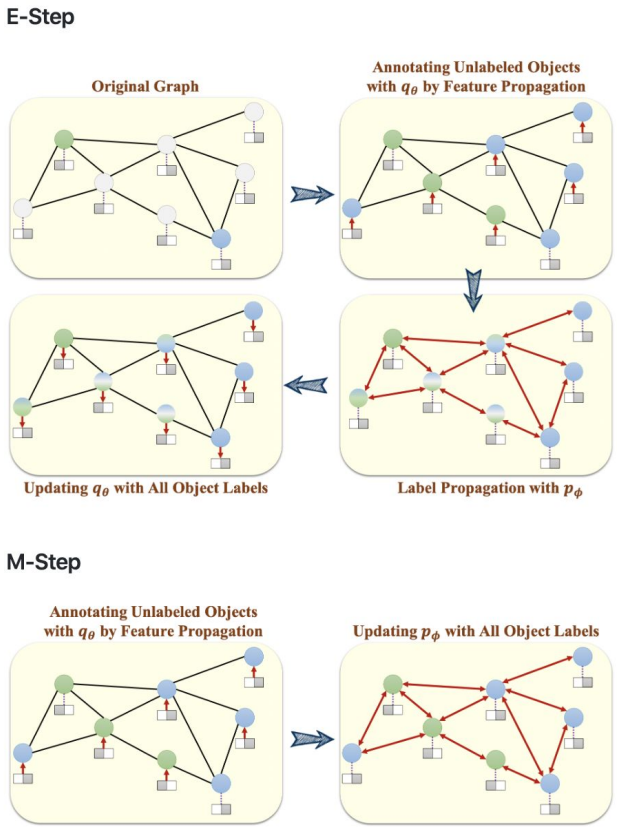
Object function:

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

# Overall

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



# Applications

**Unsupervised Node Representation Learning**

- No labeled nodes
- Instead, predict the neighbors for each node
- E-step: infer the neighbor distribution for each node with $q_\theta$
- M-step: update the $p_\phi$ to model the local dependency of the inferred neighbor distributions

# Applications

**Link classification**

- Use the original graph G to construct a dual graph $\tilde{G}$
- The object set $\tilde{V}$ in dual graph corresponds to the link set $\tilde{E}$ in original graph
- Two objects are linked in $\tilde{G}$ if their corresponding links in G share a node

# Compared Algorithms

**GNN Methods**

- Graph Convolutional Network
- Graph Attention Network

**SRL Methods**

- Probabilistic Relational Model
- Relational Markov Network
- Markov Logic Network

**SSL Methods**

- Label Propagation

# Results

| 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 $p_\phi$ | 83.4 | **73.1** | 81.4 |
| | With Attr. in $p_\phi$ | **83.7** | 72.9 | **81.8** |

| 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 $p_\phi$ | 65.59 | 66.62 |
| | With Attr. in $p_\phi$ | **65.86** | **66.83** |

# Questions

1. Briefly describe the motivation for combining SRL & GNN respectively

2. In GMNN, we used GNN twice. What's the main difference between them?

# Reference

https://www.cs.princeton.edu/courses/archive/fall11/cos597C/lectures/variational-inference-i.pdf

http://www.ipam.ucla.edu/abstract/?tid=16001&pcode=GLWS4

https://github.com/DeepGraphLearning/GMNN