Enhancing Network Embedding with Auxiliary Information: An Explicit Matrix Factorization Perspective

Junliang Guo, Linli Xu, Xunpeng Huang, and Enhong Chen

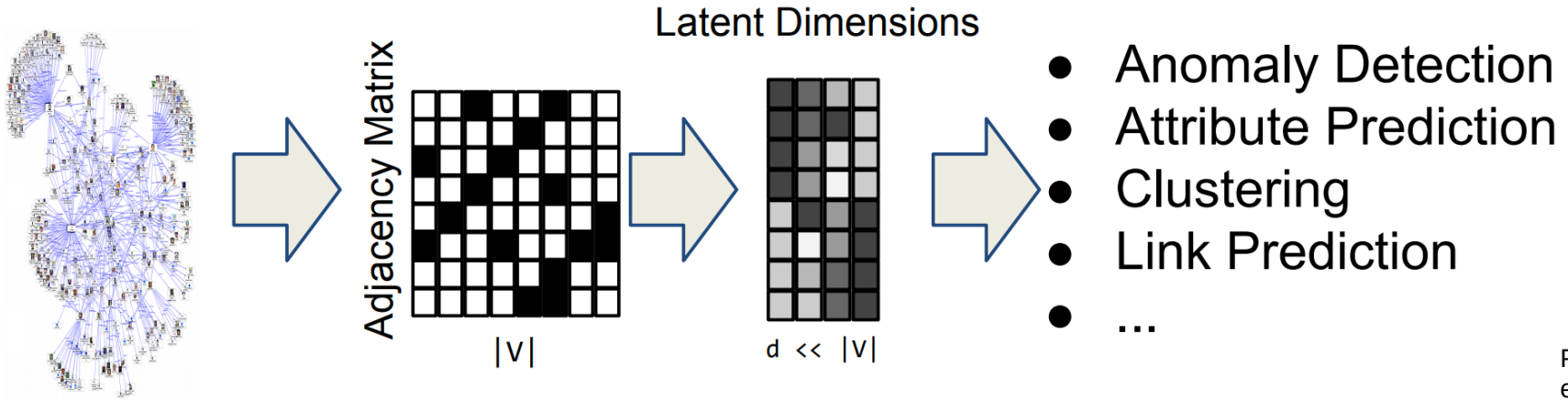
Anhui Province Key Laboratory of Big Data Analysis and Application,
School of Computer Science and Technology,
University of Science and Technology of China
guojunll@mail.ustc.edu.cn, linlixu@ustc.edu.cn
hxpsola@mail.ustc.edu.cn, cheneh@ustc.edu.cn

Reporter: Junliang Guo
Date: 22 May, 2018

Outline

Background

- Network Embedding
  - **To learn efficient low-dimensional representation for networks.**
- Necessary?
  - Encode non-Euclidean data into Euclidean space
  - Preserves structural information
  - A bunch of application scenarios
  - Ubiquitous in real world. Social networks, protein-interaction networks, citation networks, etc.



Perozzi et.al., 2016

Background

- Why is it hard?
  - **Mainly from network's non-Euclidean property**
  - Arbitrary ordering of nodes
  - Complicated structures. E.g., (un)directed / attributed edges
  - Auxiliary information. E.g., label information, node features
- How to handle?
  - Word embedding inspired methods
    1. Utilize random walk to convert networks to node sequences
    2. Skip-Gram based optimization algorithms
  - Graph convolutional networks

    Take specific graph Laplacian matrix as input
  - What we focus?

    Utilize auxiliary information to learn better embeddings

Related Work

- Random walk based methods
  - DeepWalk (Perozzi et.al.), node2vec (Leskovec et.al.)
- RW inspired matrix factorization methods
  - TADW (Yang et.al.), DMF (Zhang et.al.), HSCA (Zhang et.al)
- Neural network based methods
  - GCN (Kipf et.al.), Planetoid (Yang et.al.)

Method	Features	Supervision	Embedding based
DeepWalk & node2vec	✗	✗	√
TADW	√	✗	√
DMF	√	√	✗
HSCA	√	✗	√
GCN & Planetoid	√	√	✗
APNE (Ours)	√	√	√

Related Work

- Random walk based methods
  - DeepWalk (Perozzi et.al.), node2vec (Leskovec et.al.)
- RW inspired matrix factorization methods ← Our method
  - TADW (Yang et.al.), DMF (Zhang et.al.), HSCA (Zhang et.al)
- Neural network based methods
  - GCN (Kipf et.al.), Planetoid (Yang et.al.)

Method	Features	Supervision	Embedding based
DeepWalk & node2vec	✗	✗	√
TADW	√	✗	√
DMF	√	√	✗
HSCA	√	✗	√
GCN & Planetoid	√	√	✗
APNE (Ours)	√	√	√

Outline

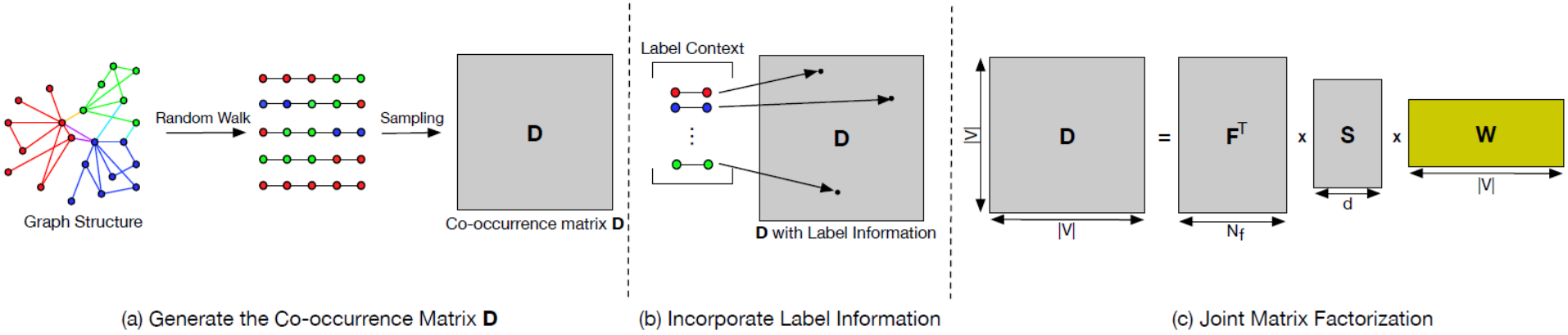
APNE

- Auxiliary information Preserved Network Embedding

Our framework can be divided into three components:

- Generate a matrix which preserves structural proximities
- Incorporate label information in an unsupervised manner
- Joint matrix factorization considering feature information

(Mainly inspired by Li et.al., 2015)



Problem Definition

- Given an undirected network $G = \{V, E\}$

Basic notes

Nodes	V
Edges	E
Content matrix	$\boldsymbol{F} \in \mathbb{R}^{\lvert V\rvert \times f}$
Adjacency matrix	$\boldsymbol{A} \in \mathbb{R}^{\lvert V\rvert \times \lvert V\rvert}$
Transition matrix	$\boldsymbol{P} \in \mathbb{R}^{\lvert V\rvert \times \lvert V\rvert}$
Rooted PageRank	$\boldsymbol{S}^{\mathrm{RPR}} \in \mathbb{R}^{\lvert V\rvert \times \lvert V\rvert}$

Where

$$A_{i,j} = w_{i,j}$$

$$P_{i,j} = \frac{w_{i,j}}{\sum_{k=1}^{|V|} w_{i,k}}$$

$$\boldsymbol{S}^{\mathrm{RPR}} = (1 - \beta_{\mathrm{RPR}})(\boldsymbol{I} - \beta_{\mathrm{RPR}}\boldsymbol{P})^{-1}$$

- Goal

To learn an embedding matrix $\boldsymbol{W} \in \mathbb{R}^{|V| \times d}$, where d is a small number of latent dimensions. $\boldsymbol{W}$ should well preserve the structural properties of G, which can be evaluated through downstream tasks such as node classification and link prediction.

Matrix Generation

- How?

  Generate a co-occurrence matrix $\boldsymbol{D}$ through general random walk sampling algorithm

Algorithm 1 Sampling the general co-occurrence matrix

Input: The transition matrix $\boldsymbol{P}$, window size l
Output: Co-occurrence matrix $\boldsymbol{D}$
1: Sample random walks C based on $\boldsymbol{P}$
2: **for** every node sequence in C **do**
3: Uniformly sample (i, j) with $|i - j| < l$
4: $D_{v_i,v_j} = D_{v_i,v_j} + 1$
5: **end for**

Matrix Generation

- Why?
  - This matrix $\boldsymbol{D}$ should contain the structural information of the network G
  - We prove that $\boldsymbol{D}$ is an approximation to Rooted PageRank, a high-order proximity of network, with a bounded l2 norm

Theorem 2. *When l is sufficiently large, for $\boldsymbol{D}^{nor}$ defined as $\boldsymbol{D}^{nor} = \frac{1}{l}\sum_{k=1}^{l}\boldsymbol{P}^k$, and $K = \lfloor -\frac{\log l(1-\beta_{RPR})}{\log \beta_{RPR}} \rfloor$, the ℓ-2 norm of the difference between $\boldsymbol{D}^{nor}$ and $\boldsymbol{S}^{RPR}$ can be bounded by K:*

$$\left\| \boldsymbol{S}^{RPR} - \boldsymbol{D}^{nor} \right\|_2 \leq 2 - 2\beta_{RPR}^{K+1} \tag{4}$$

- Advances
  - A general case of the matrix that TADW constructs
  - Elements are integers instead of real numbers, which lays the foundation of the next component

Incorporate Label Information

- Motivation
  - Can we leverage label information in an unsupervised manner? I.e., without incorporating downstream classifiers
- What we have?
  - In $\boldsymbol{D}$, larger $D_{i,j}$ indicates more similar between node V_i and V_j
  - Assume this holds for nodes in the same class. I.e., node V_i and V_j are more similar if they have a same label
- What we do?
  - Incorporate label sampling into the random walk sampling procedure

Incorporate Label Information

Algorithm 2 Sampling general co-occurrence matrix with structure and label context

Input: The transition matrix $\boldsymbol{P}$, labeled nodes L, parameters m, l, d
Output: Co-occurrence matrix $\boldsymbol{D}$
Sample random walks C of length d based on $\boldsymbol{P}$
for every node sequence in C **do**
    Uniformly sample (i, j) with $|i - j| < l$
    $D_{v_i,v_j} = D_{v_i,v_j} + 1$
end for
for $k = 1$ to m **do**
    Uniformly sample a node v_i in L
    Uniformly sample a node v_j with the same label as node v_i
    $D_{v_i,v_j} = D_{v_i,v_j} + 1$
end for

- A hyper-parameter m controls the ratio between structural and label sampling
- Simple but efficient. Verified empirically

Joint Matrix Factorization

- Incorporate the content matrix $\boldsymbol{F}$
  - Formulation

$$\min_{\boldsymbol{W},\boldsymbol{S}} MF(\boldsymbol{D}, \boldsymbol{F}^T \boldsymbol{S}\boldsymbol{W})$$

  - Inspired by Inductive Matrix Completion (Natarajan et.al., 2014)
- How to solve?
  - First rewrite in a representation learning view

$$\min_{\boldsymbol{W},\boldsymbol{S}} \sum_i MF(\boldsymbol{d}_i, \boldsymbol{F}^T \boldsymbol{S}\boldsymbol{w}_i)$$

  - Then apply the Explicit Matrix Factorization algorithm (Li et.al., 2015)

Joint Matrix Factorization

Theorem 3. *For a node i in the network, we denote $Q_{i,c}$ as a pre-defined upper bound for the possible co-occurrence count between node i and context c. With the equivalence of Skip-Gram Negative Sampling (SGNS) and Explicit Matrix Factorization (EMF) [9], the representation loss $MF(\cdot,\cdot)$ can be defined as the negative log probability of observing the structure vector $\boldsymbol{d_i}$ given i and $\boldsymbol{F}^T\boldsymbol{S}$ when $Q_{i,c}$ is set to $k\frac{\#(i)\#(c)}{|D|}+\#(i,c)$. To be more concrete,*

$$MF(\boldsymbol{d}_i, \boldsymbol{F}^T\boldsymbol{S}\boldsymbol{w}_i) = -\sum_{c\in|V|}\log P(d_{i,c}|\boldsymbol{f}_c^T\boldsymbol{S}\boldsymbol{w}_i)$$

where $\boldsymbol{f}_c \in \mathbb{R}^{N_f}$ is the c-th column of the content matrix $\boldsymbol{F}$, i.e., the feature vector of node c, $\#(i,c)$ is the co-occurrence count between node i and c, $\#(i)=\sum_{c\in|V|}\#(i,c)$, $\#(c)=\sum_{i\in|V|}\#(i,c)$, $|D|=\sum_{i,c\in|V|}\#(i,c)$ and k is the negative sampling ratio.

Joint Matrix Factorization

Theorem 3. *For a node i in the network, we denote $Q_{i,c}$ as a pre-defined upper bound for the possible co-occurrence count between node i and context c. With the equivalence of Skip-Gram Negative Sampling (SGNS) and Explicit Matrix Factorization (EMF) [9], the representation loss $MF(\cdot,\cdot)$ can be defined as the negative log probability of observing the structure vector $\boldsymbol{d_i}$ given i and $\boldsymbol{F}^T\boldsymbol{S}$ when $Q_{i,c}$ is set to $k\frac{\#(i)\#(c)}{|D|}+\#(i,c)$. To be more concrete,*

$$MF(\boldsymbol{d}_i, \boldsymbol{F}^T\boldsymbol{S}\boldsymbol{w}_i) = -\sum_{c\in|V|}\log P(d_{i,c}|\boldsymbol{f}_c^T\boldsymbol{S}\boldsymbol{w}_i)$$

where $\boldsymbol{f}_c \in \mathbb{R}^{N_f}$ is the c-th column of the content matrix $\boldsymbol{F}$, i.e., the feature vector of node c, $\#(i,c)$ is the co-occurrence count between node i and c, $\#(i) = \sum_{c\in|V|}\#(i,c)$, $\#(c) = \sum_{i\in|V|}\#(i,c)$, $|D| = \sum_{i,c\in|V|}\#(i,c)$ and k is the negative sampling ratio.

Joint Matrix Factorization

- Based on which we can derive

$$MF(\boldsymbol{D}, \boldsymbol{F}^T \boldsymbol{S}\boldsymbol{W}) \triangleq \sum_{i=1}^{|V|} MF(\boldsymbol{d}_i, \boldsymbol{F}^T \boldsymbol{S}\boldsymbol{w}_i)$$
$$= -\sum_{i=1}^{|V|}\sum_{c=1}^{|V|} \log P(d_{i,c}|\boldsymbol{f}_c^T \boldsymbol{S}\boldsymbol{w}_i)$$

  - And the final loss function comes to

$$L(\boldsymbol{W}, \boldsymbol{S}) = MF(\boldsymbol{D}, \boldsymbol{F}^T \boldsymbol{S}\boldsymbol{W})$$
$$= -\sum_{i=1}^{|V|}\sum_{c=1}^{|V|} \log P(d_{i,c}|\boldsymbol{f}_c^T \boldsymbol{S}\boldsymbol{w}_i)$$

Optimization

- Alternating Minimization (ALM) is applied

Algorithm 3 ALM algorithm for generalized explicit matrix factorization

Input: Co-occurrence matrix $\boldsymbol{D}$, content matrix $\boldsymbol{F}$, ALM step-size μ and maximum number of outer iterations I

Output: Node embedding matrix $\boldsymbol{W}$, feature embedding dictionary $\boldsymbol{S}$

1: Initialize $\boldsymbol{W}$ and $\boldsymbol{S}$ randomly
2: **for** $i = 1$ to I **do**
3: **repeat**
4: $\boldsymbol{W} = \boldsymbol{W} - \mu \cdot \text{grad}_{\boldsymbol{W}}$
5: **until** Convergence
6: **repeat**
7: $\boldsymbol{S} = \boldsymbol{S} - \mu \cdot \text{grad}_{\boldsymbol{S}}$
8: **until** Convergence
9: **end for**

- where

$$\begin{aligned}
\frac{\partial L(\boldsymbol{W},\boldsymbol{S})}{\partial \boldsymbol{S}} &= \frac{\partial MF(\boldsymbol{D},\boldsymbol{F}^T\boldsymbol{S}\boldsymbol{W})}{\partial \boldsymbol{S}} \\
&= \sum_{i\in|V|} -\boldsymbol{d}_i\boldsymbol{w}_i^T + \mathbb{E}_{\boldsymbol{d}_i'|\boldsymbol{F}^T\boldsymbol{S}\boldsymbol{w}_i}[\boldsymbol{d}_i']\boldsymbol{w}_i^T \\
&= \boldsymbol{F}(\mathbb{E}_{\boldsymbol{D}'|\boldsymbol{F}^T\boldsymbol{S}\boldsymbol{W}}\boldsymbol{D}' - \boldsymbol{D})\boldsymbol{W}^T \\
&\triangleq \text{grad}_{\boldsymbol{S}} \\
\frac{\partial L(\boldsymbol{W},\boldsymbol{S})}{\partial \boldsymbol{W}} &= \boldsymbol{S}^T\boldsymbol{F}(\mathbb{E}_{\boldsymbol{D}'|\boldsymbol{F}^T\boldsymbol{S}\boldsymbol{W}}\boldsymbol{D}' - \boldsymbol{D}) \\
&\triangleq \text{grad}_{\boldsymbol{W}}
\end{aligned}$$

Outline

Experiments

- We test our method mainly on
  - Two tasks
    1. Semi-supervised node classification
    2. Link prediction
  - Four datasets

Table 1. Dataset statistics

Dataset	# Classes	# Nodes	# Edges	# Feature
Citeseer	6	3327	4732	3703
Cora	7	2708	5429	1433
Pubmed	3	19717	44338	500
Facebook	-	4309	88234	1283

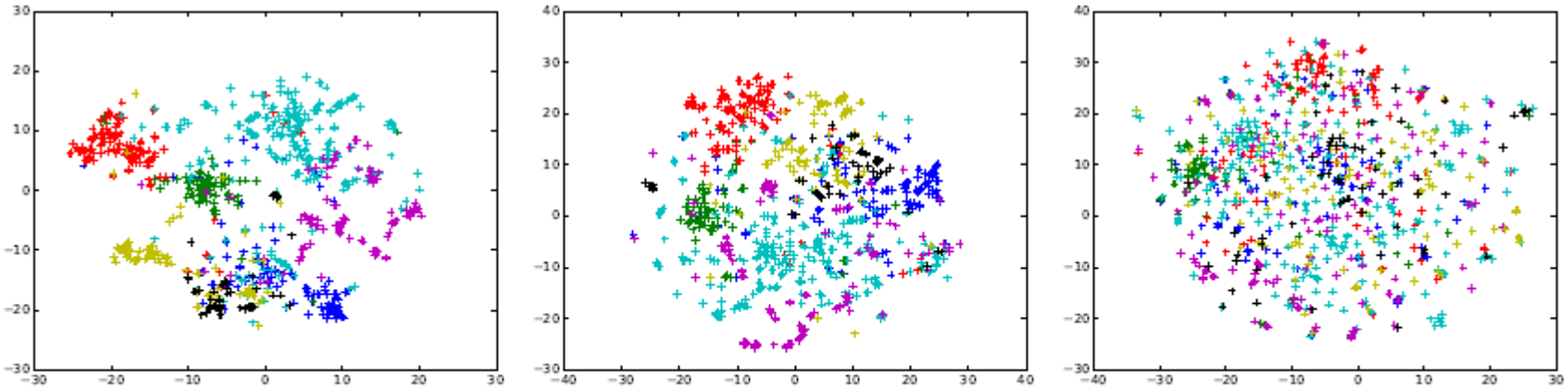
  - Eight baselines

Semi-Supervised Node Classification

Table 2. Accuracy of semi-supervised node classification (in percentage). Upper and lower rows correspond to unsupervised and semi-supervised embedding methods respectively.

Method	Citeseer	Cora	Pubmed
DeepWalk	41.5	67.3	66.4
node2vec	47.2	69.8	70.3
TADW	54.0	72.0	41.7
HSCA	47.7	65.4	41.7
APNE	**72.6**	**79.3**	**81.5**
DMF	65.5	58.5	59.3
LANE	60.3	65.2	-
Planetoid	67.3	73.4	76.7
GCN	70.3	**81.5**	79.0
APNE+label	**72.8**	79.6	**82.1**

Visualization



(a) *APNE* (b) node2vec (c) TADW

Fig. 2. t-SNE visualization of embeddings on Cora

Parameter Effect

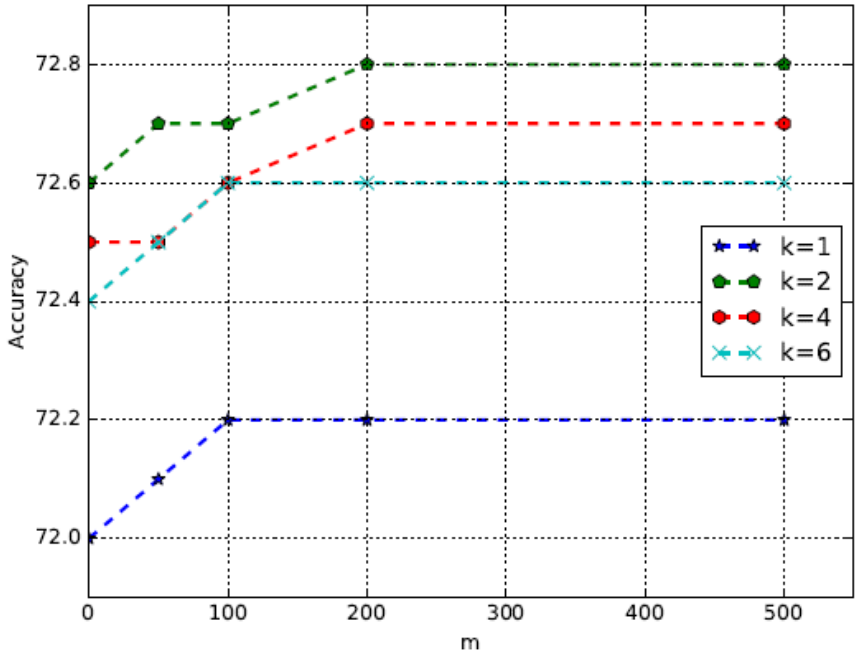


Fig. 3. Parameter effect of $APNE$ on the node classification task (in percentage)

Link Prediction

Table 3. Results of link prediction

Method	Citeseer		Cora		Pubmed		Facebook	
	AUC	MAP	AUC	MAP	AUC	MAP	AUC	MAP
Common Neighbor	0.567	0.781	0.616	0.797	0.561	0.778	0.797	0.882
Jaccard's Coefficient	0.567	0.782	0.616	0.795	0.561	0.776	0.797	0.877
Adamic Adar	0.560	0.780	0.617	0.801	0.561	0.778	0.798	0.885
Preferential Attachment	0.675	0.721	0.679	0.705	0.863	0.852	0.675	0.675
DeepWalk	0.656	0.725	0.734	0.793	0.721	0.781	0.891	0.914
node2vec	0.502	0.731	0.723	0.790	0.728	0.785	0.888	0.911
TADW	0.914	0.936	0.854	0.878	0.592	0.620	0.909	0.921
HSCA	0.905	0.928	0.861	0.885	0.632	0.660	0.926	0.917
APNE	**0.938**	**0.940**	**0.909**	**0.910**	**0.925**	**0.916**	**0.956**	**0.949**

Conclusion

- We learn a generalized network embedding which preserves structure, content and label information simultaneously
- Achieves SOTA among unsupervised methods
- Matlab code: https://github.com/lemmonation/APNE

Future work

- Finding a good approximation to reduce the time complexity
- Exploring the theoretical foundation of the good result, especially from the optimization perspective

Q & A

Thanks!