BiNE: Bipartite Network Embedding

ACM SIGIR 2018, July 8, Ann Arbor Michigan, U.S.A.

MingGao[*], Leihui Chen[*], Xiangnan He[+], Aoying Zhou[*]

[*]East China Normal University

[+]National University of Singapore

Background

☐ Network

- A ubiquitous data structure to model the relationships between entities

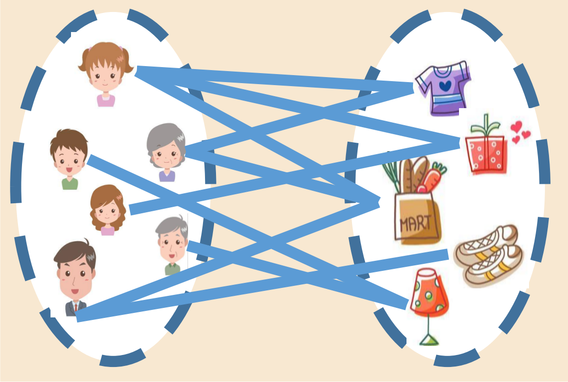
☐ Network embedding

- Crucial to obtain the representations for vertices
- Helpful to many applications, such as **vertex labeling**, **link prediction**, **recommendation**, and **clustering**, etc.

Homogeneous Network

- ✓ Social network
- ✓ Collaboration network
- ✓ Transportation network
- ✓ …

Heterogeneous Network

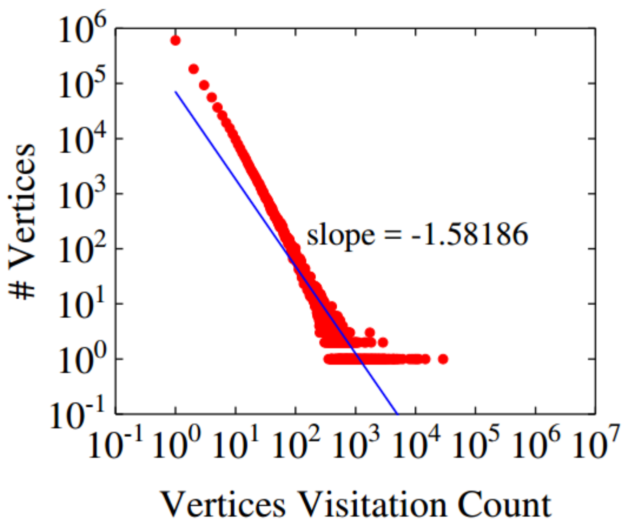


- ✓ Item adoption
- ✓ Web visiting
- ✓ Question answering
- ✓ …

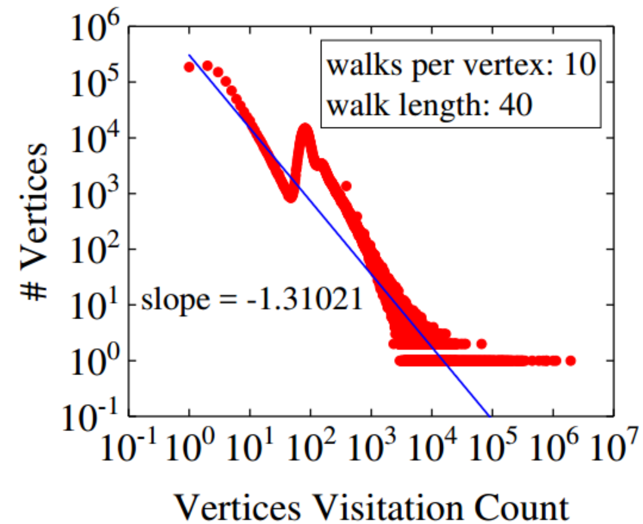
Drawbacks of Existing Works for Bipartite Networks

Homogeneous network embedding:

- Ignore **type information** of vertices (e.g., Node2vec, DeepWalk, etc.)
- Ignore key characteristic of bipartite network -- **power-law distribution** of vertex degrees



(a) YouTube



(b) Random walk generator

Heterogeneous network embedding:

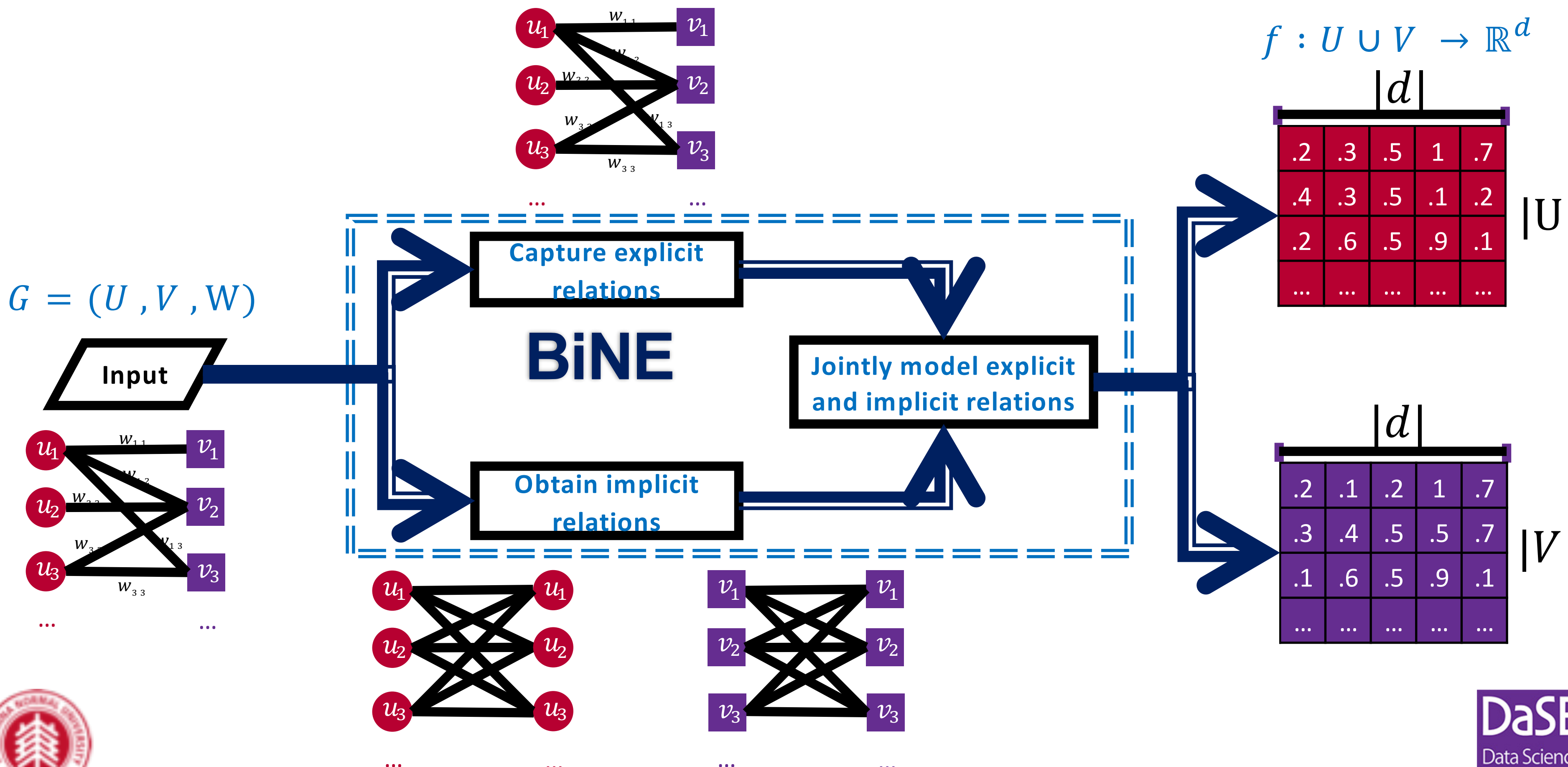
- MetaPath2vec [Dong et al, KDD'17] treats **explicit** and **implicit** relations as contributing **equally**

Outline

- ☐ Background & Motivations
- ☐ Proposed Method
- ☐ Experiments and Results
- ☐ Conclusions

BiNE: Bipartite Network Embedding

- Two Characteristics of BiNE
  - Modeling the **explicit and implicit relations** simultaneously
  - **A biased and self-adaptive random walk generator**



Modeling Explicit Relations (Observed links)

Original network space

The joint probability between vertices u_i and v_j is **defined** as:

$$P(i,j) = \frac{w_{ij}}{\sum_{e_{ij} \in E} w_{ij}}$$

Embedding space

The joint probability between vertices u_i and v_j is **estimated** as:

$$\hat{P}(i,j) = \frac{1}{1 + exp(-\vec{\mathbf{u}}_i{}^T \vec{\mathbf{v}}_j)}$$

Preserving the local proximity

Minimizing the difference (KL-divergence) between the two distributions:

$$minimize \quad O_1 = KL(P||\hat{P}) = \sum_{e_{ij} \in E} P(i,j) \log\left(\frac{P(i,j)}{\hat{P}(i,j)}\right)$$

$$\propto - \sum_{e_{ij} \in E} w_{ij} \log \hat{P}(i,j).$$

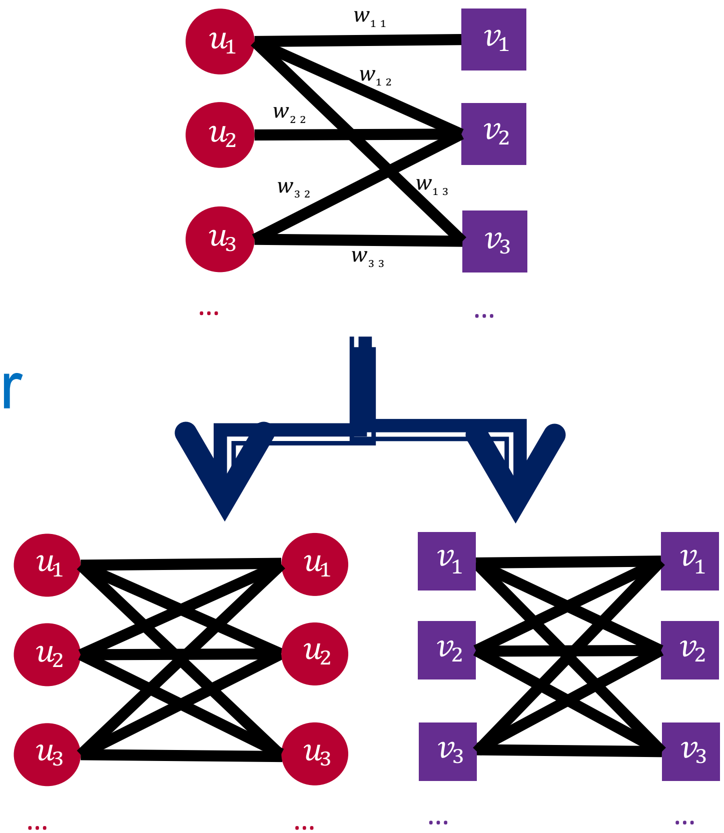
DaSE Data Science & Engineering

Modeling Implicit Relations (High-order relations)

- **Constructing Corpus of Vertex Sequences**
  - Construct **U-U** and **V-V** networks

$$w_{ij}^{U} = \sum_{k \in V} w_{ik} w_{jk}; \quad w_{ij}^{V} = \sum_{k \in U} w_{ki} w_{kj}$$

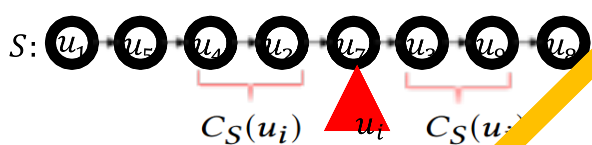
  - Run **Self-adaptive** random walker
    1) # of walks starting from a vertex depends on its centrality score.
    2) Length of a vertex sequence is controlled by a stop probability.
- **Optimizing a point-wise classification loss to capture the high-order correlations**



Capturing the High-order Relations

- **Assumption:** vertices **frequently co-occurred** in the same context of a sequence should be assigned to **similar embeddings**.

A. Taking corpus of users D^U as example , given a sequence S, *ws(=2) and a vertex* u_i*:*



B. $$maximize\ O_2 = \prod_{u_i \in S \wedge S \in D^U} \prod_{u_c \in C_S(u_i)} P(u_c|u_i).$$

C. $$maximize\ O_3 = \prod_{v_j \in S \wedge S \in D^V} \prod_{v_c \in C_S(v_j)} P(v_c|v_j).$$

$$P(u_c|u_i) = \frac{\exp(\vec{\mathbf{u}}_i{}^T \vec{\boldsymbol{\theta}}_c)}{\sum_{k=1}^{|U|} \exp(\vec{\mathbf{u}}_i{}^T \vec{\boldsymbol{\theta}}_k)}$$

$$P(v_c|v_j) = \frac{\exp(\vec{\mathbf{v}}_j{}^T \vec{\boldsymbol{\vartheta}}_c)}{\sum_{k=1}^{|V|} \exp(\vec{\mathbf{v}}_j{}^T \vec{\boldsymbol{\vartheta}}_k)}$$

Sample High-quality and Diverse Negatives with Locality Sensitive Hashing (LSH)

Joint Optimization

□ A joint optimization framework

foreach *edge* $(u_i, v_j) \in E$ **do**

- Update $\vec{\mathbf{u}}_i$ and $\vec{\mathbf{v}}_j$ using Equations (10) and (11);
- **foreach** (u_i, u_c) *in the sequence* $S \in D^U$ **do**
  - Negative sampling to generate $N_S^{ns}(u_i)$;
  - Update $\vec{\mathbf{u}}_i$ using Equation (12);
  - Update $\vec{\boldsymbol{\theta}}_z$ using Equation (14) where $z \in \{u_c\} \cup N_S^{ns}(u_i)$;
- **foreach** (v_j, v_c) *in the sequence* $S \in D^V$ **do**
  - Negative sampling to generate $N_S^{ns}(v_j)$;
  - Update $\vec{\mathbf{v}}_j$ using Equation (13);
  - Update $\vec{\boldsymbol{\vartheta}}_z$ using Equation (15) where $z \in \{v_c\} \cup N_S^{ns}(v_j)$;

$$maximize\ L = \boxed{\alpha \log O_2 + \beta \log O_3} - \boxed{\gamma O_1}$$

Explicit relations

Implicit relations

Outline

- Background & Motivations
- Proposed Method
- Experiments and Results
- Conclusions

Experimental Setting-up

◻ Tasks

- ➢ Two tasks: **link prediction (classification)** & **recommendation (ranking)**

◻ Datasets and Metrics

Task	Link Prediction		Recommendation		
Type	undirected, unweighted		undirected, weighted		
Metric	AUC-ROC,AUC-PR		F1, NDCG, MAP, MRR		
Name	Tencent	Wikipedia	VisualizeUs	DBLP	MovieLens
$\|U\|$	14,259	15,000	6,000	6,001	69,878
$\|V\|$	1,149	3,214	3,355	1,308	10,677
$\|E\|$	196,290	172,426	35,639	29,256	10,000,054
Density	1.2%	0.4%	0.2%	0.4%	1.3%

◻ Research Questions

- ➢ **RQ1** Performance of BiNE compared to representative baselines
- ➢ **RQ2** Is the implicit relations helpful?
- ➢ **RQ3** Effect of random walk generator

Baselines

- **Network embedding methods**
  - DeepWalk [*Perozzi et al KDD 2014*]
  - LINE [*Tang et al WWW 2015*]
  - Node2vec [*Grover et al KDD 2016*]
  - Metapath2vec++ [*Dong et al KDD 2017*]
- **Link Prediction methods** [***Xia et al ASONAM 2012***]
  - JC (Jaccard coefficient)
  - AA (Adamic/Adar)
  - Katz (Katz index)
  - PA (Preferential attachment)
- **Recommendation methods**
  - BPR [Rendle et al UAI 2009]
  - RankALS [Takács et al Recsys 2012]
  - FISMauc [Kabbur et al KDD 2013]

RQ1: Performance of Link Prediction

Table 3: Link prediction performance on Tencent and Wikipedia.

Algorithm	Tencent		Wikipedia	
	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR
JC	51.49%	66.18%	63.90%	73.04%
AA	50.63%	65.66%	87.37%	91.12%
Katz	50.90%	65.06%	90.84%	92.42%
PA	55.60%	68.99%	90.71%	93.37%
DeepWalk	57.62%	71.32%	89.71%	91.20%
LINE	59.68%	73.48%	91.62%	93.28%
Node2vec	59.28%	72.62%	89.93%	91.23%
Metapath2vec++	60.70%	73.69%	89.56%	91.72%
BiNE	**60.98%****	**73.77%****	**92.91%****	**94.45%****

** indicates that the improvements are statistically significant for $p < 0.01$ judged by paired t-test.

Observations:

1. **Data-dependent supervised manner** is more advantageous.
2. Positive effect of modeling **both explicit and implicit relations** into the embedding process.
3. Effectiveness of modeling the explicit and implicit relations **in different ways.**

RQ2: Performance of Recommendation

Table 4: Performance comparison of Top-10 Recommendation on VisualizeUs, DBLP, and MovieLens.

Algorithm	VisualizeUs				DBLP				Movielens			
	F1@10	NDCG@10	MAP@10	MRR@10	F1@10	NDCG@10	MAP@10	MRR@10	F1@10	NDCG@10	MAP@10	MRR@10
BPR	6.22%	9.52%	5.51%	13.71%	8.95%	18.38%	13.55%	22.25%	8.03%	7.58%	2.23%	40.81%
RankALS	2.72%	3.29%	1.50%	3.81%	7.62%	11.50%	7.52%	14.87%	8.48%	7.95%	2.66%	38.93%
FISMauc	10.25%	15.46%	8.86%	16.67%	9.81%	13.77%	7.38%	14.51%	6.77%	6.13%	1.63%	34.04%
DeepWalk	5.82%	8.83%	4.28%	12.12%	8.50%	24.14%	19.71%	31.53%	3.73%	3.21%	0.90%	15.40%
LINE	9.62%	13.76%	7.81%	14.99%	8.99%	14.41%	9.62%	17.13%	6.91%	6.50%	1.74%	38.12%
Node2vec	6.73%	9.71%	6.25%	13.95%	8.54%	23.89%	19.44%	31.11%	4.16%	3.68%	1.05%	18.33%
Metapath2vec++	5.92%	8.96%	5.35%	13.54%	8.65%	25.14%	19.06%	31.97%	4.65%	4.39%	1.91%	16.60%
BiNE	**13.63%****	**24.50%****	**16.46%****	**34.23%****	**11.37%****	**26.19%****	**20.47%****	**33.36%****	**9.14%****	**9.02%****	**3.01%****	**45.95%****

** indicates that the improvements are statistically significant for $p < 0.01$ judged by paired t-test.

Observations:

1. Positive effect of considering **information of weight**
2. Importance of focusing on the **higher-order proximities** among vertices
3. **Jointly training** is superior to **separately training + post-processing**

Utility of Implicit Relations (RQ2)

Table 5: BiNE with and without implicit relations.

	Without Implicit Relations		With Implicit Relations	
Link Prediction				
Dataset	**AUC-ROC**	**AUC-PR**	**AUC-ROC**	**AUC-PR**
Tencent	59.78%	73.05%	**60.98%****	**73.77%****
WikiPedia	91.47%	93.73%	**92.91%****	**94.45%****
Recommendation				
Dataset	**MAP@10**	**MRR@10**	**MAP@10**	**MRR@10**
VisualizeUS	7.91%	15.65%	**16.46%****	**34.23%****
DBLP	20.20%	32.95%	**20.47%****	**33.36%****
MovieLens	2.86%	43.98%	**3.01%****	**45.95%****

** indicates that the improvements are statistically significant for $p < 0.01$ judged by paired t-test.

Observation:

Modeling high-order implicit relations is effective to **complement** with explicit relation modeling.

Data Science & Engineering

Random Walk Generator (RQ3)

Table 6: BiNE with different random walk generators.

	Uniform Random Walk Generator		Biased and Self-adaptive Random Walk Generator	
Link Prediction				
Dataset	**AUC-ROC**	**AUC-PR**	**AUC-ROC**	**AUC-PR**
Tencent	59.75%	73.06%	**60.98%****	**73.77%****
WikiPedia	88.77%	91.91%	**92.91%****	**94.45%****
Recommendation				
Dataset	**MAP@10**	**MRR@10**	**MAP@10**	**MRR@10**
VisualizeUS	15.93%	33.66%	**16.46%****	**34.23%****
DBLP	11.79%	23.41%	**20.47%****	**33.66%****
MovieLens	2.91%	46.12%	**3.04%****	**46.20%****

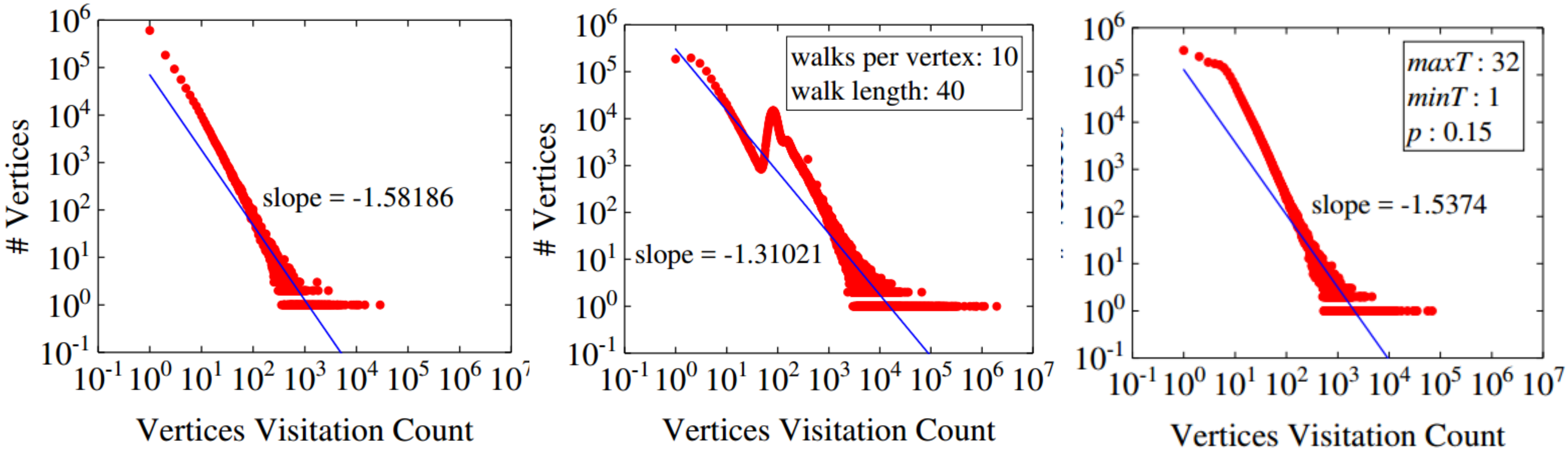
** indicates that the improvements are statistically significant for $p < 0.01$ judged by paired t-test.

Observation:

The **biased and self-adaptive** random walk generator contributes to learning better vertex embeddings.

Random Walk Generator (RQ3)

Distribution of vertex degree: DeepWalk Generator: Our Generator:



(a) YouTube (b) Random walk generator (c) Self-Adaptive generator

Observation:

The **biased and self-adaptive** random walk generator contributes to learning better vertex embeddings.

Case Study

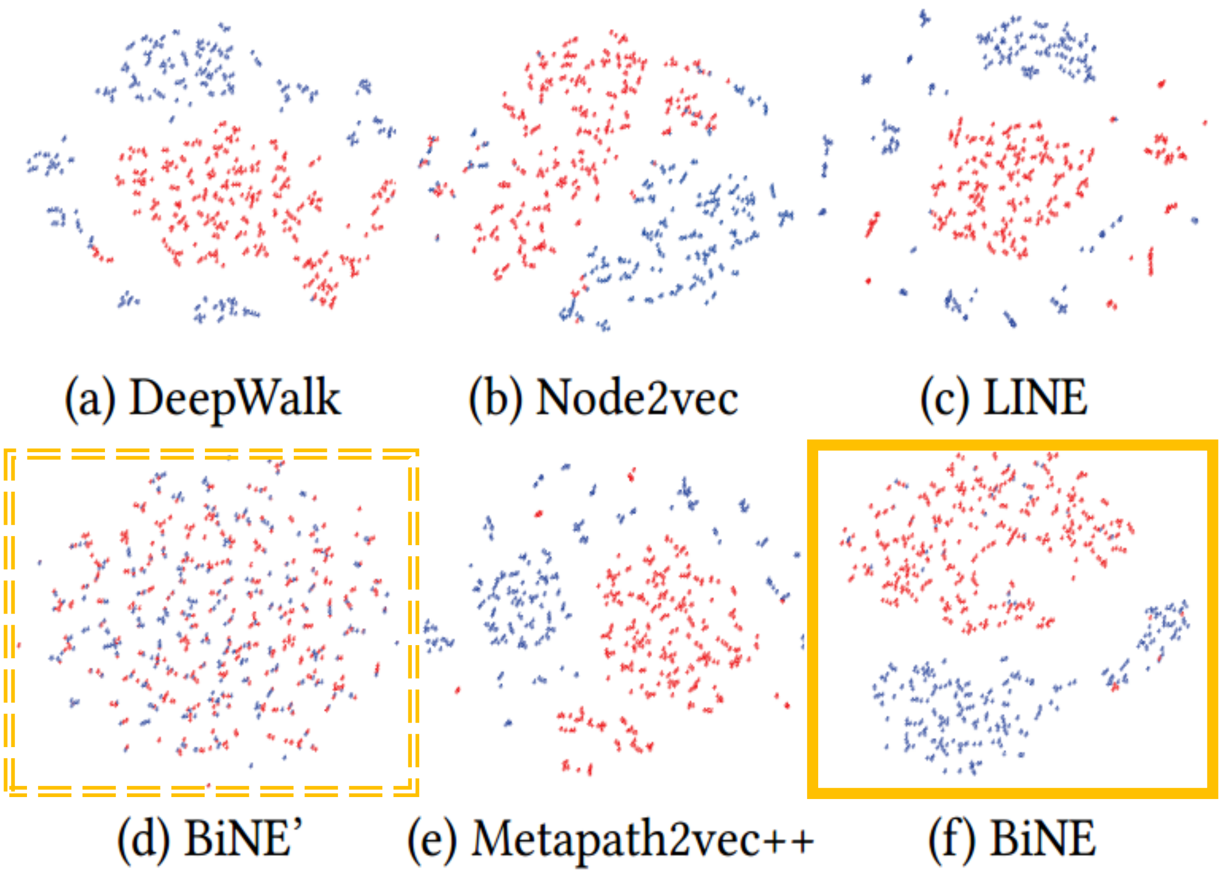


Figure 6: Visualization of authors in DBLP. Color of a vertex indicates the research fields of the authors (red: “computer science theory”, blue: “artificial intelligence”). BiNE’ is the version of BiNE – without implicit relations.

Conclusions

Conclusions

- Propose a dedicated approach for embedding bipartite networks
- Jointly model both the explicit relations and higher-order implicit relations
- Extensive experiments on several tasks of link prediction, recommendation, and visualization

Future work

- Extend our BiNE method to model auxiliary side info
- Investigate how to efficiently refresh embeddings for dynamic bipartite networks
- Network embedding + adversarial training

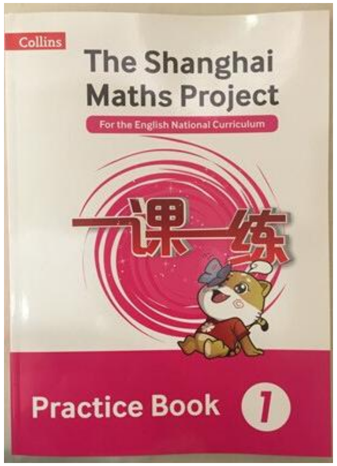
Acknowledgments

- Ming Gao (高明)
  (East China Normal University)

- Leihui Chen (陈雷慧)
  (East China Normal University)

- Aoying Zhou (周傲英)
  (East China Normal University)

- National Natural Science Foundation of China



- The Press of East China Normal University

- National Research Foundation, Prime Minister's Office, Singapore

Thank You for Your Attention

Code available:

Negative Sampling

- **Optimizing a point-wise classification loss**
  - $p(u_c|u_i)$ can be approximate as:

$$p(u_c, N_S^{ns}(u_i)|u_i) = \prod_{z \in \{u_c\} \cup N_S^{ns}(u_i)} P(z|u_i)$$

$$P(z|u_i) \quad = \quad \begin{cases} \sigma(\vec{\mathbf{u}}_i{}^T \vec{\boldsymbol{\theta}}_z), & \text{if } z \text{ is a context of } u_i \\ 1 - \sigma(\vec{\mathbf{u}}_i{}^T \vec{\boldsymbol{\theta}}_z), & \boxed{z \in N_S^{ns}(u_i)} \end{cases}$$

LSH-based

  - Following the similar formulations, we can get the counterparts for the conditional probability $p(z|v_i)$

DaSE Data Science & Engineering

LSH-based Negative Sampling

LSH-based negative sampling method

- For a center vertex u_i, high-quality negatives should be the vertices that are **dissimilar** from u_i

	Frequency-based or popularity-based sampling	LSH-based negative sampling
Strategy	High frequency objects	Dissimilar objects
Word Embedding	Useless words	
Network Embedding	Popular items or active users	

Experimental Results

☐ Performance of BiNE with different negative sampling strategies.

TABLE 8: BiNE with different negative sampling strategies.

	Frequency-based Negative Sampling		LSH-based Negative Sampling	
Link Prediction				
Dataset	**AUC-ROC**	**AUC-PR**	**AUC-ROC**	**AUC-PR**
Tencent	60.80%	73.64%	**60.98%**	**73.77%**
WikiPedia	92.21%	94.12%	**92.91%**	**94.45%**
Recommendation				
Dataset	**MAP@10**	**MRR@10**	**MAP@10**	**MRR@10**
VisualizeUS	9.10%	19.72%	**16.46%****	**32.93%****
DBLP	20.46%	32.93%	**20.47%**	**33.93%****
MovieLens	**3.01%**	45.86%	**3.01%**	**45.95%**

** indicates that the improvements are statistically significant for $p < 0.01$ judged by paired t-test.

Observations:

1. Two methods show roughly equivalent performance in most case.

2. However, there are situations (see VisualizeUS) in which LSH-based sampling method uses dissimilar information obtained from user behavior data can generate more reasonable negative samples