On Embedding Uncertain Graphs

Jiafeng Hu, Reynold Cheng, Zhipeng Huang, Yixiang Fang and Siqiang Luo

Department of Computer Science

The University of Hong Kong

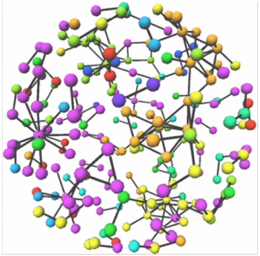
jhu@cs.hku.hk

November 7, 2017

Graphs are everywhere

Social Network

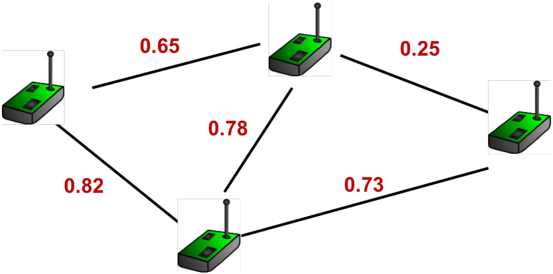
Collaboration Network



Protein-Protein interaction Network

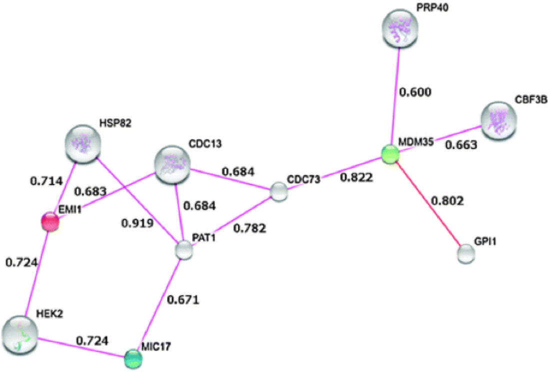
Uncertainty in Graph data

- Wireless sensor networks (WSNs)
  - Nodes: sensors
  - Edges: wireless connectivity between sensors
  - Uncertainties: probabilities of wireless connectivity



Uncertainty in Graph data

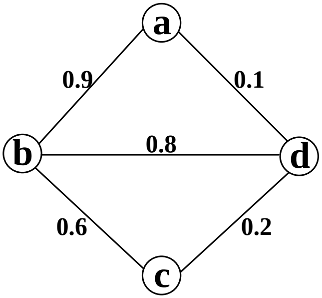
- Protein-Protein Interaction Networks (PPI)
  - Nodes: Proteins
  - Edges: an interaction between proteins
  - Uncertainties: probabilities of interactions between proteins derived from experimental evidence



Gabriele Cavallaro [Genome-wide analysis of eukaryotic twin CX_9C proteins]

Uncertainty in Graph data

Uncertain Graphs: each edge has an existence probability.



- Social Networks
- Traffic Networks
- Wireless Sensor Networks
- Protein-interaction Networks
- ...

Possible World Semantics (**PWS**)

- Given an uncertain graph $\mathcal{G}$, a **possible world** of $\mathcal{G}$ is a deterministic graph $G = (V, E_G \subseteq E)$. Assume the existence probabilities of edges are mutually **independent** [VLDB'10, KDD'10].

$$Pr[G] = \prod_{e \in E_G} \mathbf{P}_e \prod_{e \in E \setminus E_G} (1 - \mathbf{P}_e)$$

Possible World Semantics (**PWS**)

- Given an uncertain graph $\mathcal{G}$, a **possible world** of $\mathcal{G}$ is a deterministic graph $G = (V, E_G \subseteq E)$. Assume the existence probabilities of edges are mutually **independent** [VLDB'10, KDD'10].

$$Pr[G] = \prod_{e \in E_G} \mathbf{P}_e \prod_{e \in E \setminus E_G} (1 - \mathbf{P}_e)$$

(a) An uncertain graph $\mathcal{G}$ (b) A possible world of $\mathcal{G}$

$$\begin{aligned} Pr[G] &= \mathbf{P}_{ad}\mathbf{P}_{bd}\mathbf{P}_{bc}(1 - \mathbf{P}_{ab})(1 - \mathbf{P}_{cd}) \\ &= 0.1 \times 0.8 \times 0.6 \times 0.1 \times 0.8 = 0.00384. \end{aligned}$$

Uncertain Graph Mining

Tasks:

- Clustering [TKDE'13, ICDM'12]
- Classification [ICDM'09, SSDBM'14]
- k-NN queries [VLDB'10]
- $\cdots$

Uncertain Graph Mining

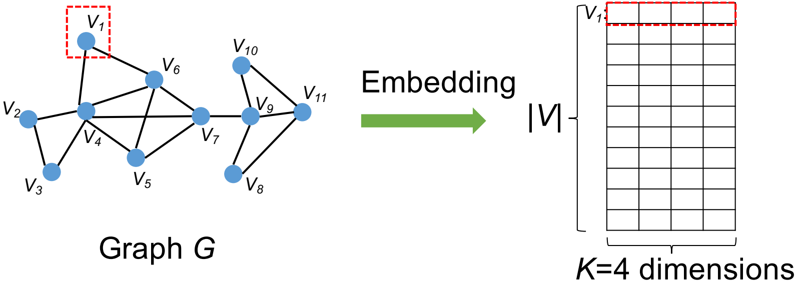
Tasks:

- Clustering [TKDE'13, ICDM'12]
- Classification [ICDM'09, SSDBM'14]
- k-NN queries [VLDB'10]
- $\cdots$

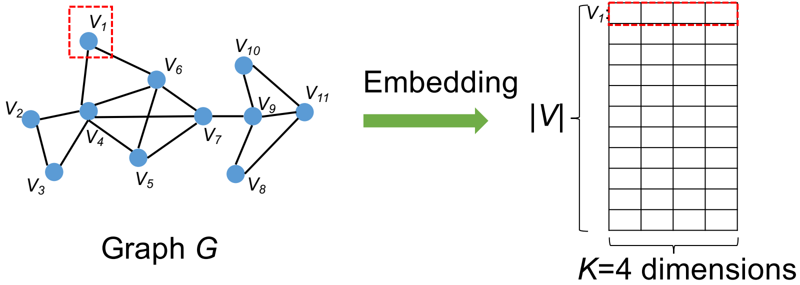
Shortcomings:

- High computational cost: expensive to compute similarities between nodes under PWS.
- Low adaptability: solutions are tailored for a particular mining task.

Graph Embedding

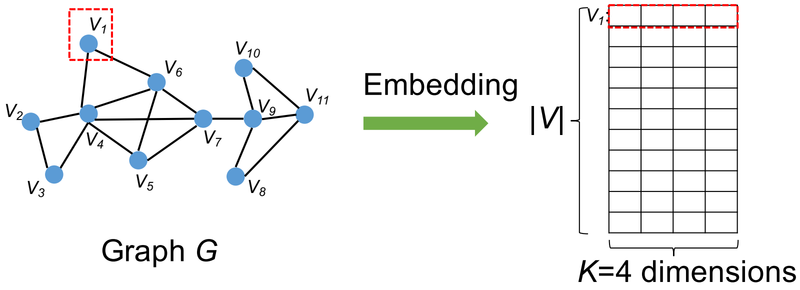


Graph Embedding



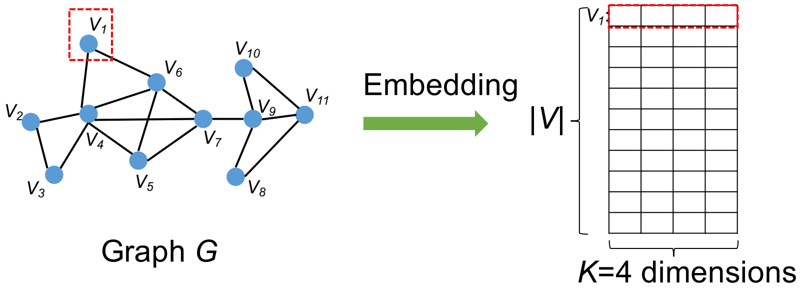
- Existing embedding solutions (for **deterministic** graphs): DeepWalk [KDD'14], LINE [WWW'15], node2vec [KDD'16], etc

Graph Embedding



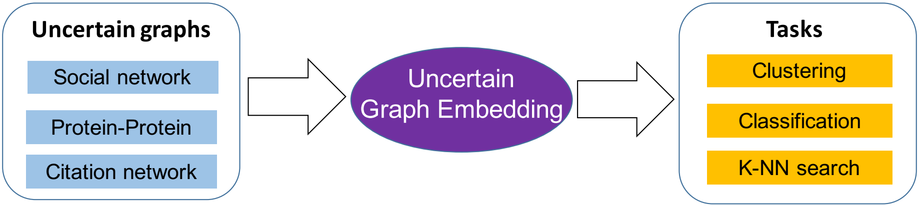
- Existing embedding solutions (for **deterministic** graphs): DeepWalk [KDD'14], LINE [WWW'15], node2vec [KDD'16], etc
- Not designed for uncertain graphs

Graph Embedding

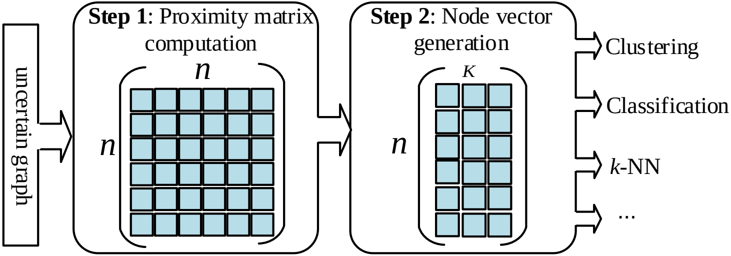


- Existing embedding solutions (for **deterministic** graphs): DeepWalk [KDD'14], LINE [WWW'15], node2vec [KDD'16], etc
- Not designed for uncertain graphs
- Simply remove the probabilities → poor performance

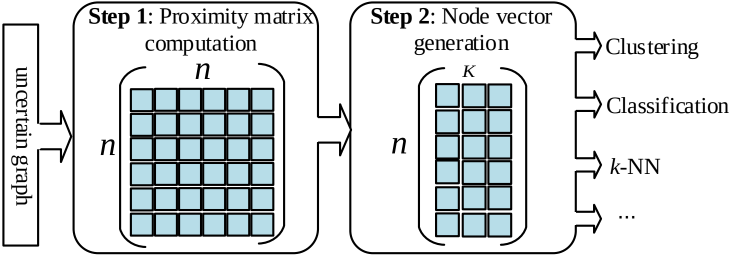
Uncertain Graph Embedding



URGE: UnceRtain Graph Embedding



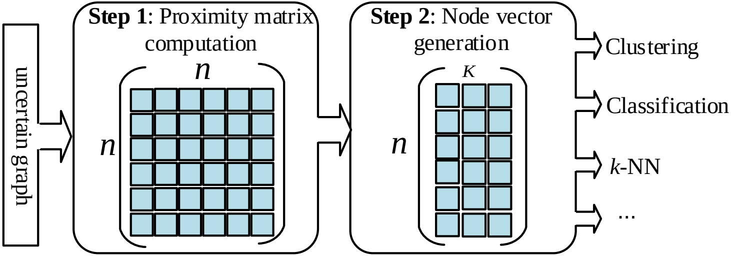
URGE: UnceRtain Graph Embedding



Step 1: Proximity matrix S

- (second order) Expected Jaccard Similarity (EJS)
- (high order) Probabilistic Random Walk with Restart (PRWR)

URGE: UnceRtain Graph Embedding

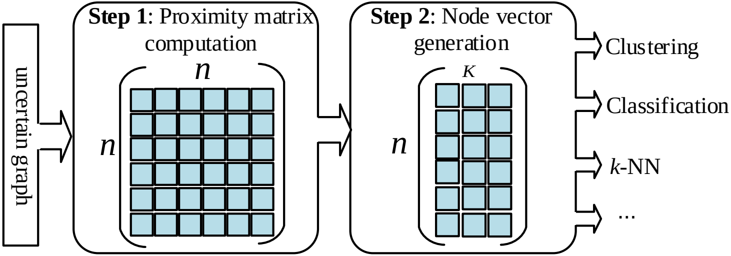


Step 2: Objective function

$$\min_{\mathbf{U},\tilde{\mathbf{U}}} \|\mathbf{S} - \mathbf{U}\tilde{\mathbf{U}}^T\|^2 + \frac{\lambda}{2}(\|\mathbf{U}\|^2 + \|\tilde{\mathbf{U}}\|^2)$$

- (Input) $\mathbf{S} \in \Re^{n\times n}$: proximity matrix;
- (Input) λ controls the weight of the regularization term.
- (Output) matrices $\mathbf{U} \in \Re^{n\times K}$ and $\tilde{\mathbf{U}} \in \Re^{n\times K}$

URGE: UnceRtain Graph Embedding



Step 2: Objective function

$$\min_{\mathbf{U},\tilde{\mathbf{U}}} \|\mathbf{S} - \mathbf{U}\tilde{\mathbf{U}}^T\|^2 + \frac{\lambda}{2}(\|\mathbf{U}\|^2 + \|\tilde{\mathbf{U}}\|^2)$$

Negative sampling + asynchronous stochastic gradient algorithm (ASGD)

How to compute the proximity matrix efficiently?

Second-order Proximity: Expected Jaccard Similarity

Jaccard similarity between node u and v on a deterministic graph G:

$$S_{uv}^{J} = \frac{|N_G(u) \cap N_G(v)|}{|N_G(u) \cup N_G(v)|}$$

$N_G(x)$: the neighbor set of node x on G.

Second-order Proximity: Expected Jaccard Similarity

Jaccard similarity between node u and v on a deterministic graph G:

$$S_{uv}^{J} = \frac{|N_G(u) \cap N_G(v)|}{|N_G(u) \cup N_G(v)|}$$

$N_G(x)$: the neighbor set of node x on G.

Expected Jaccard Similarity (EJS):

$$S_{uv}^{\mathrm{EJS}} = \sum_{G \in \Omega(\mathcal{G})} (S_{uv}^{\mathrm{J}})_G Pr[G]$$

$\Omega(\mathcal{G})$: the set of all possible worlds of $\mathcal{G}$.

Computation of EJS

Lemma (A. Stuart, 1998; Z. Zhou, ICDM'13)

Given two nodes u *and* v *of* $\mathcal{G}$, *let* $X_{uv} = |N_G(u) \cap N_G(v)|$ *and* $Y_{uv} = |N_G(u) \cup N_G(v)|$, *where* G *is a possible world of* $\mathcal{G}$. *Then,*

$$\mathbf{S}_{uv}^{\mathrm{EJS}} = E\left[\frac{X_{uv}}{Y_{uv}}\right] \approx \frac{E[X_{uv}]}{E[Y_{uv}]} - \frac{\mathit{Cov}(X_{uv}, Y_{uv})}{E[Y_{uv}]^2} + \frac{E[X_{uv}]\mathit{Var}(Y_{uv})}{E[Y_{uv}]^3}$$

- Z. Zou et al. study the problem of computing EJS between a pair of nodes $\rightarrow \mathcal{O}(d)$

- Z. Zou et al. study the problem of computing EJS between a pair of nodes $\rightarrow \mathcal{O}(d)$
- (Basic method) for each node u on $\mathcal{G}$, enumerate all 2-hop neighbors, v, and compute $\mathbf{S}_{uv}^{\mathtt{EJS}} \rightarrow \mathcal{O}(nd^2 * d)$

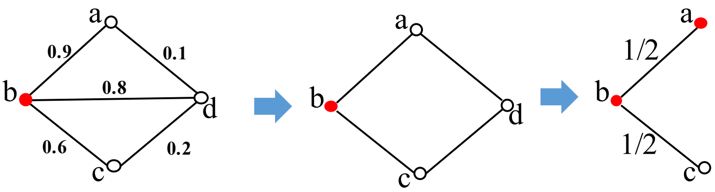
- Z. Zou et al. study the problem of computing EJS between a pair of nodes $\rightarrow \mathcal{O}(d)$
- (Basic method) for each node u on $\mathcal{G}$, enumerate all 2-hop neighbors, v, and compute $\mathbf{S}^{\mathrm{EJS}}_{uv} \rightarrow \mathcal{O}(nd^2 * d)$
- (**Our solution**) compute the EJS for all pair of nodes incrementally (i.e., the whole EJS matrix $\mathbf{S}^{\mathrm{EJS}}$) $\rightarrow \mathcal{O}(nd^2)$

High-order Proximity: Probabilistic Random Walk with Restart

- Random walk (transition procedure) on uncertain graphs (for node u) [VLDB'10]:
  1. generate a possible world G for u;
  2. walk to a neighbor uniformly at random if there exists any neighbors of u in G, otherwise stay at u.

High-order Proximity: Probabilistic Random Walk with Restart

- Random walk (transition procedure) on uncertain graphs (for node u) [VLDB'10]:
  1. generate a possible world G for u;
  2. walk to a neighbor uniformly at random if there exists any neighbors of u in G, otherwise stay at u.



High-order Proximity: Probabilistic Random Walk with Restart

- Random walk (transition procedure) on uncertain graphs (for node u) [VLDB'10]:
  1. generate a possible world G for u;
  2. walk to a neighbor uniformly at random if there exists any neighbors of u in G, otherwise stay at u.
- **Probabilistic Transition Matrix, PTM** [VLDB'10, IS'15]:

$$\mathbf{W}_{uv} = \begin{cases} \prod_{(u,q)\in E}(1-\mathbf{P}_{uq}), & u = v \\ \sum_{G\in\Omega(\mathcal{G})\wedge(u,v)\in E_G} \frac{1}{d_u^G} Pr[G], & u \neq v \end{cases}$$

(E_G: edge set of G; d_u^G: out-degree of node u in G)

High-order Proximity: Probabilistic Random Walk with Restart

- Random walk (transition procedure) on uncertain graphs (for node u) [VLDB'10]:
  1. generate a possible world G for u;
  2. walk to a neighbor uniformly at random if there exists any neighbors of u in G, otherwise stay at u.

- **Probabilistic Transition Matrix, PTM** [VLDB'10, IS'15]:

$$\mathbf{W}_{uv} = \begin{cases} \prod_{(u,q)\in E}(1-\mathbf{P}_{uq}), & u = v \\ \sum_{G\in\Omega(\mathcal{G})\wedge(u,v)\in E_G} \frac{1}{d_u^G} Pr[G], & u \neq v \end{cases}$$

(E_G: edge set of G; d_u^G: out-degree of node u in G)

- **Probabilistic Random Walk with Restart, PRWR**

$$\mathbf{S}^{\text{PRWR}} = (1-\alpha)\mathbf{S}^{\text{PRWR}}\mathbf{W} + \alpha\mathbf{I}, \quad (\mathbf{I} \text{ is an identity matrix})$$

Computation of PTM and PRWR

Computation of PTM:

- (Basic method) an existing algorithm $\rightarrow \mathcal{O}(nd^3)$ [IS'15]
- (Our method) further improvement, a hash-based method $\rightarrow \mathcal{O}(hnd^2)$, $h \ll d$.

Computation of PRWR:

- Monte Carlo method $\rightarrow \mathcal{O}(nR\frac{1}{\alpha})$
  - R: number of walkers;
  - $1/\alpha$: expected length of random paths

Experiments

Tasks: node clustering, node classification and k-NN search

Algorithms:

- Our algorithms:
  - URGE$_{\texttt{EJS}}$: URGE algorithm based on EJS
  - URGE$_{\texttt{PRWR}}$: URGE algorithm based on PRWR
- Existing embedding algorithms:
  - DeepWalk
  - LINE
  - node2vec$_p^q$ (node2vec$_{0.25}^{0.25}$ and node2vec$_4^1$)
- Existing non-embedding algorithms:
  - MCL (for deterministic graph clustering)
  - pKwikCluster (for uncertain graph clustering)
  - uBayes$^+$ (for uncertain graph classification)
  - MostProbPath (for uncertain graph k-NN)

Task 1: Clustering

Dataset: 4 real uncertain Protein-Protein Interaction (PPI) networks[1]
Ground truth: the complex-memberships lists from the MIPS database

Name	#Nodes	#Edges	Avg. Prob.
Krogan_core	2,708	7,123	0.68
Krogan_extend	3,672	14,317	0.42
Collins	1,622	9,074	0.78
Gavin	1,855	7,669	0.36

Table : Statistics of the PPI networks.

[1] `http://www.nature.com/nmeth/journal/v9/n5/full/nmeth.1938.html`

Task 1: Clustering (Cont'd)

- Metric: F1 score based on the confusion matrix (true positive, false positive, true negative and false negative)
- Hierarchical clustering in vector space (embedding-based algorithms)

Task 1: Clustering (Cont'd)

- Metric: F1 score based on the confusion matrix (true positive, false positive, true negative and false negative)
- Hierarchical clustering in vector space (embedding-based algorithms)

Algorithm	Krogan_core	Krogan_extend	Collins	Gavin
DeepWalk	39.21	33.43	55.15	47.33
LINE	38.73	33.07	48.28	44.14
node2vec$_{4}^{1}$	39.30	33.06	52.42	46.17
node2vec$_{0.25}^{0.25}$	38.96	33.75	53.23	46.13
MCL	36.01	30.83	57.55	47.84
pKwikCluster	16.94	12.88	24.59	5.65
URGE$_{\text{EJS}}$	38.39	30.08	55.61	**54.54**
URGE$_{\text{PRWR}}$	**44.86**	**35.58**	**58.16**	52.59

Table : F1 scores (%) for clustering tasks.

Task 1: Clustering (Cont'd)

- Metric: F1 score based on the confusion matrix (true positive, false positive, true negative and false negative)
- Hierarchical clustering in vector space (embedding-based algorithms)

Algorithm	Krogan_core	Krogan_extend	Collins	Gavin
DeepWalk	39.21	33.43	55.15	47.33
LINE	38.73	33.07	48.28	44.14
node2vec$_4^1$	39.30	33.06	52.42	46.17
node2vec$_{0.25}^{0.25}$	38.96	33.75	53.23	46.13
MCL	36.01	30.83	57.55	47.84
pKwikCluster	16.94	12.88	24.59	5.65
URGE$_{\mathrm{EJS}}$	38.39	30.08	55.61	**54.54**
URGE$_{\mathrm{PRWR}}$	**44.86**	**35.58**	**58.16**	52.59

Table : F1 scores (%) for clustering tasks.

Task 2: Classification

Dataset:

- DBLP (co-authorship network): 45,583 edges, 14,376 papers, 4 classes
- Cora (citation network): 8,365 edges and 2,708 papers, 7 classes

*Use the method proposed by P. Boldi et al. [VLDB'12] to do **obfuscation**.

Other setting:

- k nearest neighbor classifiers (embedding-based algorithms).
- Varying training ratio (T_R) from 20% to 80%
- Metric: Micro-F1, Macro-F1.

Task 2: Classification (Cont'd)

Metric	Algorithm	20%	40%	60%	80%
Micro-F1(%)	DeepWalk	40.81	49.71	54.33	57.27
	LINE	40.87	47.96	53.26	56.52
	node2vec_4^1	41.25	51.29	55.67	59.27
	$\text{node2vec}_{0.25}^{0.25}$	40.16	49.33	53.53	56.98
	uBayes^+	32.43	45.13	44.57	57.44
	URGE_{EJS}	**58.00**	**63.31**	**66.36**	**69.45**
	$\text{URGE}_{\text{PRWR}}$	52.16	58.08	61.12	63.97
Macro-F1(%)	DeepWalk	38.12	48.22	52.95	55.98
	LINE	39.02	46.37	51.70	55.08
	node2vec_4^1	39.42	49.21	53.93	57.69
	$\text{node2vec}_{0.25}^{0.25}$	38.11	47.64	51.93	55.37
	uBayes^+	31.07	42.02	45.03	55.33
	URGE_{EJS}	**55.48**	**61.45**	**64.49**	**67.50**
	$\text{URGE}_{\text{PRWR}}$	49.86	56.41	59.64	62.42

Table : Results of classification on DBLP under different training ratio(%).

Task 2: Classification (Cont'd)

Metric	Algorithm	20%	40%	60%	80%
Micro-F1(%)	DeepWalk	40.81	49.71	54.33	57.27
	LINE	40.87	47.96	53.26	56.52
	node2vec$_4^1$	41.25	51.29	55.67	59.27
	node2vec$_{0.25}^{0.25}$	40.16	49.33	53.53	56.98
	uBayes$^+$	32.43	45.13	44.57	57.44
	URGE$_{\text{EJS}}$	**58.00**	**63.31**	**66.36**	**69.45**
	URGE$_{\text{PRWR}}$	52.16	58.08	61.12	63.97
Macro-F1(%)	DeepWalk	38.12	48.22	52.95	55.98
	LINE	39.02	46.37	51.70	55.08
	node2vec$_4^1$	39.42	49.21	53.93	57.69
	node2vec$_{0.25}^{0.25}$	38.11	47.64	51.93	55.37
	uBayes$^+$	31.07	42.02	45.03	55.33
	URGE$_{\text{EJS}}$	**55.48**	**61.45**	**64.49**	**67.50**
	URGE$_{\text{PRWR}}$	49.86	56.41	59.64	62.42

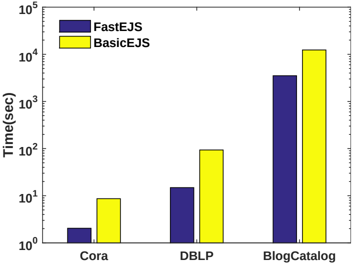
Table : Results of classification on DBLP under different training ratio(%).

Efficiency

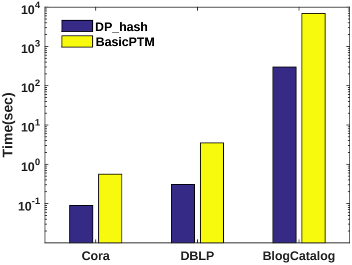
Datasets: DBLP, Cora, and BlogCatalog (relationships of the bloggers, 10**K** nodes and 665**K** edges)

Algorithms:

- EJS: `BasicEJS` vs `FastEJS` (**6+ times faster**)
- PTM: `BasicPTM` vs `DP_hash` (**20+ times faster**)



(c) EJS



(d) PTM

Conclusion

- Formulate the problem of uncertain graph embedding.
- Propose URGE, a proximity preserved embedding method for uncertain graphs.
- Develop efficient algorithms for two kinds of proximities (EJS and PRWR).
- Detailed evaluation on various tasks demonstrates the efficiency and effectiveness of the URGE solution.

Our team

THE UNIVERSITY OF HONG KONG
DEPARTMENT OF
COMPUTER SCIENCE

Dr. Reynold Cheng (ckcheng@cs.hku.hk)

Jiafeng

Zhipeng

Yixiang

Siqiang

Thanks!

Q&A