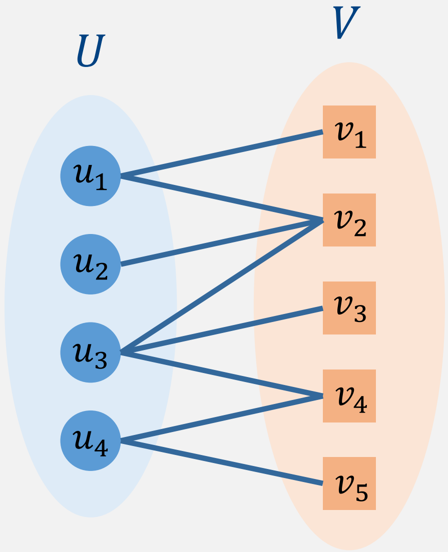
ACM SIGMOD PODS 2022
Philadelphia, PA, USA

Scalable and Effective Bipartite Network Embedding

Renchi Yang, Jieming Shi, Keke Huang, Xiaokui Xiao

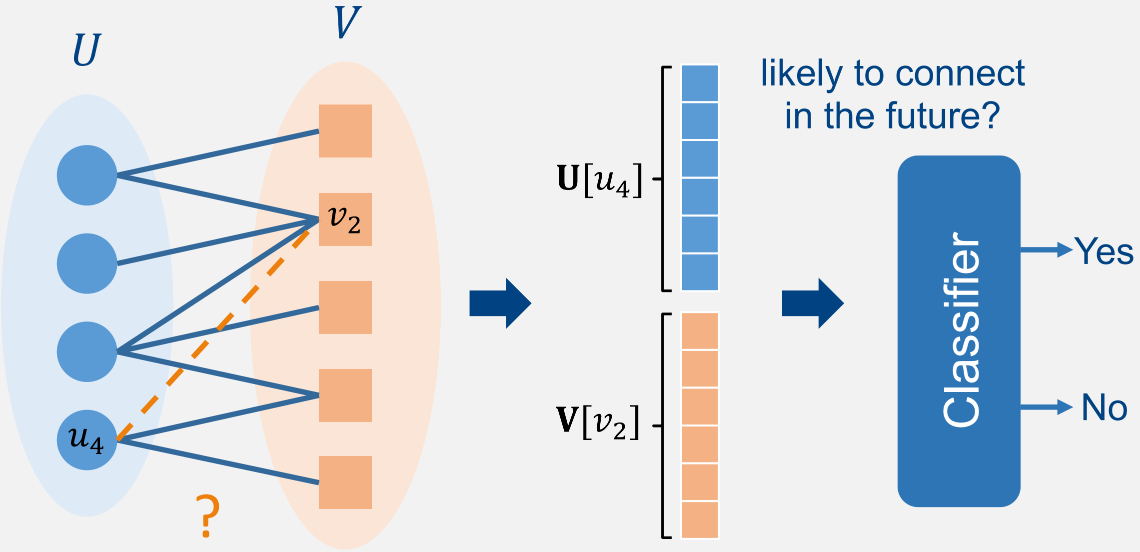
Bipartite Network Embedding (BNE)

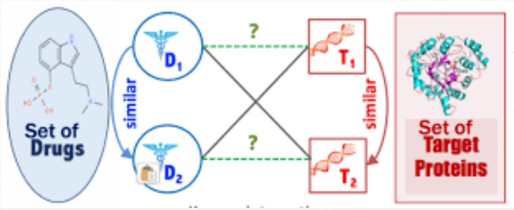
- Input: A bipartite graph $G = (U, V, E)$
  - 2 disjoint node sets U and V
  - the inter-set edges between nodes in U and V
- Output: For every node $u_i \in U$ and $v_j \in V$
  - A length-k embedding vector $\mathbf{U}[u_i]$
  - A length-k embedding vector $\mathbf{V}[v_j]$
  - $\mathbf{U}[u_i]$ and $\mathbf{V}[v_j]$ capture the hidden topological features surrounding u_i and v_j



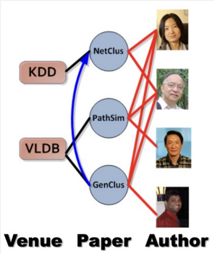
Applications

- Link Prediction





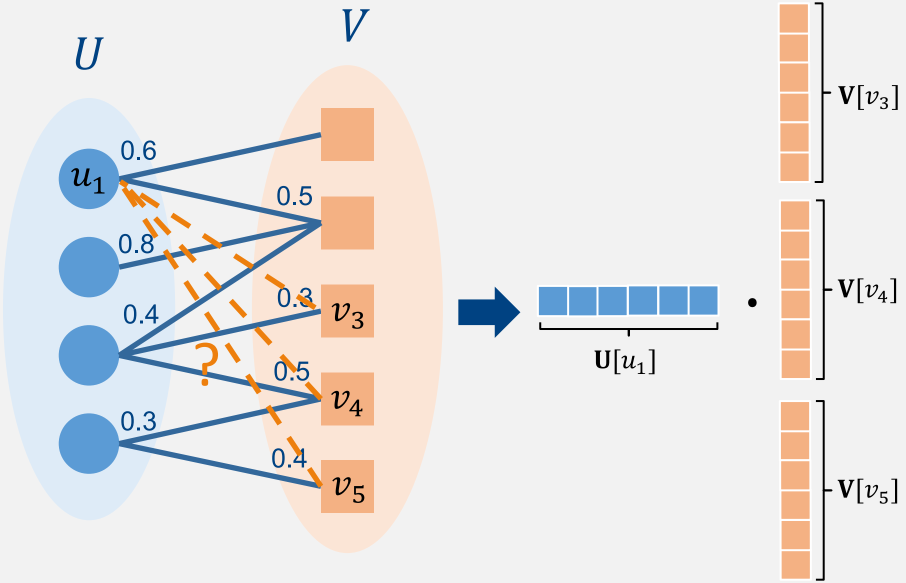
drug target prediction



collaboration prediction

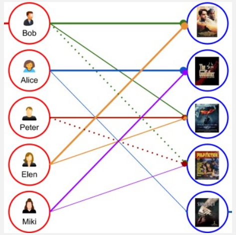
Applications

Recommendation

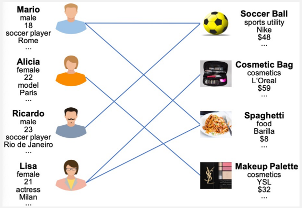


Top-2 recommendation

	score
v_5	**0.4**
v_3	**0.3**
v_4	0.2



video streaming websites



E-commerce websites

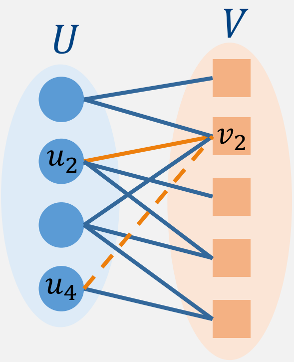
Existing Solutions

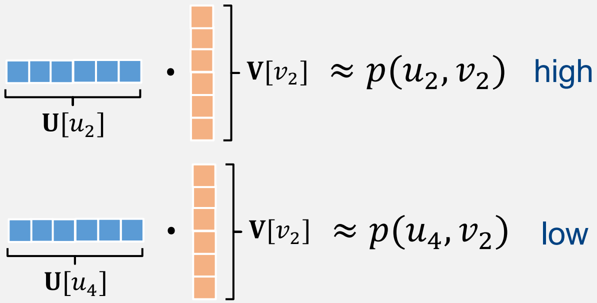
- Traditional Network Embedding Methods
  - DeepWalk, node2vec, LINE, etc.
  - Low-quality embeddings as they overlook the bipartite structures
- BNE Methods
  - BiNE, BiGI, etc.
  - Incur substantial computational costs due to sampling a large number of random walks or expensive training courses
- Embedding-based Collaborative filtering (CF) methods
  - BPR, NCF, NGCF, LightGCN, etc.
  - Only for recommendation purpose and most only support unweighted bipartite graphs (implicit feedback)

Our Solution: Objective Function

- For heterogeneous nodes
  - MHP $p(u_i, v_j)$ measures the strength of association between u_i, v_j
  - The dot product of embeddings preserves strength of connections.

$$\mathcal{O}_1 = \min_{\mathbf{U},\mathbf{V} \in \mathbb{R}^{|U| \cdot k}} \frac{1}{|U| \cdot |V|} \sum_{u_i \in U, v_j \in V} \left(\mathbf{U}[u_i] \cdot \mathbf{V}[v_j] - p(u_i, v_j)\right)^2$$



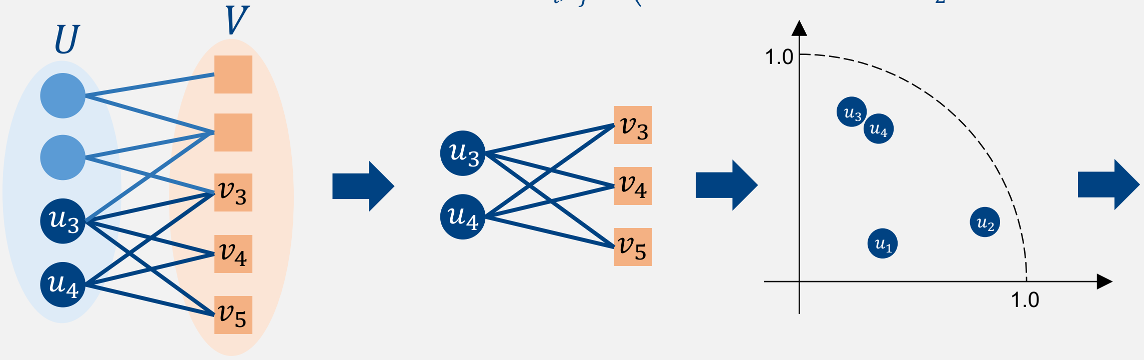


Good for ranking tasks, e.g., recommendation.

Our Solution: Objective Function

- For homogeneous nodes
  - MHS $s(u_i, u_j)$ measures similarity of u_i, u_j in topology
  - Similar nodes have similar embeddings.

$$\mathcal{O}_2 = \min_{\mathbf{U} \in \mathbb{R}^{|U| \cdot k}} \frac{1}{|U|^2} \sum_{u_i, u_j \in U} \left(\left\| \frac{\mathbf{U}[u_i]}{\|\mathbf{U}[u_i]\|_2} - \frac{\mathbf{U}[u_j]}{\|\mathbf{U}[u_j]\|_2} \right\|_2^2 - 2(1 - s(u_i, u_j)) \right)^2$$



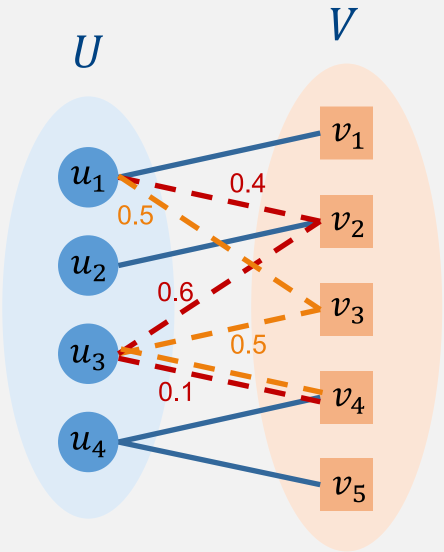
Good for classification tasks, e.g., link prediction.

- Unified objective function $\mathcal{O} = \mathcal{O}_1 + \mathcal{O}_2$

Our Solution: MHP

ACM SIGMOD PODS 2022
Philadelphia, PA, USA

- Multi-Hop Heterogeneous Proximity (MHP)



Let $P_{2\ell+1}$ be the set of paths of length-$(2\ell+1)$

Length-3 paths P_3:

- path 1: $u_1 \rightarrow v_2 \rightarrow u_3 \rightarrow v_4$
- path 2: $u_1 \rightarrow v_3 \rightarrow u_3 \rightarrow v_4$

$$q_3(u_1, v_4) = w(u_1, v_2) \times w(v_2, u_3) \times w(u_3, v_4) + w(u_1, v_3) \times w(v_3, u_3) \times w(u_3, v_4) = (0.4 \times 0.6 + 0.5 \times 0.5) \times 0.1 = 0.049$$

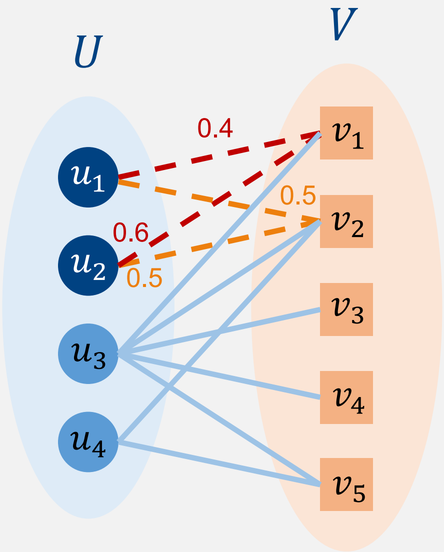
$$p(u_1, v_4) = \sum_{\ell=0}^{\tau} \omega(\ell) \cdot q_{2\ell+1}(u_1, v_4)$$

- $\omega(\ell)$ is the weight for length ℓ
  - Uniform distribution
  - Geometric distribution
  - Poisson distribution

MHP $p(u_1, v_4)$ describes the overall strength of multi-hop connections between heterogeneous nodes u_1 and v_4

Our Solution: MHS

Multi-Hop Homogeneous Similarity (MHS)



Let $P_{2\ell}$ be the set of paths of length-2ℓ

Length-2 paths P_2:

- path 1: $u_1 \rightarrow v_1 \rightarrow u_2$
- path 2: $u_1 \rightarrow v_2 \rightarrow u_2$

$$q_2(u_1,u_2) = w(u_1,v_1)\times w(v_1,u_2) + w(u_1,v_2)\times w(v_2,u_2) = 0.4\times 0.6 + 0.5\times 0.5 = 0.49$$

$$h(u_1,u_2) = \sum_{\ell=0}^{\tau} \omega(\ell)\cdot q_{2\ell}(u_1,u_2)$$

- $\omega(\ell)$ is the weight for length ℓ

$h(u_3,u_1) = h(u_3,u_2) > h(u_1,u_2)$
Large-degree nodes introduce distortion!

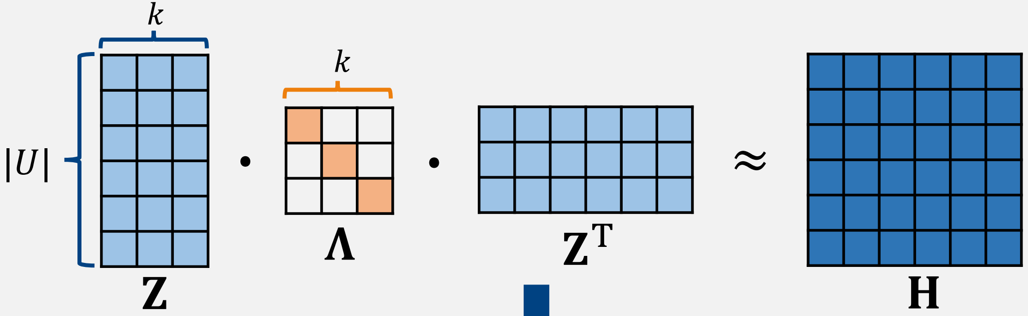
normalization

$$s(u_1,u_2) = \frac{h(u_1,u_3)}{\sqrt{h(u_1,u_1)}\cdot\sqrt{h(u_3,u_3)}}$$

- $0 \leq s(u_1,u_3) \leq 1$

Our Solution: Overview

- Edge weight matrix $\mathbf{W} \in \mathbb{R}^{|U|\times|V|}$, $\mathbf{W}[u_i, v_j]$ is the weight of edge (u_i, v_j)
- Matrix $\mathbf{H} = \sum_{\ell=0}^{\tau} \omega(\ell) \cdot \left(\mathbf{W}\mathbf{W}^{\mathrm{T}}\right)^{\ell}$, $\mathbf{H}[u_i, u_j] = h(u_i, u_j)$
- Theoretically, optimizing objective $\mathcal{O} = \mathcal{O}_1 + \mathcal{O}_2$ is equivalent to finding

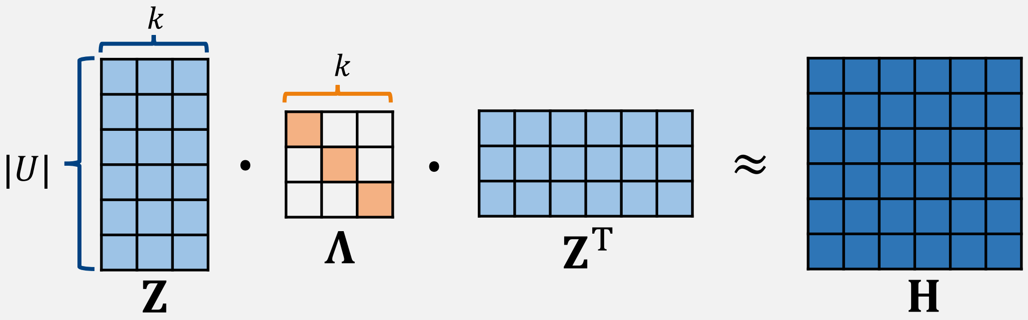


$\mathbf{Z}$ is top-k eigenvectors of $\mathbf{H}$

$$\boxed{\mathbf{U} = \mathbf{Z} \cdot \sqrt{\mathbf{\Lambda}} \;\;\&\;\; \mathbf{V} = \mathbf{W}^{\mathrm{T}} \cdot \mathbf{U}}$$

Our Solution: Overview

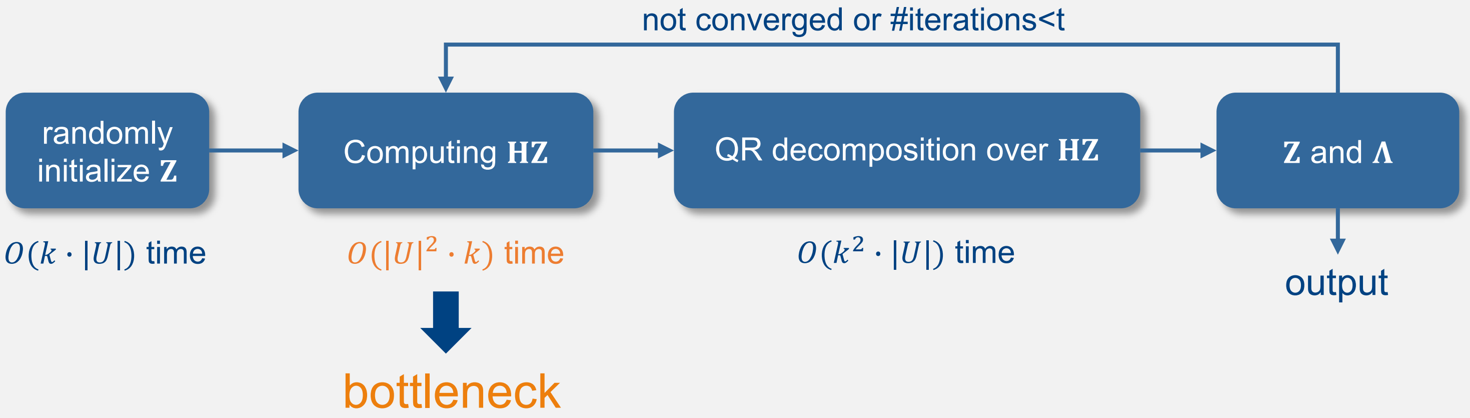
- Edge weight matrix $\mathbf{W} \in \mathbb{R}^{|U|\times|V|}$, $\mathbf{W}[u_i, v_j]$ is the weight of edge (u_i, v_j)
- Matrix $\mathbf{H} = \sum_{\ell=0}^{\tau} \omega(\ell) \cdot \left(\mathbf{W}\mathbf{W}^{\mathrm{T}}\right)^{\ell}$, $\mathbf{H}[u_i, u_j] = h(u_i, u_j)$
- Challenges:
  - The computation of matrix $\mathbf{H}$ requires $O(|U|^2)$ space and $O(|E| \cdot |V|)$ time
  - Finding the top-k eigenvectors of $\mathbf{H}$ takes $O(|U|^2 \cdot k)$ time



$\mathbf{Z}$ is top-k eigenvectors of $\mathbf{H}$

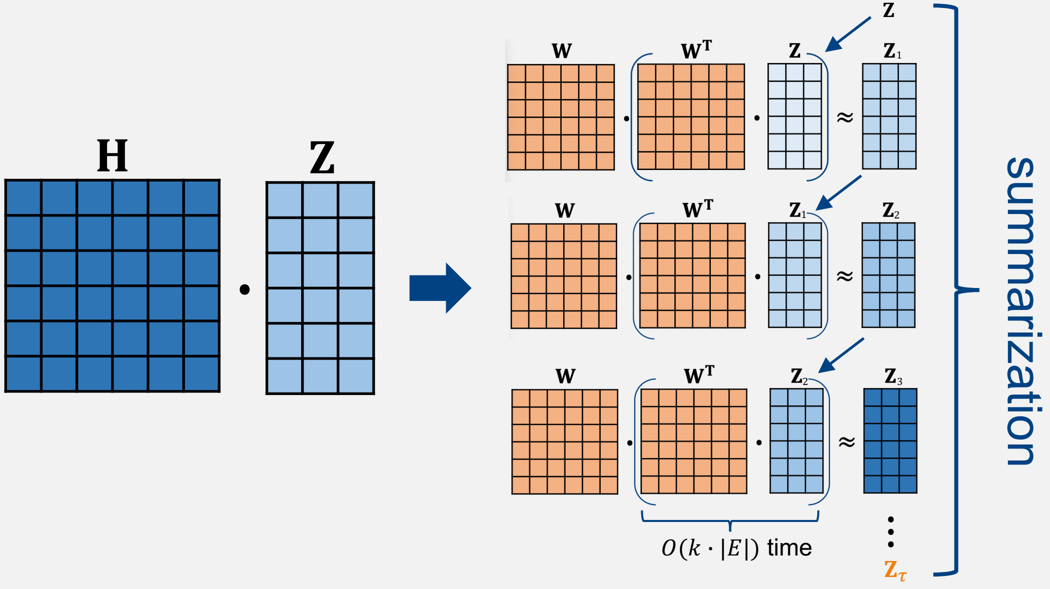
Our Solution: GEBE

- Krylov subspace iterations
  - Find top-k eigenvectors and eigenvalues in the symmetric matrix **H** iteratively
  - Max number of iterations t (e.g., 200)



Our Solution: GEBE

- Power iterations
  - $\mathbf{HZ} = \sum_{\ell=0}^{\tau} \omega(\ell) \cdot \left(\mathbf{WW}^{\mathrm{T}}\right)^{\ell} \cdot \mathbf{Z}$



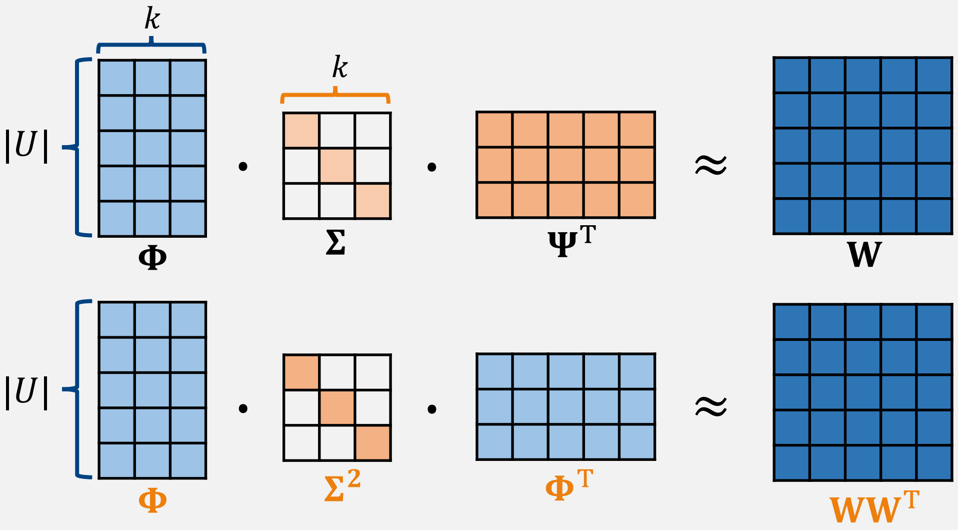
- No need to materialize $\mathbf{H}$
- $O(k\tau \cdot |E|)$ time
- $O\big((|U| + |V|) \cdot k + |E|\big)$ space

Our Solution: GEBE

- Limitations of GEBE
  - t Krylov subspace iterations are needed to approximate the eigenvectors $\mathbf{Z}$ and eigenvalues $\mathbf{\Lambda}$
    - t is large, more QR decomposition & matrix multiplication costs
    - t is small, inaccurate $\mathbf{Z}$ and $\mathbf{\Lambda}$ → compromised quality
  - τ power iterations are needed to approximate $\mathbf{HZ}$
    - $\boldsymbol{\tau}$ is large, more matrix multiplication costs
    - $\boldsymbol{\tau}$ is small, higher-order information is not preserved → compromised quality

Our Solution: GEBE^p

- If $\omega(\ell)$ follows Poisson distribution, $\mathbf{H} = \sum_{\ell=0}^{\infty} \frac{e^{-\lambda}\lambda^{\ell}}{\ell!} \cdot \left(\mathbf{W}\mathbf{W}^{\mathrm{T}}\right)^{\ell}$
- The top-k singular value decomposition of $\mathbf{W}$



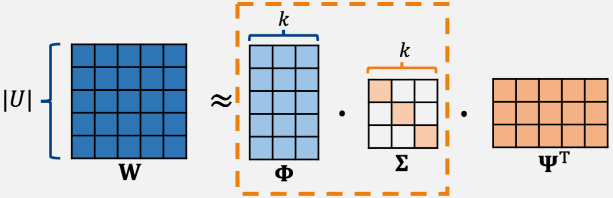
$\boldsymbol{\Psi}\boldsymbol{\Psi}^{\mathrm{T}} = \mathbf{I}$ since $\boldsymbol{\Psi}$ is unitary

Our Solution: GEBE^p

- If $\omega(\ell)$ follows Poisson distribution, $\mathbf{H} = \sum_{\ell=0}^{\infty} \frac{e^{-\lambda}\lambda^{\ell}}{\ell!} \cdot \left(\mathbf{W}\mathbf{W}^{\mathrm{T}}\right)^{\ell}$
- Using the matrix property $e^{\mathbf{M}} = \sum_{\ell=0}^{\infty} \frac{\mathbf{M}^{\ell}}{\ell!}$

$$\mathbf{H} = \frac{e^{\lambda \mathbf{W}\mathbf{W}^{\mathrm{T}}}}{e^{\lambda}} = \frac{e^{\lambda \mathbf{\Phi}\mathbf{\Sigma}^2\mathbf{\Phi}^{\mathrm{T}}}}{e^{\lambda}} = \mathbf{\Phi}\,\frac{e^{\lambda\mathbf{\Sigma}^2}}{e^{\lambda}}\,\mathbf{\Phi}^{\mathrm{T}}$$

$$\boxed{\mathbf{U} = \mathbf{\Phi} \cdot \sqrt{\frac{e^{\lambda\mathbf{\Sigma}^2}}{e^{\lambda}}} \quad \& \quad \mathbf{V} = \mathbf{W}^{\mathbf{T}} \cdot \mathbf{U}}$$



The top-k singular value decomposition of $\mathbf{W}$

- takes $O\left(\left(|E| \cdot k + |U| \cdot k^2\right) \cdot \frac{\log(|V|)}{\epsilon}\right)$ time
- takes $O\left(\left(|U| + |V|\right) \cdot k + |E|\right)$ space

Experiments: Methods

- Our methods: GEBE, GEBE^p
- BNE methods
  - BiNE [SIGIR'18]
  - BiGI [WSDM'21]
- Network embedding methods
  - DeepWalk [KDD'14]
  - Node2vec [KDD'16]
  - LINE [WWW'15]
  - NRP [PVLDB'20]
- Collaborative filtering (CF) methods
  - BPR [UAI'09]
  - NCF [WWW'17]
  - NGCF [SIGIR'19]
  - LightGCN [SIGIR'20]
  - LCFN [ICML'20]
  - LR-GCCF [AAAI'20]
  - SCF [RecSys'18]
  - GCMC [KDD'19]
  - CSE [WWW'19]

for implicit data

for explicit data

- All methods are implemented in Python

Experiments: Link Prediction

- Unweighted bipartite graphs
  - randomly sample 60% edges as the training data
  - remaining 40% edges and an equal number of negative edges as test data

Name	$\|U\|$	$\|V\|$	$\|E\|$	Type
Wikipedia	15K	3.2K	64K	Author-page-edit
Pinterest	55.2K	9.9K	1.5M	User-image-pin
Yelp	31.7K	38K	1.56M	User-restaurant-visit
MIND	877K	97.5K	18.2M	User-news-click
Orkut	2.78M	8.73M	327M	User-group-affiliation

Experiments: Link Prediction

Method	Wikipedia		Pinterest		Yelp		MIND		Orkut	
	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR	AUC-ROC	AUC-PR
GEBE^{p}	**0.964**	**0.965**	**0.728**	**0.701**	**0.799**	**0.803**	**0.959**	**0.956**	**0.95**	**0.958**
GEBE (Poisson)	0.959	0.964	0.722	0.698	0.796	0.799	0.951	0.949	**0.95**	**0.958**
GEBE (Geometric)	0.949	0.959	0.718	0.696	0.795	0.796	0.944	0.942	0.928	0.934
GEBE (Uniform)	0.951	0.96	0.715	0.697	0.794	0.796	0.942	0.941	0.928	0.932
BiNE	0.956	0.964	0.704	0.647	0.797	0.801	-	-	-	-
BiGI	0.954	0.96	0.697	0.641	0.762	0.754	-	-	-	-
DeepWalk	0.899	0.84	0.659	0.563	0.663	0.622	-	-	-	-
node2vec	0.866	0.838	0.66	0.566	0.674	0.643	-	-	-	-
LINE	0.963	**0.965**	0.713	0.692	0.796	0.801	0.885	0.907	0.752	0.753
NRP	0.929	0.941	0.712	0.689	0.781	0.785	0.924	0.909	0.911	0.912
BPR	0.951	0.96	0.7	0.624	0.733	0.711	0.939	0.931	-	-
NCF	0.947	0.952	0.691	0.622	0.749	0.732	0.914	0.911	-	-
NGCF	0.96	0.964	0.703	0.637	0.772	0.764	-	-	-	-
LightGCN	0.955	0.964	0.661	0.613	0.668	0.682	-	-	-	-
GCMC	0.948	0.945	0.712	0.689	0.752	0.756	-	-	-	-
CSE	0.944	0.954	0.719	0.656	0.761	0.759	0.949	0.947	0.858	0.856
LCFN	0.922	0.932	0.707	0.682	0.687	0.692	-	-	-	-
LR-GCCF	0.865	0.872	0.658	0.551	0.673	0.581	0.919	0.89	-	-
SCF	0.936	0.944	0.719	0.697	0.748	0.753	-	-	-	-

- The highest scores are in bold
- The 2nd (3rd) best results are double- (single-) underlined
- GEBE^{p} is consistently the best
- MHS is important for link prediction
  - BiNE and CSE consider the preservation of relationships between homogeneous nodes, but totally different from MHS
  - Only GEBE & GEBE^{p} consider multi-hop similarities between homogeneous nodes

Experiments: Top-10 Recommendation

- Weighted bipartite graphs.
  - randomly sample 60% edges as the training data
  - use the remaining 40% edges as the ground-truth for testing

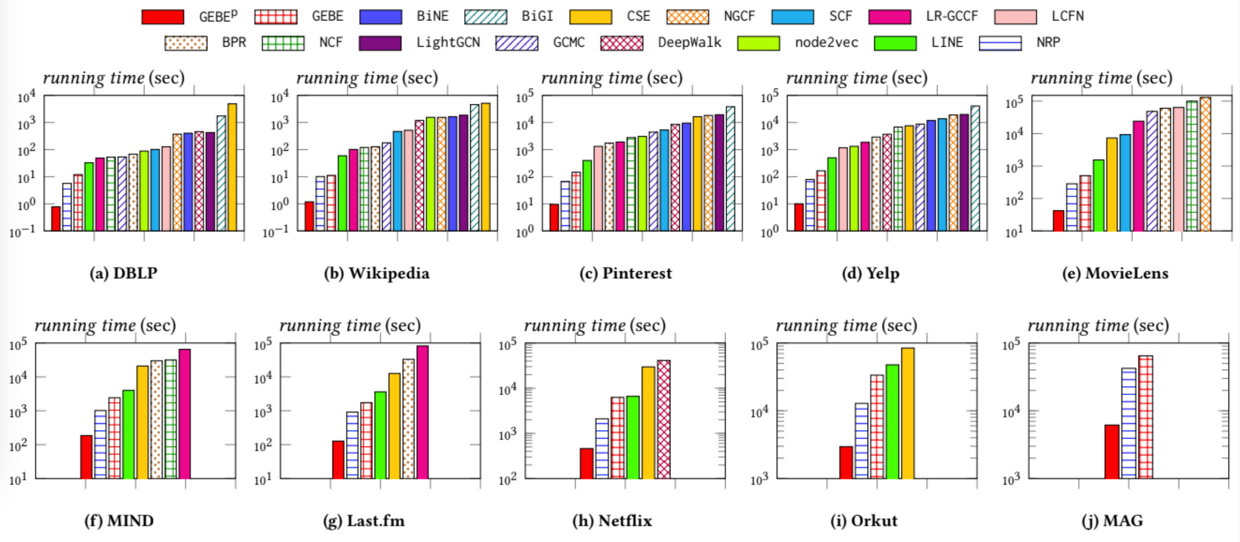
Name	$\|U\|$	$\|V\|$	$\|E\|$	Type
DBLP	6K	1.3K	29.3K	Author-venue-count
MovieLens	69.9K	10.7K	10M	User-movie-rating
Last.fm	359.4K	160.2K	17.56M	User-music-#play
Netflix	480.2K	17.8K	100.48M	User-movie-rating
MAG	10.54M	2.78M	1.1B	Paper-word-occurrence

Experiments: Top-10 Recommendation

Method	DBLP			MovieLens			Last.fm			Netflix			MAG		
	F1	NDCG	MRR	F1	NDCG	MRR	F1	NDCG	MRR	F1	NDCG	MRR	F1	NDCG	MRR
GEBE^p	**0.214**	**0.261**	**0.601**	**0.266**	**0.304**	**0.558**	0.534	0.554	**0.778**	**0.342**	**0.366**	**0.611**	**0.265**	**0.286**	**0.468**
GEBE (Poisson)	0.212	0.258	0.581	0.256	0.294	0.548	0.53	0.549	0.766	0.334	0.354	0.586	0.261	0.284	0.465
GEBE (Geometric)	0.209	0.255	0.579	0.255	0.294	0.55	0.53	0.549	0.75	0.334	0.354	0.582	0.256	0.284	0.467
GEBE (Uniform)	0.209	0.255	0.579	0.253	0.292	0.55	0.53	0.55	0.755	0.334	0.353	0.58	0.255	0.282	0.464
BiNE	0.18	0.216	0.5	0.252	0.28	0.527	-	-	-	-	-	-	-	-	-
BiGI	0.191	0.23	0.526	0.255	0.293	0.536	-	-	-	-	-	-	-	-	-
DeepWalk	0.028	0.027	0.066	-	-	-	-	-	-	-	-	-	-	-	-
node2vec	0.032	0.029	0.063	-	-	-	-	-	-	-	-	-	-	-	-
LINE	0.061	0.063	0.16	0.242	0.262	0.477	0.182	0.196	0.315	0.155	0.166	0.284	-	-	-
NRP	0.191	0.244	0.567	0.212	0.236	0.464	0.509	0.542	0.745	0.287	0.308	0.534	0.222	0.23	0.436
BPR	0.209	0.253	0.568	0.258	0.278	0.519	0.497	0.511	0.709	-	-	-	-	-	-
NCF	0.185	0.209	0.493	0.212	0.215	0.458	-	-	-	-	-	-	-	-	-
NGCF	0.189	0.215	0.495	0.219	0.22	0.462	-	-	-	-	-	-	-	-	-
LightGCN	0.192	0.224	0.492	0.19	0.199	0.414	-	-	-	-	-	-	-	-	-
GCMC	0.151	0.182	0.463	0.234	0.244	0.491	-	-	-	-	-	-	-	-	-
CSE	0.211	0.234	0.483	0.26	0.296	0.55	**0.54**	**0.555**	0.753	0.317	0.343	0.58	-	-	-
LCFN	0.188	0.21	0.493	0.22	0.231	0.454	-	-	-	-	-	-	-	-	-
LR-GCCF	0.168	0.214	0.493	0.166	0.168	0.392	0.161	0.174	0.359	-	-	-	-	-	-
SCF	0.196	0.222	0.496	0.165	0.171	0.372	-	-	-	-	-	-	-	-	-

- 3 popular metrics: F1, NDCG, and MRR
- GEBE^p performs best in most cases
- Most CF-based methods neglect edge weights (only support implicit data), leading to inferior performance
- MHP is important for recommendation

Experiments: Efficiency



- GEBE^p is faster often by orders of magnitude.
- On MAG (1.1B edges)
  - GEBE^p takes 1.7 hrs.
  - NRP needs 11.8 hrs.
  - Other methods fail to finish within 3 days

THANK YOU

Scalable and Effective Bipartite Network Embedding

Renchi Yang, Jieming Shi, Keke Huang, Xiaokui Xiao