Arbitrary-Order Proximity Preserved Network Embedding

Ziwei Zhang
Tsinghua U

Peng Cui
Tsinghua U

Xiao Wang
Tsinghua U

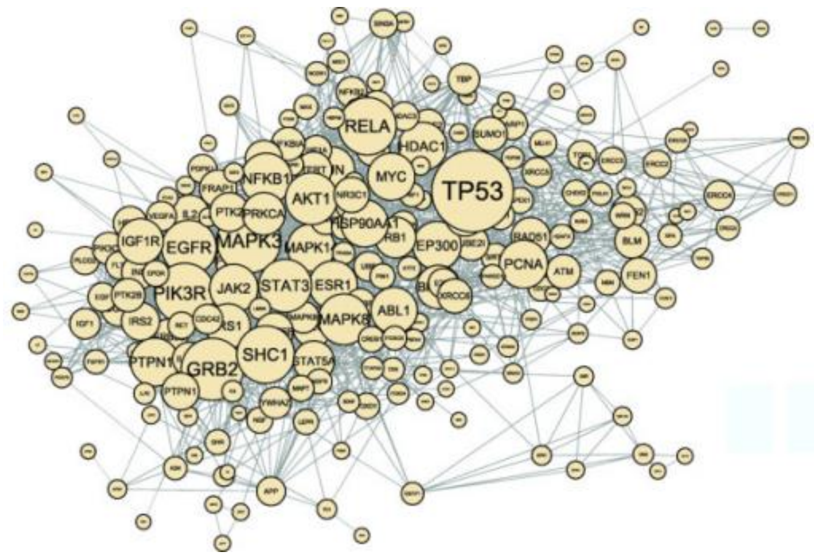
Jian Pei
JD&Simon Fraser U

Xuanrong Yao
Tsinghua U

Wenwu Zhu
Tsinghua U


Network Data is Ubiquitous

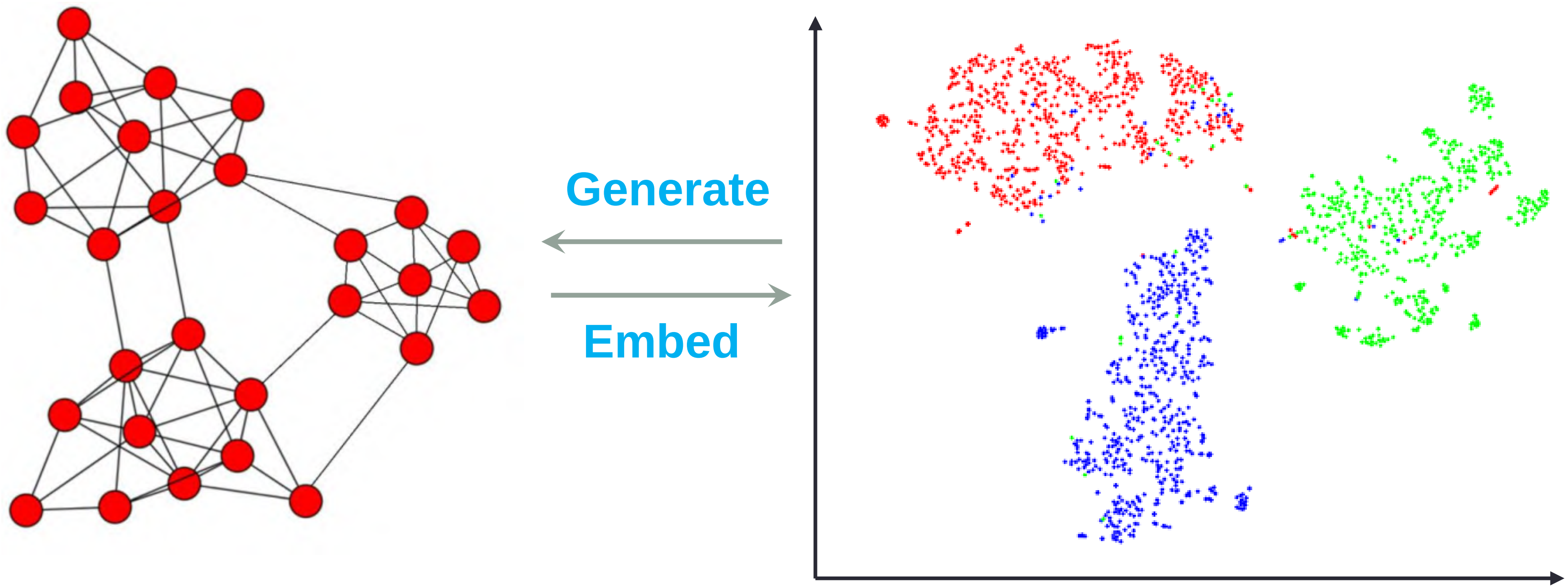
Social Network



Biology Network

Traffic Network

Network Embedding: Vector Representation of Nodes

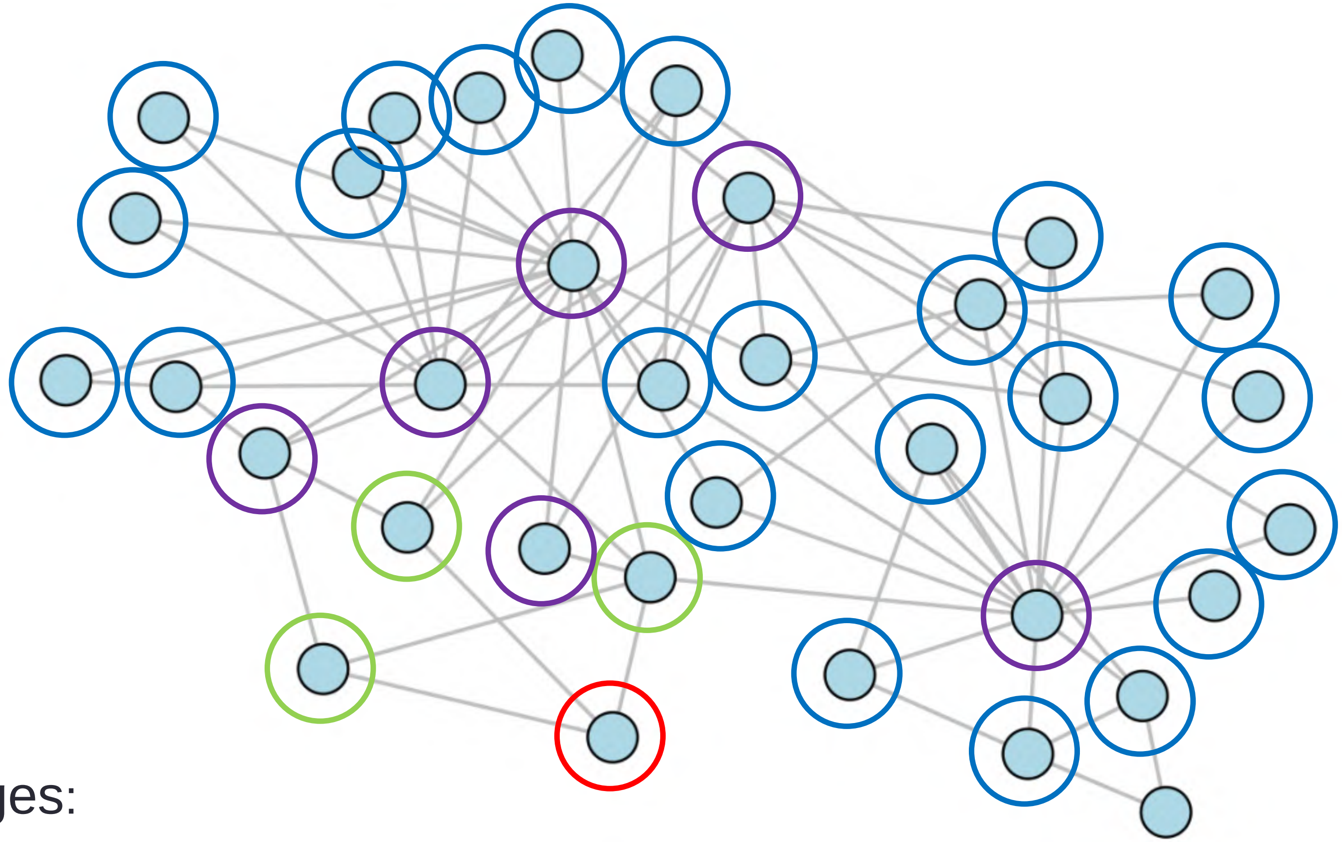


- Apply feature-based machine learning algorithms
- Fast compute nodes similarity
- Support parallel computing

- Applications: link prediction, node classification, community detection, measuring centrality, anomaly detection ...

High-Order Proximity

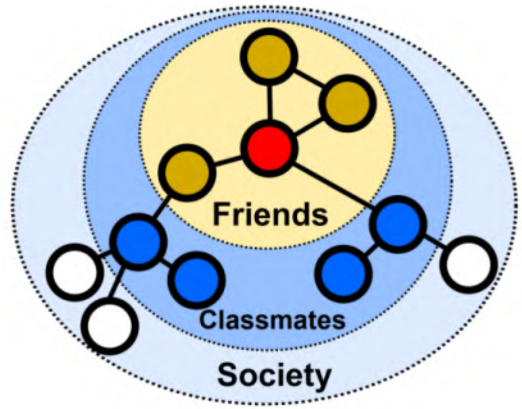
- High-order proximity: key in capturing the underlying structure of networks



- Advantages:
  - Solve the sparsity problem of network connections
  - Measure indirect relationship between nodes

Different High-Order Proximities

- Different networks/tasks require different high-order proximities
  - E.g., multi-scale classification (Bryan Perozzi, et al, *ASONAM, 2017*)



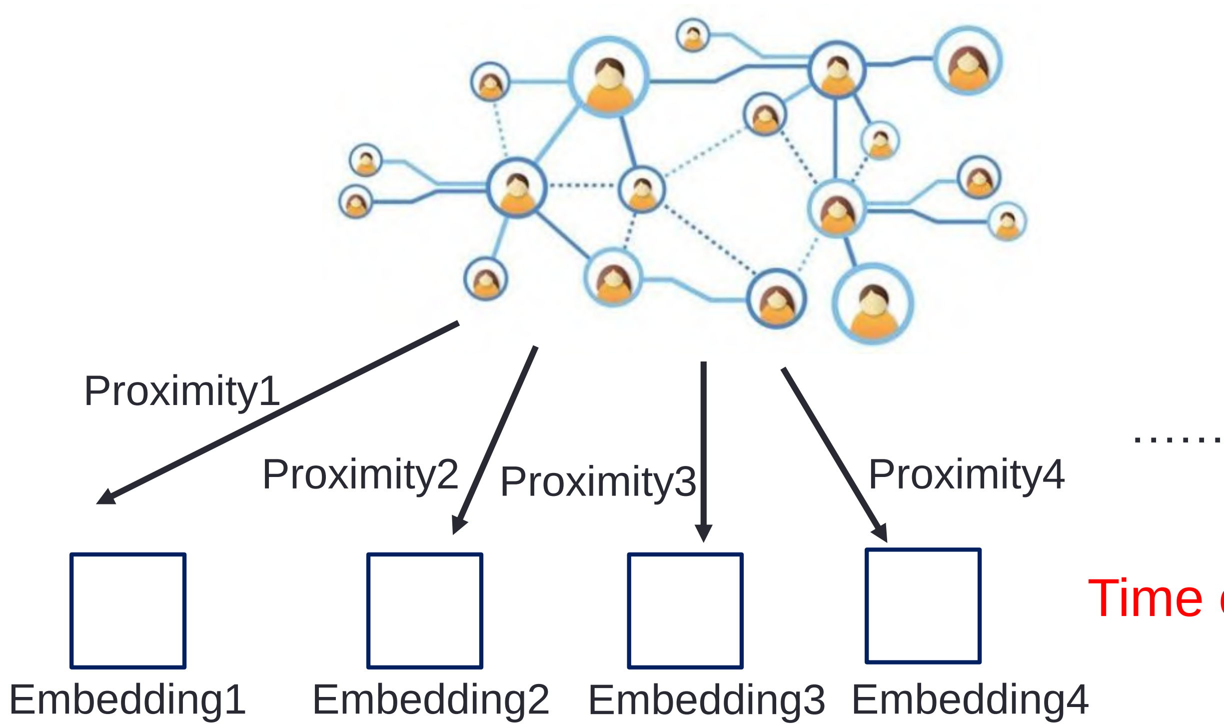
  - E.g., networks with different scales and sparsity
- Proximities of different orders can also be arbitrarily weighted
  - E.g., equal weights, exponentially decayed weights (Katz)

Existing Methods

- Methods based on random-walks
  - DeepWalk, B. Perozzi, et al. *KDD 2014.*
  - LINE, J. Tang, et al. *WWW 2015.*
  - Node2vec, A. Grover, et al. *KDD 2016.*
  - Random walks on networks + skip-gram model from NLP
- Methods based on matrix factorization
  - GraRep, S. Cao, et al. *CIKM, 2015.*
  - HOPE, M. Ou, et al. *KDD 2016.*
  - M-NMF, X. Wang, et al. *AAAI 2017.*
  - Objective function based on matrix factorization + optimization
- Methods based on deep learning
  - SDNE, D. Wang, et al. *KDD 2016.*
  - DVNE, D. Zhu, et al. *KDD 2018.*
  - Deep auto-encoder to preserve the non-linearity

Existing Methods (cont.)

- Existing methods can only preserve one fixed high-order proximity
  - Different high-order proximities have to be calculated separately



Time consuming!

→ How to preserve arbitrary-order proximity simultaneously?

Key question: what is the underlying relationship between different proximities?

Problem Formulation

- High-order proximity: a polynomial function of the adjacency matrix

$$S = \mathcal{F}(A) = w_1 A^1 + w_2 A^2 + \cdots + w_q A^q$$

  - q: order; $w_1 \ldots w_q$: weights, assuming to be non-negative
  - A: could be replaced by other variations (such as the Laplacian matrix)
- Objective function: matrix factorization

$$\min_{U^*, V^*} \left\| S - U^* {V^*}^T \right\|_F^2$$

  - $U^*, V^* \in \mathbb{R}^{N \times d}$: left/right embedding vectors
  - d: dimensionality of the space
- Optimal solution: Singular Value Decomposition (SVD)
  - $[U, \Sigma, V]$: top-d SVD results

$$U^* = U\sqrt{\Sigma}, V^* = V\sqrt{\Sigma}$$

- However, direct calculation is time-consuming

Problem Transformation

- Problem Transformation
  - $[U, \Sigma, V]$: top-d SVD . $[\Lambda, \mathrm{X}]$: top-d eigen-decomposition
  - Theorem:

$$\begin{cases} \mathbf{U}(:, i) = \mathbf{X}(:, i) \\ \Sigma(i, i) = abs(\mathbf{\Lambda}(i, i)) \\ \mathbf{V}(:, i) = \mathbf{X}(:, i) sign(\mathbf{\Lambda}(i, i)) \end{cases} \quad , and$$

$$\begin{cases} \mathbf{X}(:, i) = \mathbf{U}(:, i) \\ \mathbf{\Lambda}(i, i) = \Sigma(i, i) sign\left(\mathbf{U}(:, i) \cdot \mathbf{V}(:, i)\right) \end{cases}$$

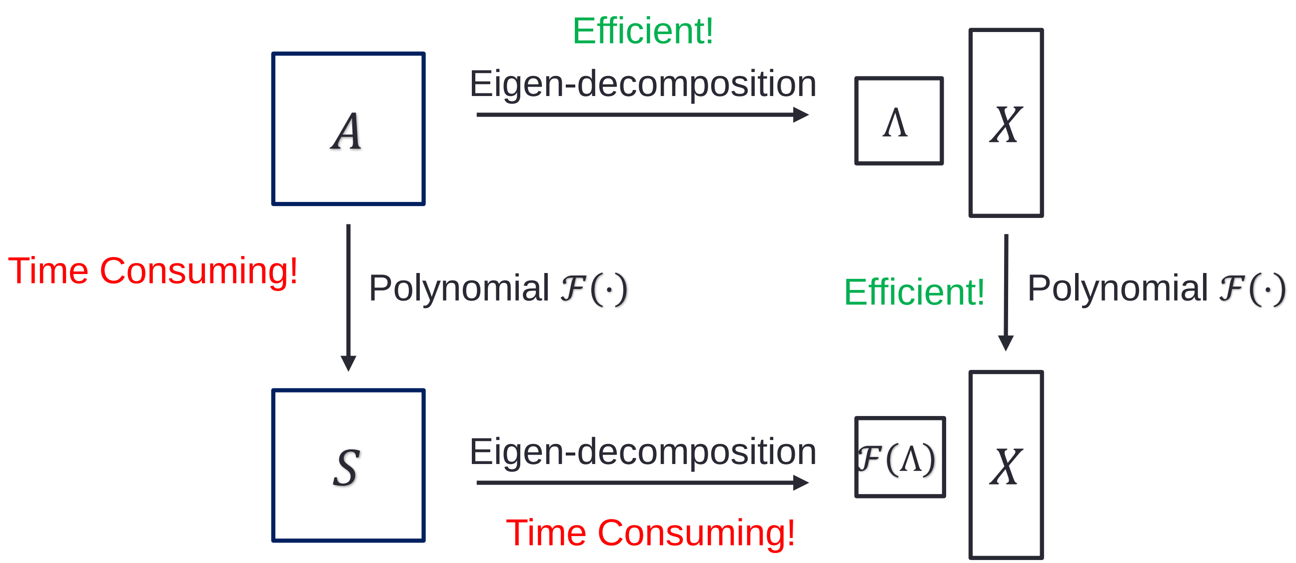
  - How to solve $[\Lambda, \mathrm{X}]$ for $S = f(A) = w_1 A^1 + w_2 A^2 + \cdots + w_q A^q$

Eigen-decomposition Reweighting

- Eigen-decomposition reweighting

THEOREM 4.2 (EIGEN-DECOMPOSITION REWEIGHTING). *If* $[\boxed{\lambda}, \boxed{\mathbf{x}}]$ *is an eigen-pair of* $\mathbf{A}$*, then* $[\boxed{\mathcal{F}(\lambda)}, \boxed{\mathbf{x}}]$ *is an eigen-pair of* $\mathbf{S} = \mathcal{F}(\mathbf{A})$.

- $Ax = \lambda x \rightarrow A^2 x = \lambda^2 x \rightarrow \mathcal{F}(A)x = \mathcal{F}(\lambda)x$



- Insights: high-order proximity is simply re-weighting dimensions!
  - Eigenvectors as coordinates, eigenvalues as weights

Eigen-decomposition Reweighting (cont.)

- Re-ordering of dimensions

$\lambda_1 \quad \lambda_2 \quad \lambda_3 \quad \ldots \quad \lambda_d$

polynomial function ?

$\lambda'_1 \quad \lambda'_2 \quad \lambda'_3 \quad \ldots \quad \lambda'_d$

top-d eigen-decomposition of A

$\downarrow$ ×

top-d eigen-decomposition of S

$\lambda_1 \quad \lambda_2 \quad \lambda_3 \quad \ldots \quad \lambda_d \quad \ldots \quad \lambda_l$

polynomial function

$\lambda'_1 \quad \lambda'_2 \quad \lambda'_3 \quad \ldots \quad \lambda'_d$

top-l eigen-decomposition of A

$\downarrow$ √

top-d eigen-decomposition of S

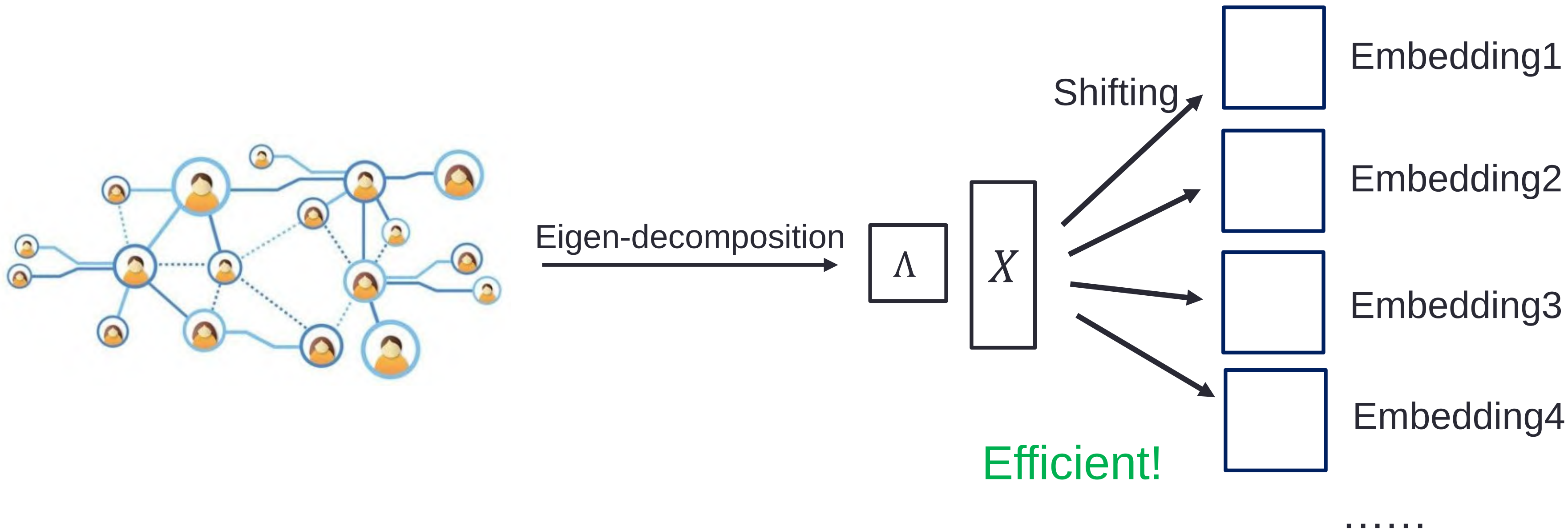
THEOREM 4.3. *l satisfies that the top l eigenvalues of* $\mathbf{A}$ *have d positive, i.e.*

$$l = \mathcal{L}(\mathbf{A}, d) = \min\ l' \quad s.t. \sum_{j=1}^{l'} \mathbb{I}(\lambda_j > 0) = d,$$

- d vs. l: $l \approx 2d$
  - Proven for random (Erdos-Renyi), random power-law networks
  - Verified on experiments

Preserving Arbitrary-Order Proximity

- Shifting across different orders/weights:



- Preserve arbitrary-order proximity simultaneously
- Low marginal cost for preserving multiple proximities
- Accurate (global optimal) and efficient (linear time complexity)

Algorithm Framework

Algorithm 1 AROPE: ARbitrary-Order Proximity preserved Embedding

---

Require: Adjacency Matrix $\mathbf{A}$, Dimensionality d, Different High-Order Proximity Functions $\mathcal{F}_1(\cdot), \ldots, \mathcal{F}_r(\cdot)$

Ensure: Embedding vectors $\mathbf{U}_i^*, \mathbf{V}_i^*$ for $\mathcal{F}_i(\cdot)$, $1 \leq i \leq r$

1: Calculate the top-l eigen-decomposition $[\mathbf{\Lambda}, \mathbf{X}]$ of $\mathbf{A}$
2: **for** i in 1:r **do**
3: Calculate the reweighted eigenvalues $\mathbf{\Lambda}' = \mathcal{F}_i(\mathbf{\Lambda})$
4: Sort $\mathbf{\Lambda}'$ in descending order of the absolute value and select the top-d
5: Calculate the top-d SVD results using Eq. (4)
6: Return $\mathbf{U}_i^*, \mathbf{V}_i^*$ using Eq. (3)
7: **end for**

- Time complexity: $O\big(T(Nl^2 + Ml) + r(l + Nd)\big)$
  - N: number of nodes; M: number of edges; T: iteration; d: embedding dimension ($l \approx 2d$); r: number of shifting
  - Linear w.r.t. the network size
  - Marginal cost for preserving multiple proximities

Special Cases of the Proposed Method

- Common Neighbors: the second order

$$S = A^2$$

- Propagation: weighted combination of the second and the third order

$$S = w_2 A^2 + w_3 A^3$$

- Katz Proximity: infinite order with exponentially decayed weights

$$S = \sum_{i=1}^{+\infty} \beta^i A^i$$

- Eigenvector Centrality: the first dimension

$$U^*(:,1) \propto eigenvector\_centrality$$

  - Regardless of what high-order proximity is

Experimental Setting: Datasets

- Datasets:
  - BlogCatalog, Flickr, Youtube: online social networks where nodes represent users and edges represent relationships between users.
  - Wiki: wikipedia hyperlinks, where each node represents a page and each edge represents a hyperlink between two pages. The edges are treated as undirected.

Table 1: The Statistics of Datasets

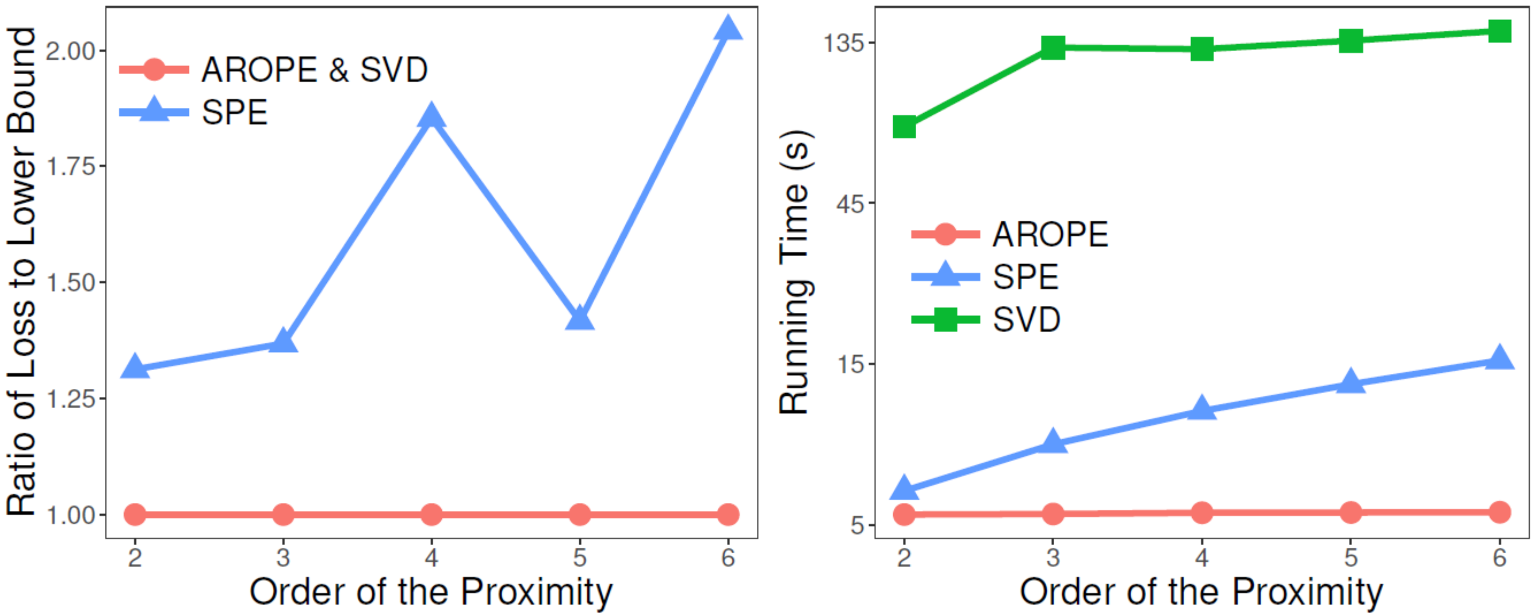
Dataset	# Nodes	# Edges	Average Degree
BlogCatalog	10,312	667,966	64.8
Flickr	80,513	11,799,764	146.6
Youtube	1,138,499	5,980,886	5.3
Wiki	1,791,486	50,888,414	28.4

Experimental Setting: Baselines

- Baselines:
  - DeepWalk (KDD 2014): DFS random walk + skip-gram
  - LINE (WWW 2015): BFS random walk + skip-gram
  - Node2vec (KDD 2016): biased random walk + skip-gram
  - SDNE (KDD 2016): deep auto-encoder
  - NEU (IJCAI 2017): matrix factorization approximation
- Our method:
  - AROPE: search q from {1,2,3,4} and grid search weights
  - AROPE-F: search q from {1,2,3,4} while fixing weights $w_i = 0.1^i$
    - Limit the search space for hyper-parameters
  - Code: https://github.com/ZW-ZHANG/AROPE

Experimental Results

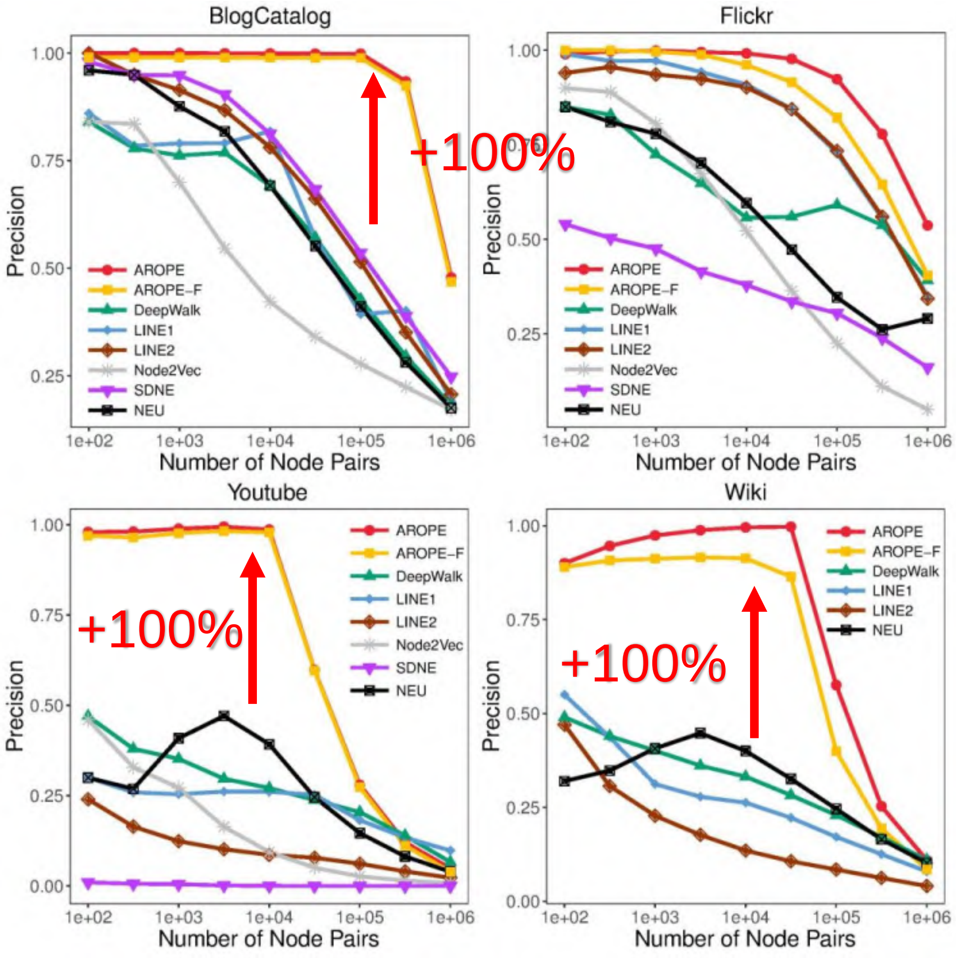
- Preserving the High-Order Proximity



Achieves the global optimal solution while being extremely efficient

Experimental Results

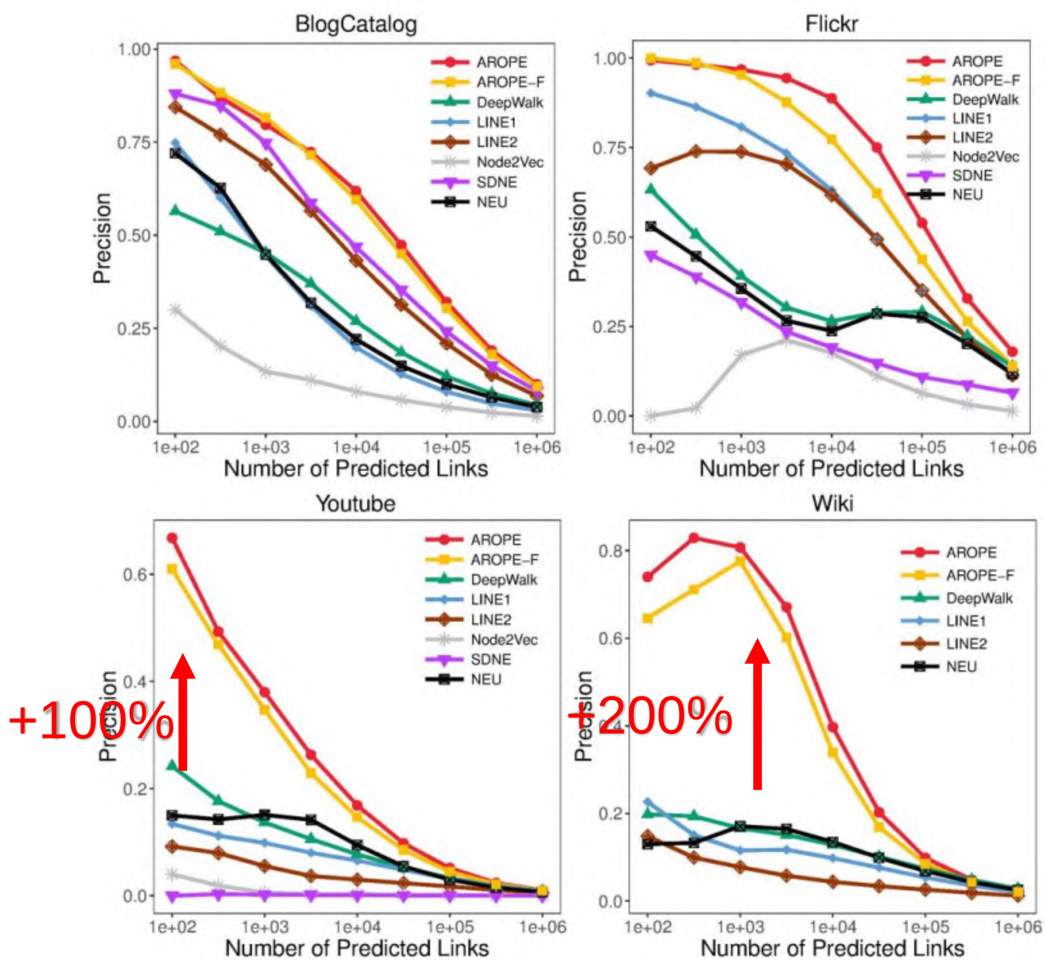
- Network Reconstruction



Better preserve network structure

Experimental Results

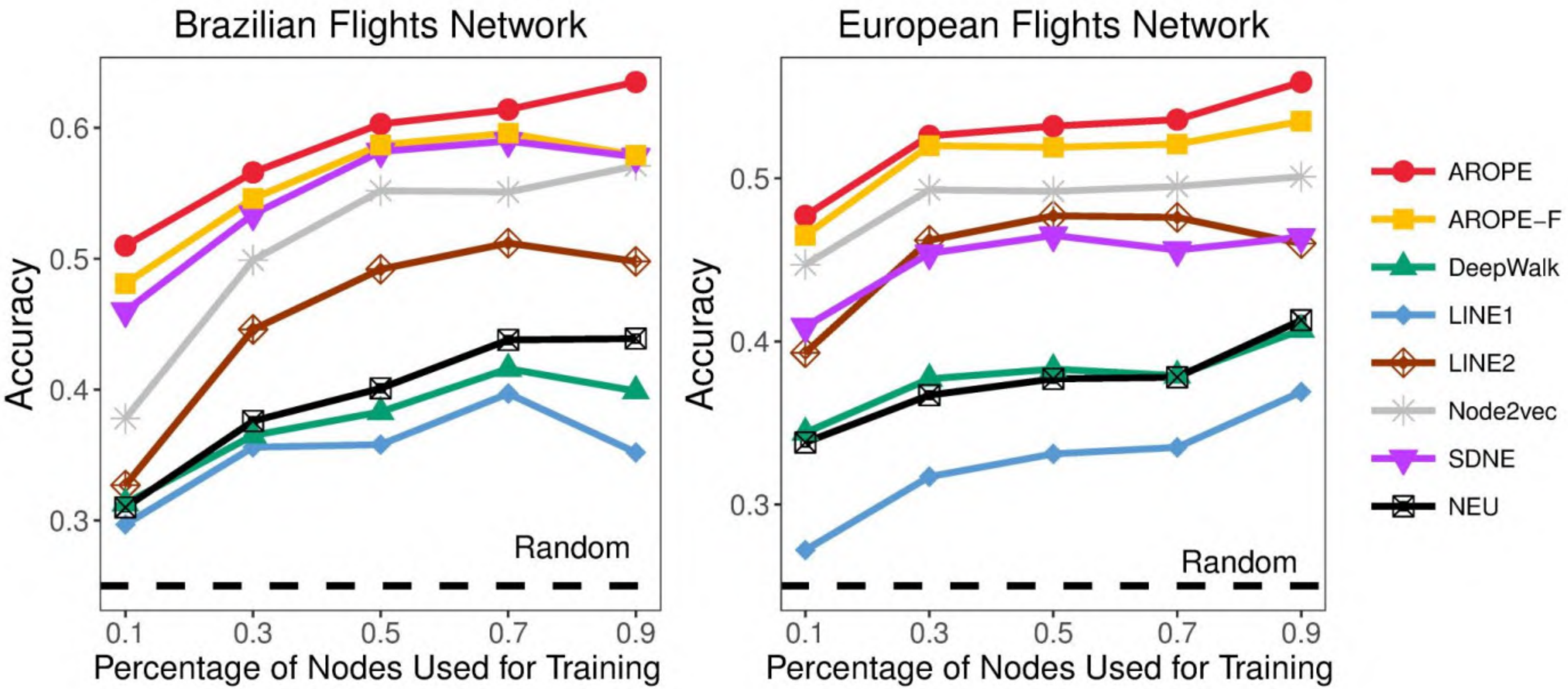
- Link Prediction



Good inference ability: preserve arbitrary-order proximity

Experimental Results

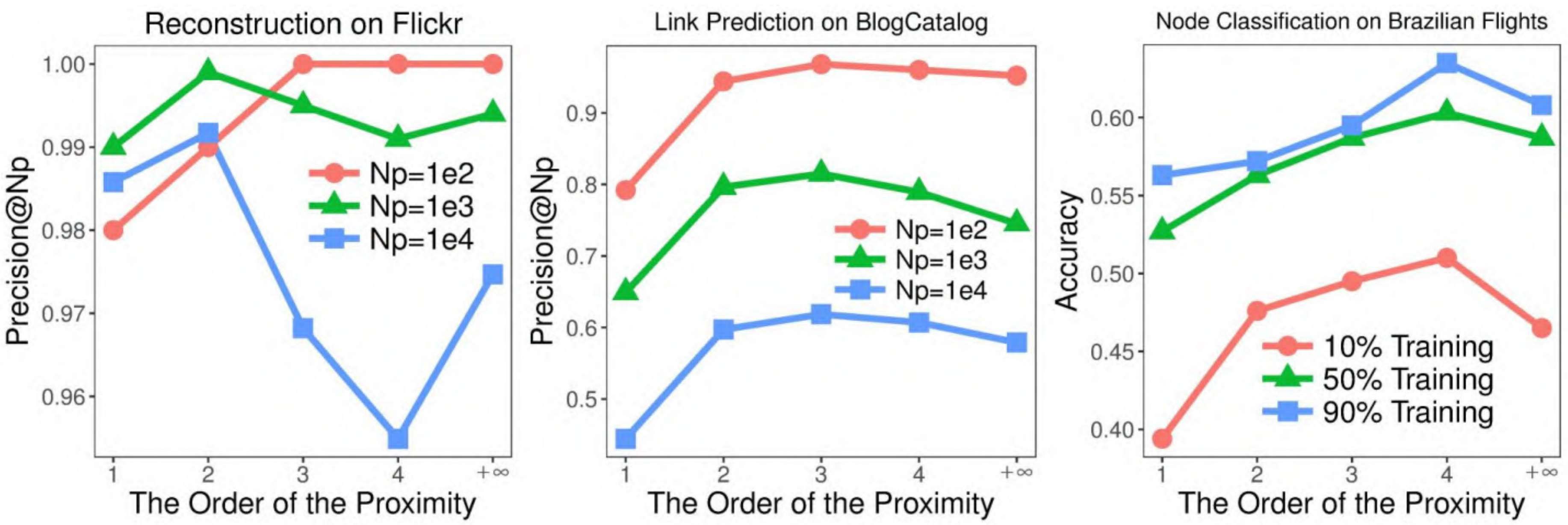
- Node structural role classification (struc2vec, *KDD 2017*)



Capture the structural role of nodes

Experimental Results

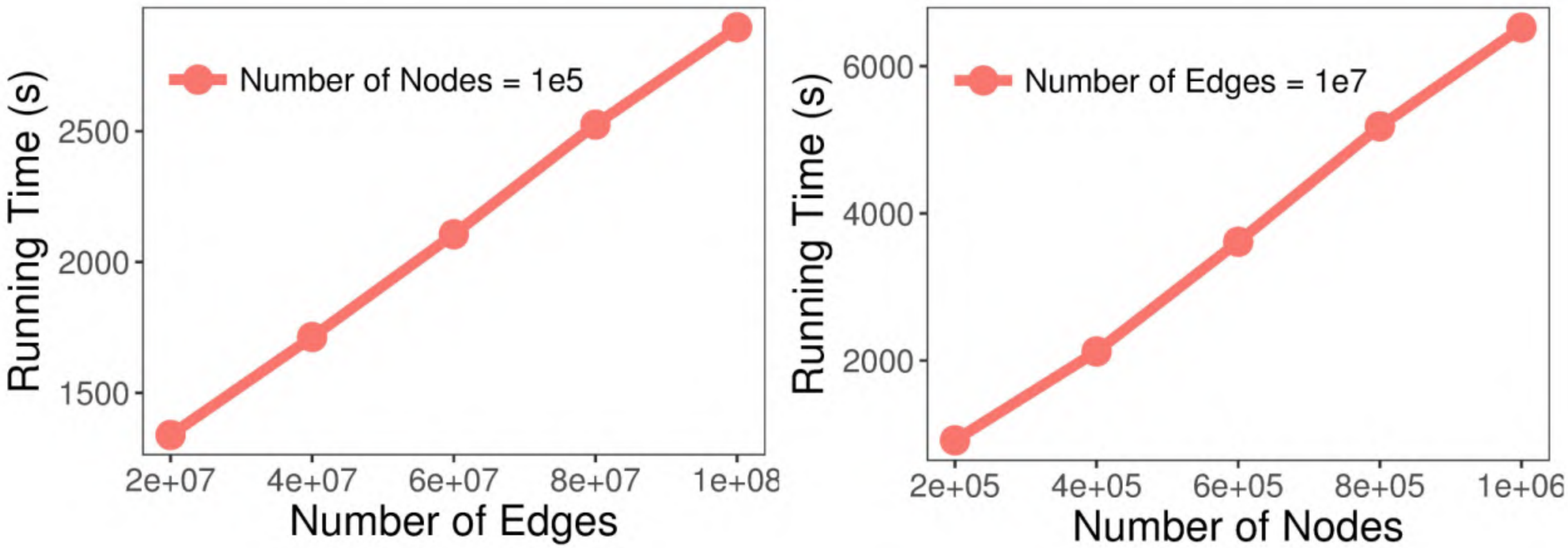
- Parameter analysis



The optimal order varies greatly on different tasks and datasets

Experimental Results

- Scalability analysis



Linear scalability w.r.t. number of nodes and number of edges

(< 2 hours on network with 1 million nodes and 10 millions edges in a single PC)

Conclusion

- Study the problem of preserving arbitrary-order proximity in network embedding
  - Different networks/tasks require different proximities
- Eigen-decomposition Reweighting
  - The intrinsic relationship between different proximities is reweighting and reordering dimensions
  - Preserving arbitrary-order proximity
  - Incorporate many commonly used proximity measures as special cases
- Experimental results:
  - +100% improvements in network reconstruction and link prediction
  - Capture the structural roles of node
  - Linear scalability

Thanks!

Ziwei Zhang, Tsinghua University

zw-zhang16@mails.tsinghua.edu.cn

https://zw-zhang.github.io/

http://nrl.thumedialab.com/