Homogeneous Network Embedding for Massive Graphs via Reweighted Personalized PageRank

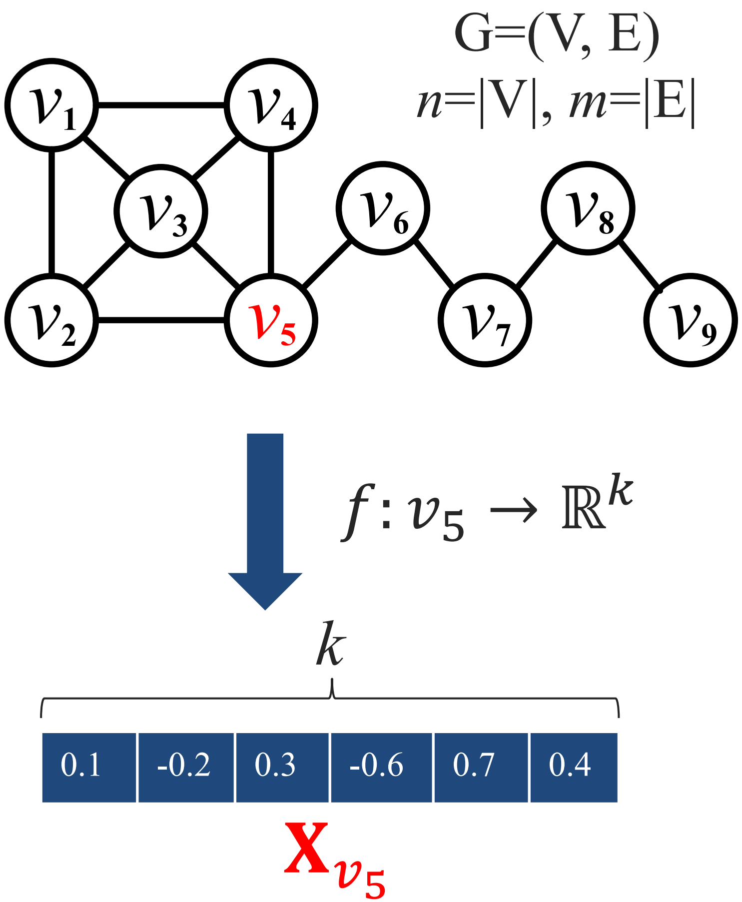
**Renchi Yang, Jieming Shi, Xiaokui Xiao,
Yin Yang, Sourav Saha Bhowmick**

August 2020

Outline

- Problem Definition & Applications
- Existing Work & Motivations
- Proposed solution: NRP
- Experiments

Homogeneous Network Embedding (HNE)



- Link Prediction
  - [Backstrom *et al*., WSDM'2011]
  - [Gupta *et al*., KDD'2013]
- Graph Reconstruction
  - [Radivojac *et al*., Nature methods'2004]
- Node Classification
  - [Perozzi *et al*., KDD'2014]
  - [Ribeiro *et al*., KDD'2017]

NANYANG TECHNOLOGICAL UNIVERSITY

Existing Work

- Learning-based HNE methods
  - with random walks
    - *truncated random walks*: Deepwalk [Perozzi *et al*. KDD14],
    - *biased random walks*: Node2vec [Grover *et al*. KDD16],
    - ***Personalized PageRank (PPR)***:VERSE [Tsitsulin *et al*. WWW18], APP [Zhou *et al*. AAAI]

$$\mathbf{X}_u \cdot \mathbf{X}_v \sim \Pr[u \rightarrow v]$$

A large number of random walks are required !

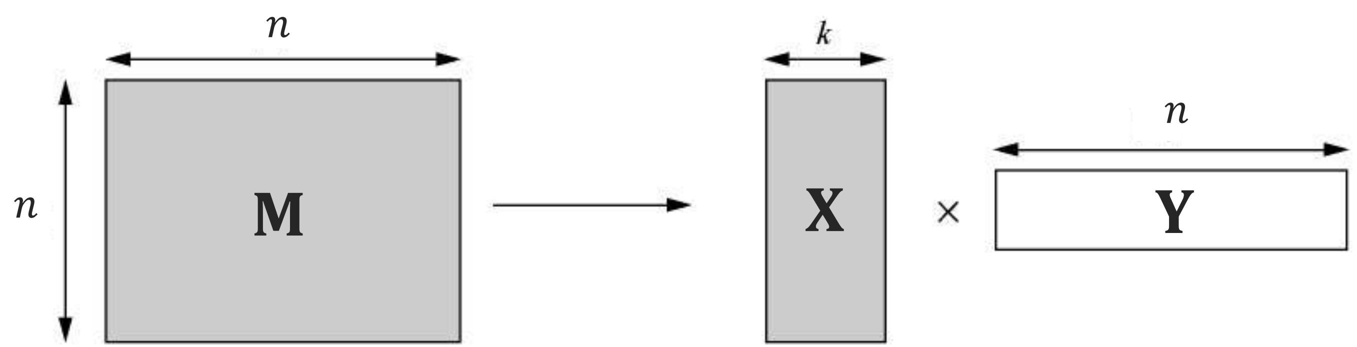
  - without random walks
    - Auto-encoders, graph neural networks (GNN), generative adversarial networks (GAN), long short-term memory networks (LSTM)

Expensive training courses!

Existing Work

- Factorization-based HNE methods
  - Construct an $n \times n$ proximity matrix $\mathbf{M}$
    - Katz score, AROPE [Zhang *et al*. KDD18]
    - PPR, STRAP [Yin *et al*. KDD 2019]
  - Factorize $\mathbf{M} = \mathbf{X} \cdot \mathbf{Y}^{\mathrm{T}}$ (*e.g.*, SVD, NMF)

$O(n^2)$!



Motivations: Efficiency

Exact PPR

$$\mathbf{\Pi} = \sum_{t=0}^{\infty} \alpha(1-\alpha)^t \mathbf{P}^t$$

ι_1-truncated PPR

$$\mathbf{M} = \sum_{t=1}^{\iota_1} \alpha(1-\alpha)^t \mathbf{P}^t$$

Space: $\cancel{\boldsymbol{O}(\boldsymbol{n}^2)}$ **too dense!**

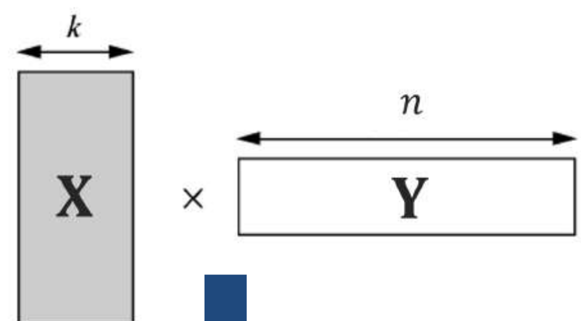
Time: $\cancel{\boldsymbol{O}(\boldsymbol{n}^3)}$ **too slow!**

$$\mathbf{P} \begin{bmatrix} 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 0 \\ 1/4 & 1/4 & 0 & 1/4 & 1/4 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 1/4 & 1/4 & 1/4 & 0 & 1/4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Space: $\boldsymbol{O}(\boldsymbol{m})$

SVD

Time: $\boldsymbol{O}(\boldsymbol{mk}\log(\boldsymbol{n}))$



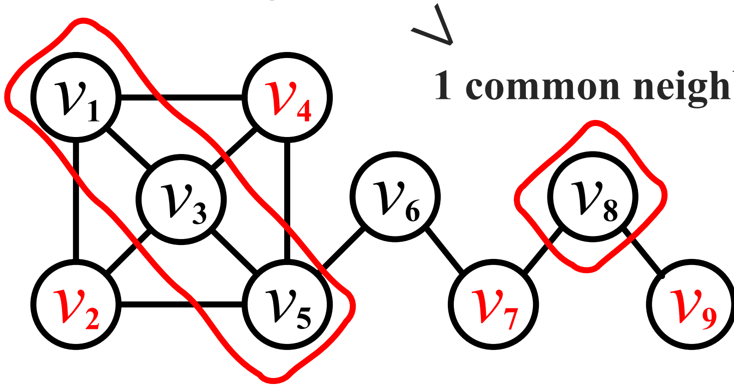
How to refine?

Motivations: Effectiveness

Table 1: PPR for v_2 and v_9 in Fig. 1 ($\alpha = 0.15$).

v_i	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
$\pi(v_2, v_i)$	0.15	0.269	0.188	**0.118**	0.17	0.048	0.029	0.019	0.008
$\pi(v_4, v_i)$	0.15	**0.118**	0.188	0.269	0.17	0.048	0.029	0.019	0.008
$\pi(v_7, v_i)$	0.036	0.043	0.056	0.043	0.093	0.137	0.29	0.187	**0.12**
$\pi(v_9, v_i)$	0.02	0.024	0.031	0.024	0.056	0.083	**0.168**	0.311	0.282

3 common neighbours $>$ **1 common neighbour**



Potential link: $(v_2, v_4) > (v_7, v_9)$

$$\pi(v_2, v_4) + \pi(v_4, v_2) = 0236$$
$$<$$
$$\pi(v_9, v_7) + \pi(v_7, v_9) = 0.288$$

Why?

Neglect node global importance

How?

Reweight nodes with weights !

Proposed solution: NRP

- Basic idea: $\forall v \in V$
  - A forward embedding $\mathbf{X}_v$
  - A backward embedding $\mathbf{Y}_v$
  - A forward weight $\overrightarrow{w}_v$
  - A backward weight $\overleftarrow{w}_v$

Preserve node proximity

Preserve edge direction

$\mathbf{X}_u \cdot \mathbf{Y}_v^{\mathrm{T}} \neq \mathbf{X}_v \cdot \mathbf{Y}_u^{\mathrm{T}}$

Preserve node global importance

$$\mathbf{X}_u \cdot \mathbf{Y}_v^{\mathrm{T}} \approx \overrightarrow{w}_u \cdot \pi(u, v) \cdot \overleftarrow{w}_v$$

- Challenges
  - Approximate PPR $\pi(u, v)$ for all (u, v) pairs efficiently
  - Learn $\overrightarrow{w}_v / \overleftarrow{w}_v$ reflecting node importance

NANYANG TECHNOLOGICAL UNIVERSITY

NRP: Step 1: Approximate PPR

$$\mathbf{P}\begin{bmatrix} 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 0 \\ 1/4 & 1/4 & 0 & 1/4 & 1/4 & 0 & 0 & 0 & 0 \\ 1/3 & 0 & 1/3 & 0 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 1/4 & 1/4 & 1/4 & 0 & 1/4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \xrightarrow[\mathbf{P} \approx \mathbf{X}_1 \cdot \mathbf{Y}^{\mathrm{T}}]{\text{SVD}}$$

$$\mathbf{Y} = \begin{bmatrix} \mathbf{Y}_{v_1} \\ \mathbf{Y}_{v_2} \\ \mathbf{Y}_{v_3} \\ \mathbf{Y}_{v_4} \\ \mathbf{Y}_{v_5} \\ \mathbf{Y}_{v_6} \\ \mathbf{Y}_{v_7} \\ \mathbf{Y}_{v_8} \\ \mathbf{Y}_{v_9} \end{bmatrix} = \begin{bmatrix} -0.652, & 0.243 \\ -0.668, & -0.359 \\ -0.823, & -0.142 \\ -0.668, & -0.359 \\ -0.737, & 0.547 \\ -0.314, & -0.42 \\ -0.105, & 0.633 \\ -0.094, & -0.225 \\ -0.071, & 0.818 \end{bmatrix}, \quad \mathbf{X}_1 = \begin{bmatrix} -0.217, & -0.121 \\ -0.223, & 0.091 \\ -0.206, & 0.008 \\ -0.223, & 0.091 \\ -0.184, & -0.13 \\ -0.157, & 0.4 \\ -0.083, & -0.16 \\ -0.047, & 0.481 \\ -0.032, & -0.034 \end{bmatrix}$$

$O(mk\log(n))$

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_{v_1} \\ \mathbf{X}_{v_2} \\ \mathbf{X}_{v_3} \\ \mathbf{X}_{v_4} \\ \mathbf{X}_{v_5} \\ \mathbf{X}_{v_6} \\ \mathbf{X}_{v_7} \\ \mathbf{X}_{v_8} \\ \mathbf{X}_{v_9} \end{bmatrix} = \begin{bmatrix} -0.182, & -0.014 \\ -0.18, & 0.004 \\ -0.14, & -0.002 \\ -0.18, & 0.004 \\ -0.13, & -0.008 \\ -0.182, & 0.075 \\ -0.126, & 0.072 \\ -0.092, & 0.141 \\ -0.157, & 0.236 \end{bmatrix} = \sum_{t=1}^{\iota_1} \alpha(1-\alpha)^t \mathbf{P}^{t-1}\mathbf{X}_1 \Rightarrow \mathbf{M} \approx \mathbf{X} \cdot \mathbf{Y}^{\mathrm{T}}$$

$O(mk\iota_1)$

NANYANG TECHNOLOGICAL UNIVERSITY

NRP: Step 2: Node Reweighting

- Intuition: 1 total strength of connections from other nodes to u = in-degree of u

  2 total strength of connections from u to other nodes = out-degree of u

- Objective function:

$$O = \min_{\overrightarrow{w}, \overleftarrow{w}} \sum_{v} \left\| \boxed{\sum_{u \neq v} \left(\overrightarrow{w}_u \mathbf{X}_u \mathbf{Y}_v^\top \overleftarrow{w}_v \right) - d_{in}(v)} \right\|_2 \quad 1$$

$$+ \sum_{u} \left\| \boxed{\sum_{v \neq u} \left(\overrightarrow{w}_u \mathbf{X}_u \mathbf{Y}_v^\top \overleftarrow{w}_v \right) - d_{out}(u)} \right\|_2 \quad 2$$

$$+ \lambda \sum_{u} \left(\|\overrightarrow{w}_u\|_2 + \|\overleftarrow{w}_u\|_2 \right),$$

$$\text{subject to } \forall u \in V, \overrightarrow{w}_u, \overleftarrow{w}_u \geq \frac{1}{n}.$$

by coordinate descent in $O(nk^2)$ time

- Output: $\forall v \in V$

$$\mathbf{X}_v \leftarrow \overrightarrow{w}_v \cdot \mathbf{X}_v$$

$$\mathbf{Y}_v \leftarrow \overleftarrow{w}_v \cdot \mathbf{Y}_v$$

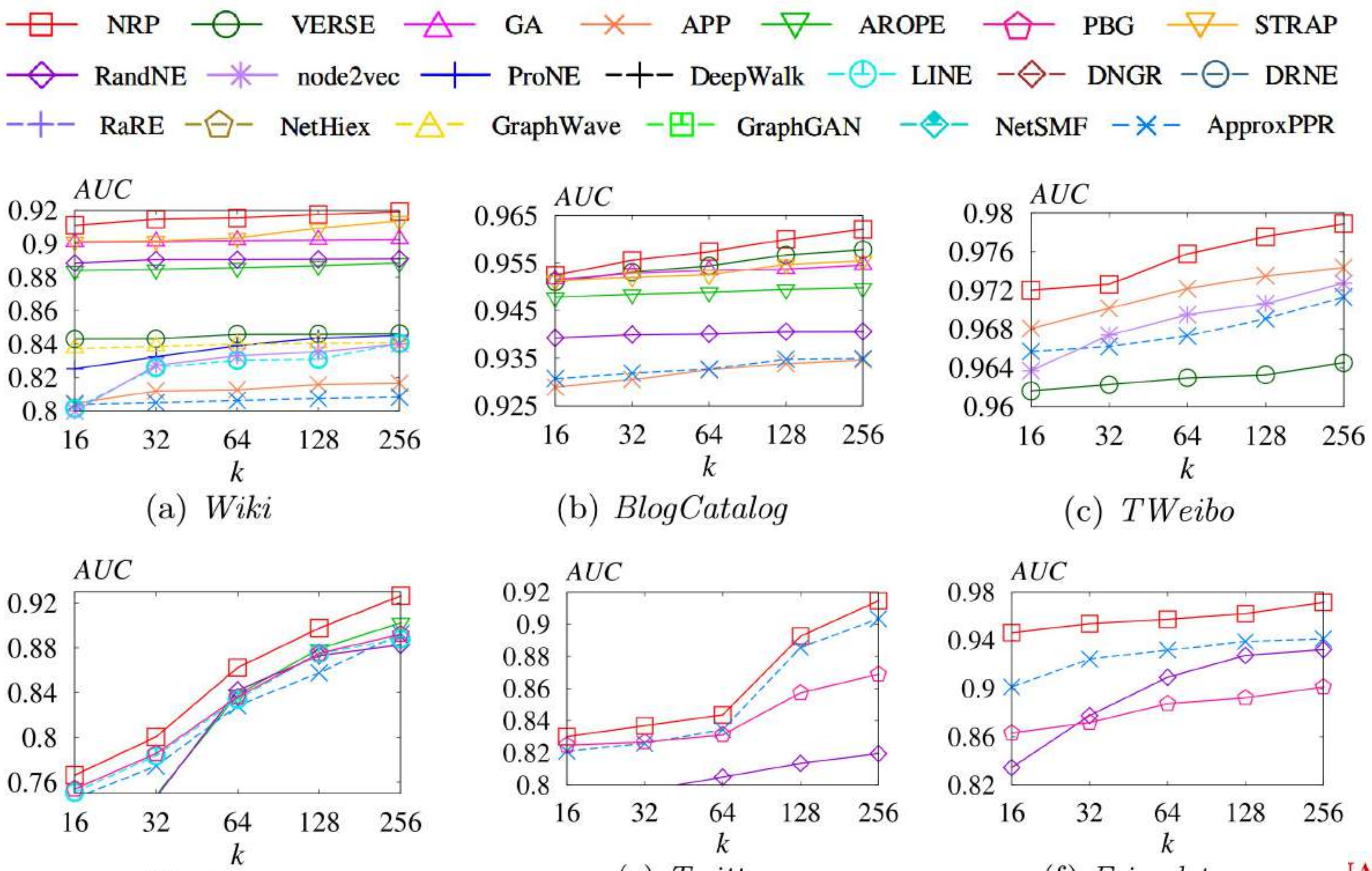
Experiments: Settings

Table 2. Data Sets

Name	$\|V\|$	$\|E\|$	Type	#labels
Wiki	4.78K	184.81K	directed	40
BlogCatalog	10.31K	333.98K	undirected	39
Youtube	1.13M	2.99M	undirected	47
TWeibo	2.32M	50.65M	directed	100
Orkut	3.1M	234M	undirected	100
Twitter	41.6M	1.2B	directed	-
Friendster	65.6M	1.8B	undirected	-

- NRP: $k = 128, \iota_1 = 20, \iota_2 = 10, \alpha = 0.15$
- ApproxPPR: $k = 128, \iota_1 = 20, \alpha = 0.15$ (without reweighting)
- an Intel Xeon(R) E5-2650 v2@2.60GHz CPU and 96GB RAM

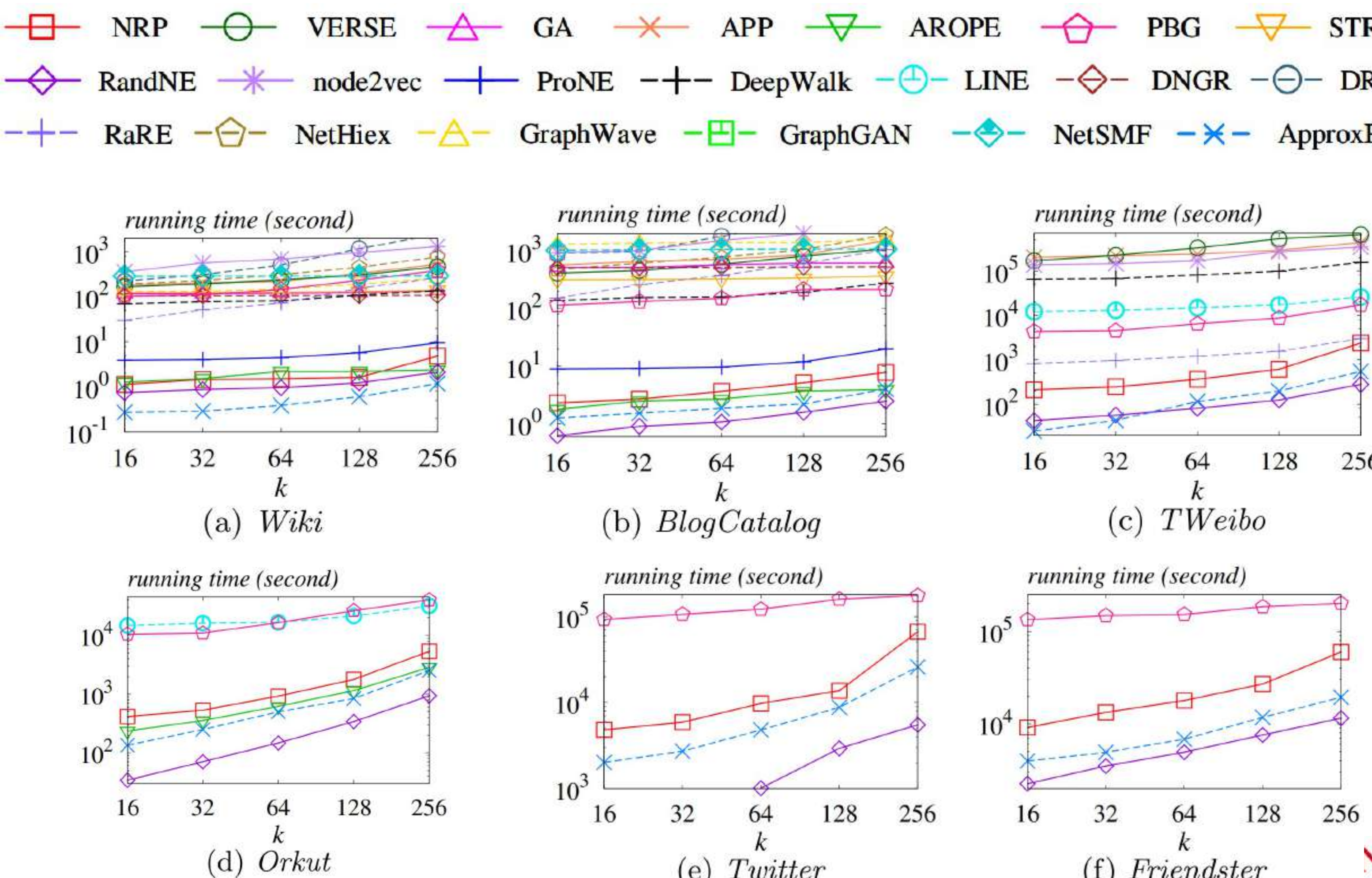
Experiments: Link Prediction



(a) *Wiki* (b) *BlogCatalog* (c) *TWeibo*

(d) *Orkut* (e) *Twitter* (f) *Friendster*

Experiments: Efficiency



(a) *Wiki* (b) *BlogCatalog* (c) *TWeibo*

(d) *Orkut* (e) *Twitter* (f) *Friendster*

Thanks

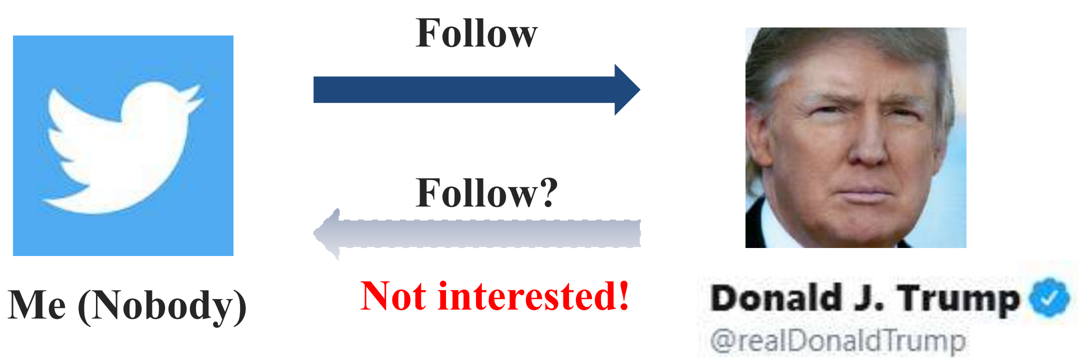
Q & A

NANYANG
TECHNOLOGICAL
UNIVERSITY

Why NRP Works

NRP preserves

- Multi-hop proximity between nodes (PPR)
- The global importance of nodes (Reweighting)
- Edge directions (forward/backward embeddings)
  - For example



NANYANG TECHNOLOGICAL UNIVERSITY

Competitors

- Factorization-based
  - AROPE, RandNE, NetSMF, ProNE, STRAP
- Random-walk-based
  - DeepWalk, LINE, node2vec, PBG, APP, VERSE
- Neural-network-based
  - DNGR, DRNE, GraphGAN, GA
- Other
  - RaRE, NetHiex, GraphWave

NANYANG TECHNOLOGICAL UNIVERSITY

Experiments: Graph Reconstruction

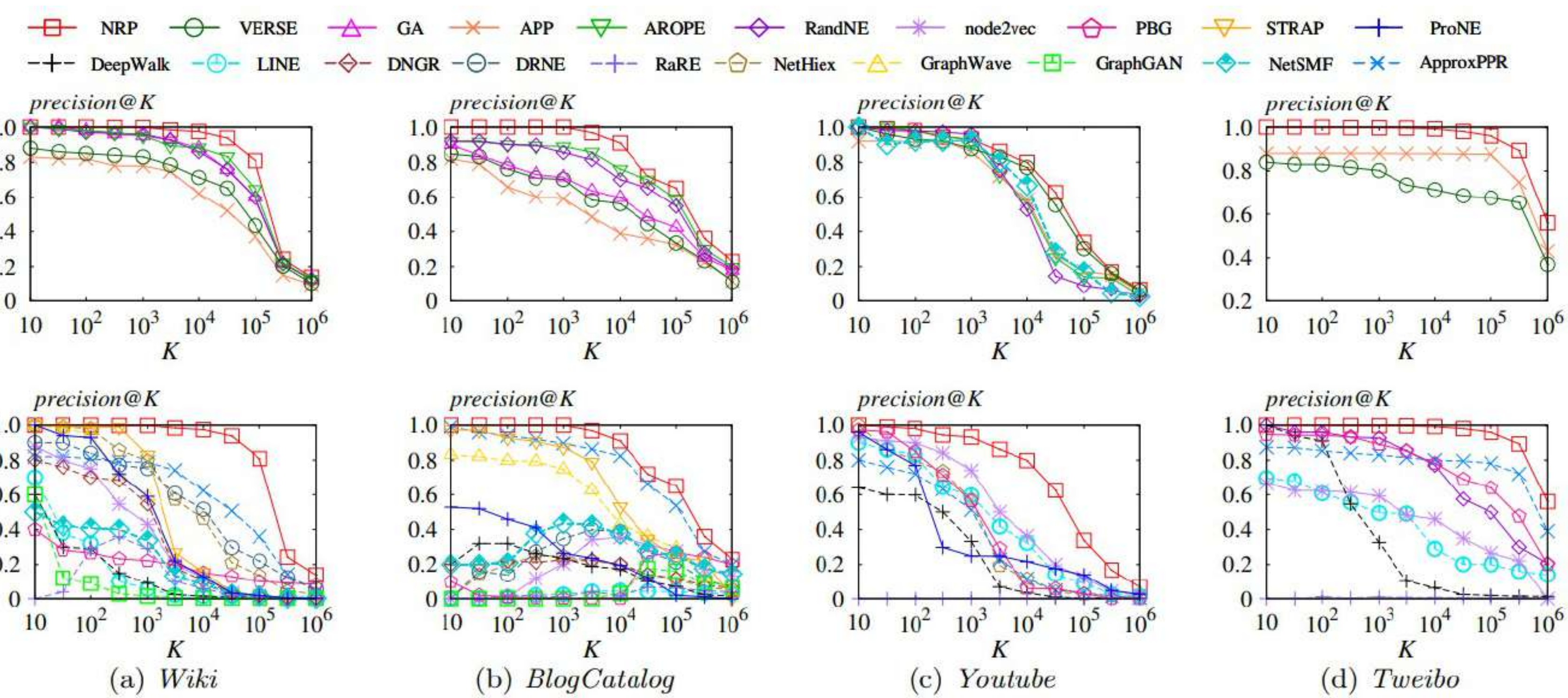


Figure 5: Graph reconstruction results vs. K (best viewed in color).

NANYANG TECHNOLOGICAL UNIVERSITY

Experiments: Node Classification

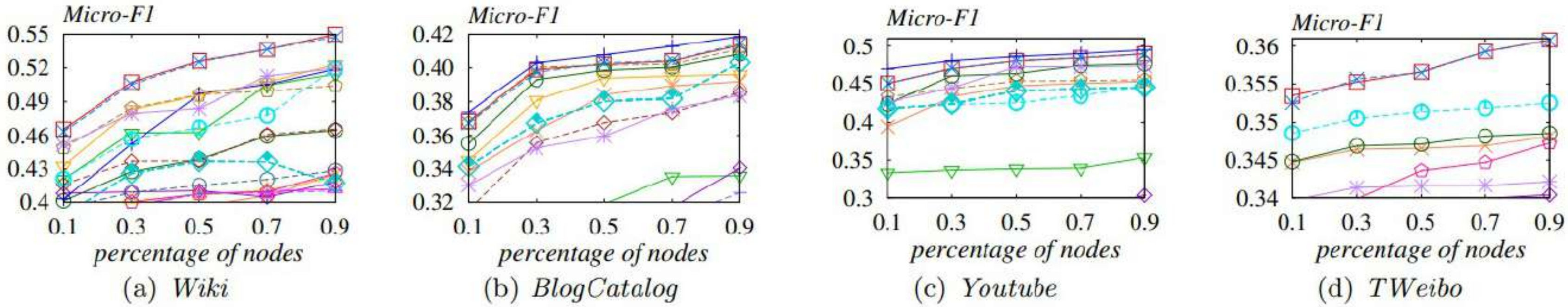


Figure 6: Node classification results (best viewed in color).

Experiments: Parameter Analysis

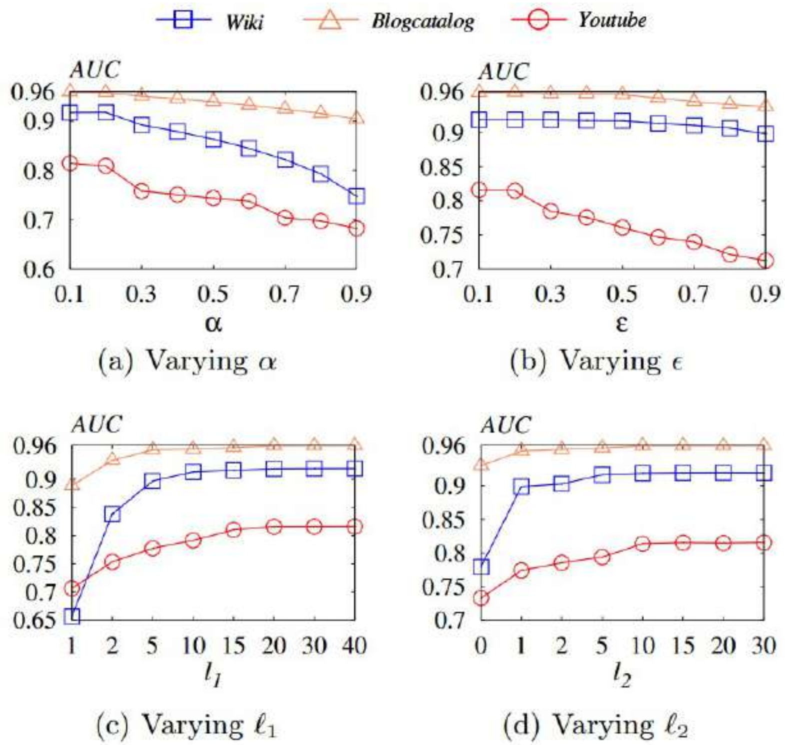


Figure 8. Link prediction results

Experiments: Link Prediction on Dynamic Graphs

Name	$\|V\|$	$\|E\|$	$\|E_{old}\|$	$\|E_{new}\|$	Type
VK	78.59K	5.35M	2.68M	2.67M	undirected
Digg	279.63K	1.73M	1.03M	701.59K	directed

Table 4: Dataset statistics ($K = 10^3$, $M = 10^6$).

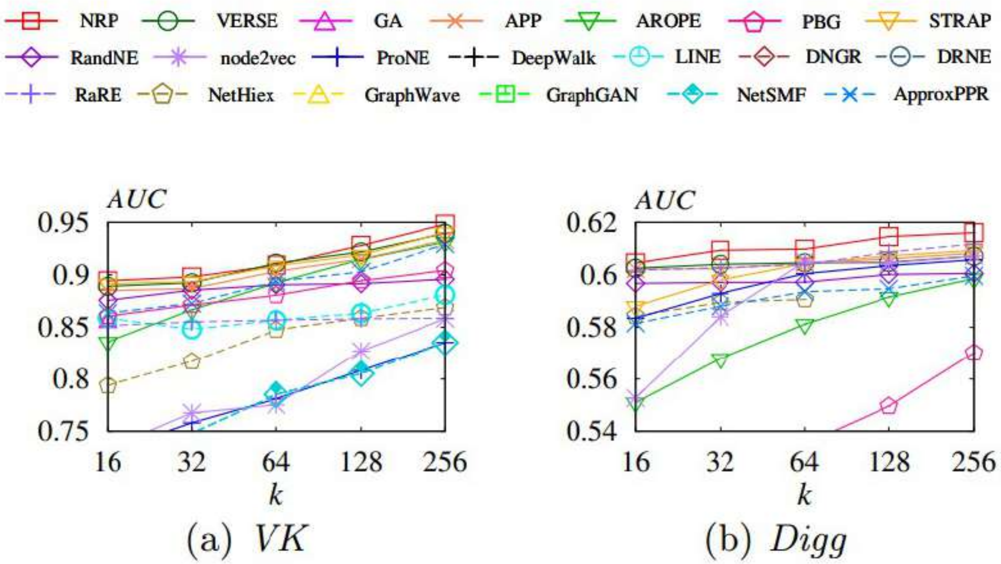
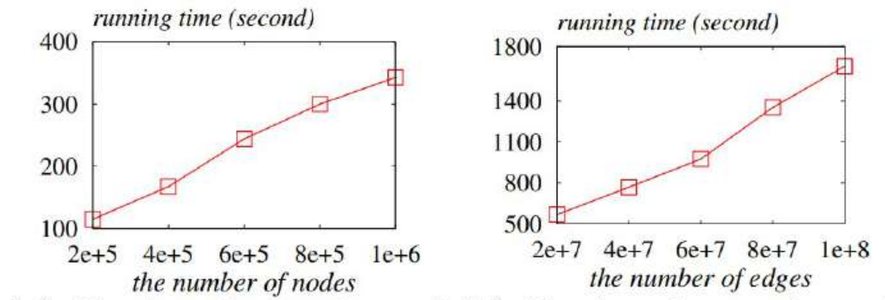


Figure 9: Link prediction performance on dynamic graphs (best viewed in color).

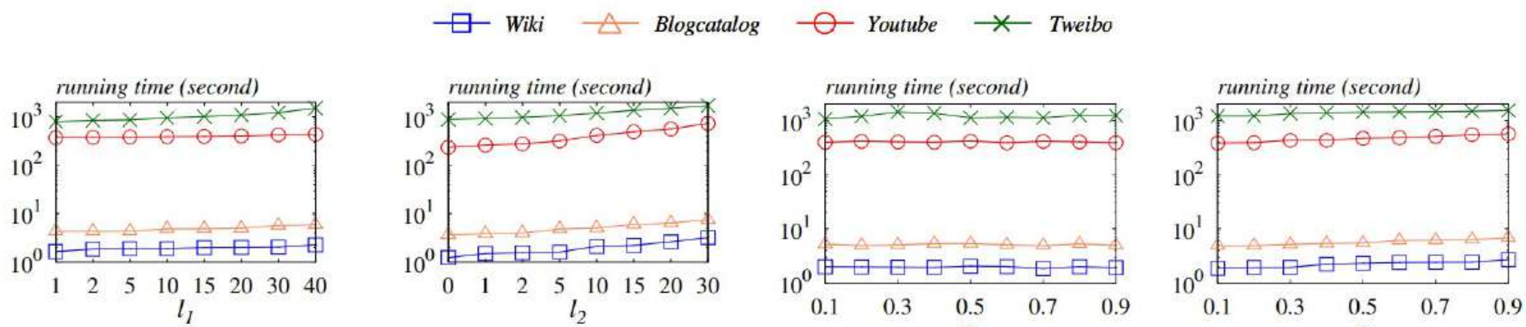
NANYANG TECHNOLOGICAL UNIVERSITY

Experiments: Efficiency



(a) Varying the number of nodes (b) Varying the number of edges

Figure 10: Scalability tests.



(a) Varying ℓ_1 (b) Varying ℓ_2 (c) Varying α (d) Varying ϵ

Figure 11: Running time with varying parameters (best viewed in color).

NANYANG TECHNOLOGICAL UNIVERSITY

PPR Approximation

Algorithm 1: ApproxPPR

Input: $\mathbf{A}$, $\mathbf{D}^{-1}$, $\mathbf{P}$, α, k', ℓ_1, ϵ.

Output: $\mathbf{X}$, $\mathbf{Y}$.

$[\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}] \leftarrow \mathsf{BKSVD}(\mathbf{A}, k', \epsilon)$;

$\mathbf{X}_1 \leftarrow \mathbf{D}^{-1}\mathbf{U}\sqrt{\mathbf{\Sigma}}, \quad \mathbf{Y} \leftarrow \mathbf{V}\sqrt{\mathbf{\Sigma}}$;

for $i \leftarrow 2$ *to* ℓ_1 **do**

  $\mathbf{X}_i \leftarrow (1-\alpha)\mathbf{P}\mathbf{X}_{i-1} + \mathbf{X}_1$;

$\mathbf{X} \leftarrow \alpha(1-\alpha)\mathbf{X}_{\ell_1}$;

return $\mathbf{X}$, $\mathbf{Y}$;

Node Reweighting

Algorithm 2: updateBwdWeights

Input: G, k', $\overrightarrow{\mathbf{w}}$, $\overleftarrow{\mathbf{w}}$, $\mathbf{X}$, $\mathbf{Y}$.
Output: $\overleftarrow{\mathbf{w}}$

Compute $\boldsymbol{\xi}, \boldsymbol{\chi}, \boldsymbol{\rho_1}, \boldsymbol{\rho_2}, \boldsymbol{\Lambda}$, and $\boldsymbol{\Phi}$ based on Eq. (9), (10), and (13);
for $r \leftarrow 1$ *to* k' **do**
  $\phi[r] = \sum_u \overrightarrow{w}_u^2 \mathbf{X}_u[r]^2$;
for $v^* \in V$ *in random order* **do**
  Compute a_1, a_2, a_3, b_1, b_2 by Eq. (9), (10), and (14);
  $\overleftarrow{w}'_{v^*} = \overleftarrow{w}_{v^*}$;
  $\overleftarrow{w}_{v^*} = \max\left\{\frac{1}{n}, \frac{a_1+a_2-a_3}{b_1+b_2+\lambda}\right\}$;
  $\boldsymbol{\rho}_1 = \boldsymbol{\rho}_1 + \left(\overleftarrow{w}_{v^*} - \overleftarrow{w}'_{v^*}\right)\mathbf{Y}_{v^*}$;
  $\boldsymbol{\rho}_2 = \boldsymbol{\rho}_2 + \left(\overleftarrow{w}_{v^*} - \overleftarrow{w}'_{v^*}\right)\overrightarrow{w}^2_{v^*}\left(\mathbf{X}_{v^*}\mathbf{Y}^\top_{v^*}\right)\mathbf{X}_{v^*}$
return $\overleftarrow{\mathbf{w}}$;

Algorithm 4: updateFwdWeights

Input: G, k', $\overrightarrow{\mathbf{w}}$, $\overleftarrow{\mathbf{w}}$, $\mathbf{X}$, $\mathbf{Y}$.
Output: $\overrightarrow{\mathbf{w}}$

Compute $\boldsymbol{\xi}, \boldsymbol{\chi}, \boldsymbol{\rho_1}, \boldsymbol{\rho_2}, \boldsymbol{\Lambda}$ based on Eq. (24), (25);
for $r \leftarrow 1$ *to* k' **do**
  $\phi[r] = \sum_v \overleftarrow{w}_v^2 \mathbf{Y}_v[r]^2$;
for $u^* \in V$ *in random order* **do**
  Compute $a'_1, a'_2, a'_3, b'_1, b'_2$ by Eq. (24), (25), and (29);
  $\overrightarrow{w}'_{u^*} = \overrightarrow{w}_{u^*}$;
  $\overrightarrow{w}_{u^*} = \max\left\{\frac{1}{n}, \frac{a'_1+a'_2-a'_3}{b'_1+b'_2+\lambda}\right\}$;
  $\boldsymbol{\rho}_1 = \boldsymbol{\rho}_1 + \left(\overrightarrow{w}_{u^*} - \overrightarrow{w}'_{u^*}\right)\mathbf{X}_{u^*}$;
  $\boldsymbol{\rho}_2 = \boldsymbol{\rho}_2 + \left(\overrightarrow{w}_{u^*} - \overrightarrow{w}'_{u^*}\right)\overleftarrow{w}^2_{u^*}\left(\mathbf{X}_{u^*}\mathbf{Y}^\top_{u^*}\right)\mathbf{Y}_{u^*}$
return $\overrightarrow{\mathbf{w}}$;

NANYANG TECHNOLOGICAL UNIVERSITY

NRP

Algorithm 3: NRP

Input: Graph G, embedding dimensionality k, thresholds ℓ_1, ℓ_2, random walk decay factor α and error threshold ϵ

Output: Embedding matrices $\mathbf{X}$ and $\mathbf{Y}$.

$k' \leftarrow k/2$;

$[\mathbf{X}, \mathbf{Y}] \leftarrow \mathsf{ApproxPPR}(\mathbf{A}, \mathbf{D}^{-1}, \mathbf{P}, \alpha, k', \ell_1, \epsilon)$;

for $v \in V$ **do**

  $\overrightarrow{w}_v = d_{out}(v),\ \overleftarrow{w}_v = 1$;

for $l \leftarrow 1$ *to* ℓ_2 **do**

  $\overleftarrow{\mathbf{w}} = \mathsf{updateBwdWeights}(G, k', \overrightarrow{\mathbf{w}}, \overleftarrow{\mathbf{w}}, \mathbf{X}, \mathbf{Y})$;

  $\overrightarrow{\mathbf{w}} = \mathsf{updateFwdWeights}(G, k', \overrightarrow{\mathbf{w}}, \overleftarrow{\mathbf{w}}, \mathbf{X}, \mathbf{Y})$;

for $v \in V$ **do**

  $\mathbf{X}_v = \overrightarrow{w}_v \cdot \mathbf{X}_v,\quad \mathbf{Y}_v = \overleftarrow{w}_v \cdot \mathbf{Y}_v$;

return $\mathbf{X}$, $\mathbf{Y}$;