Dynamic Network Embedding

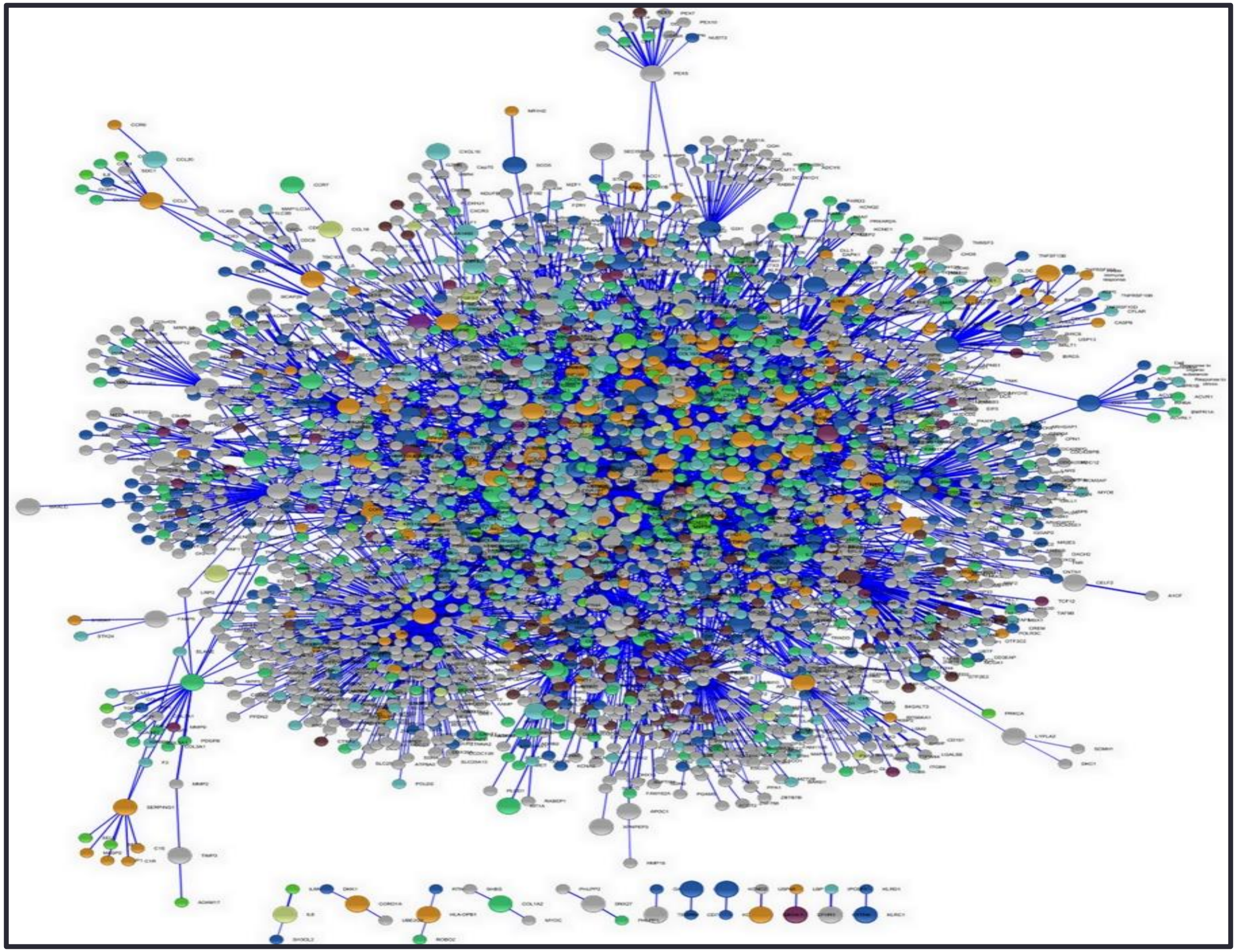
Peng Cui
Tsinghua University

Networks are ubiquitous

Social Networks

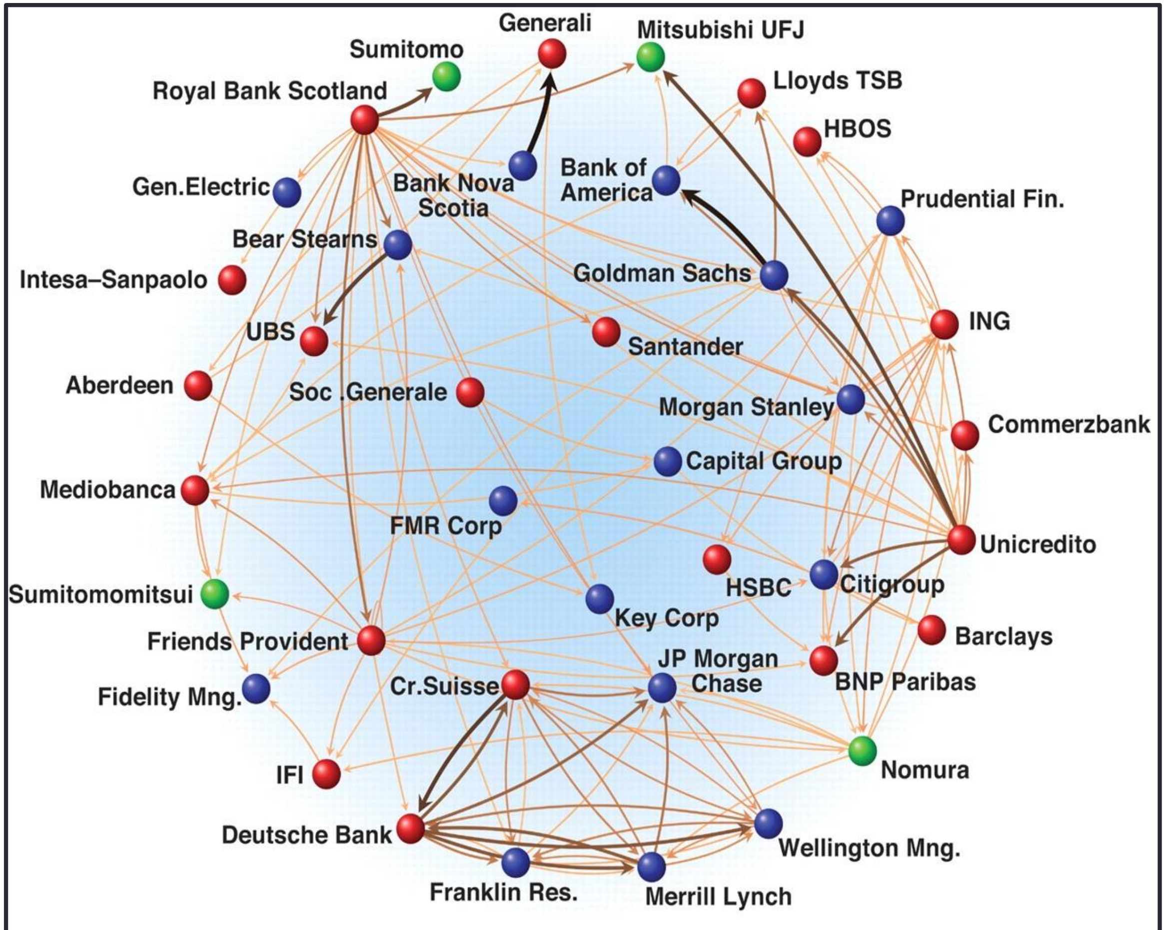
Networks are ubiquitous

Biology Networks



Networks are ubiquitous

Finance Networks

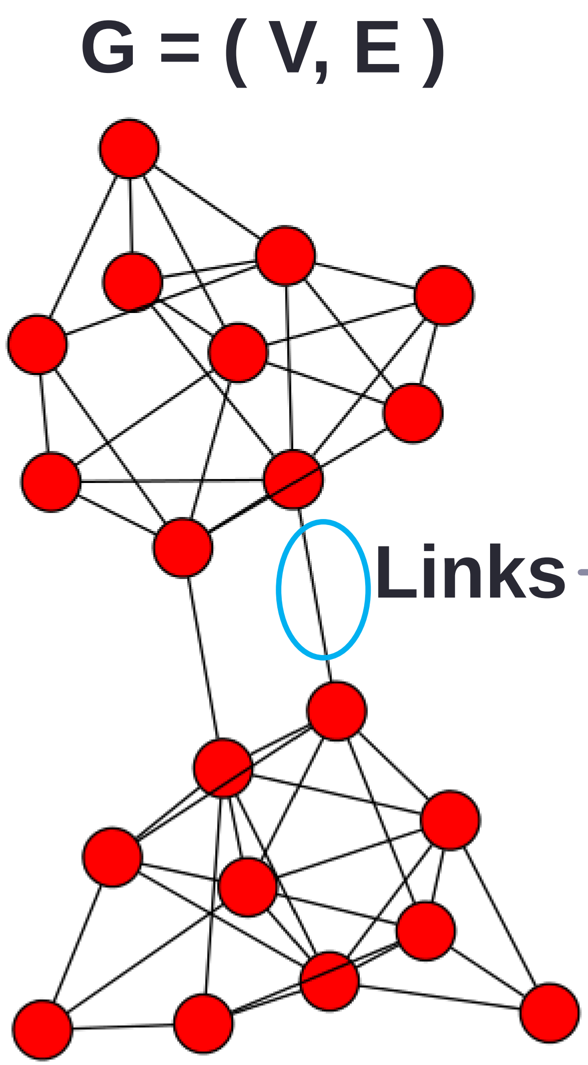


Sample of the international financial network
Credits: ETH Zurich

Networks are ubiquitous

Internet of Things

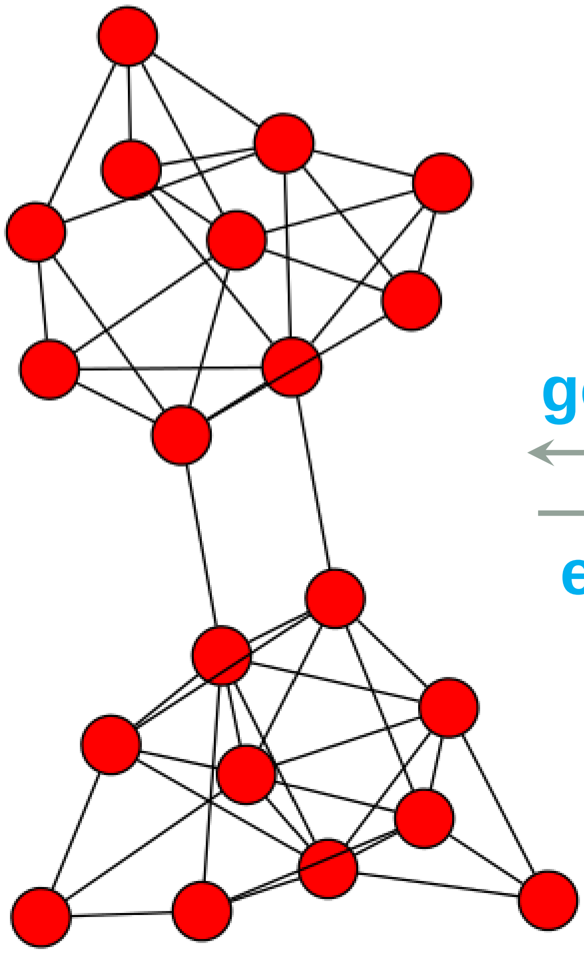
What is the bottleneck?



- Iterative & Combinatorial → **Complexity**
- Coupling → **Parallelizability**
- Dependency among nodes → **Applicability of ML methods**

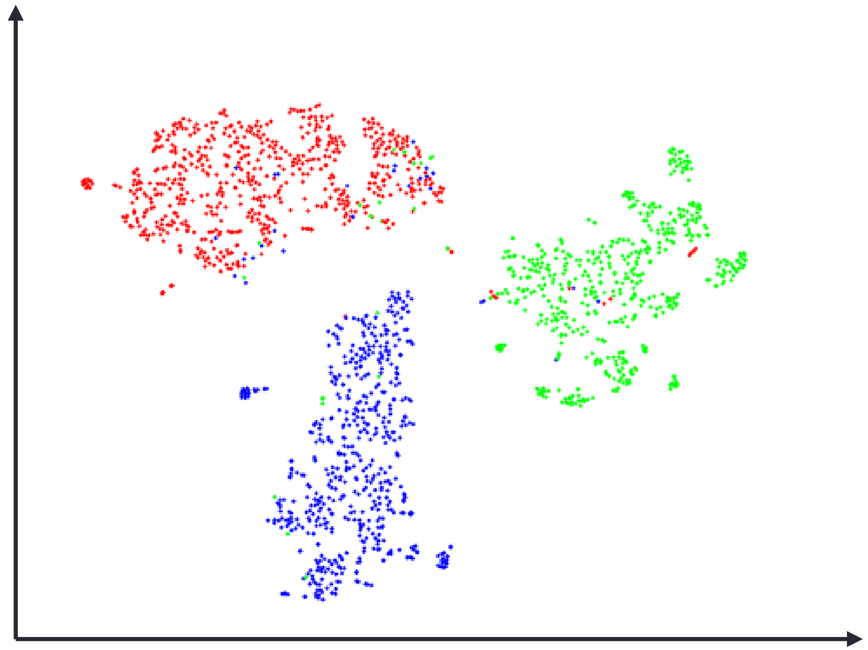
Revisit network representation

G = (V, E)



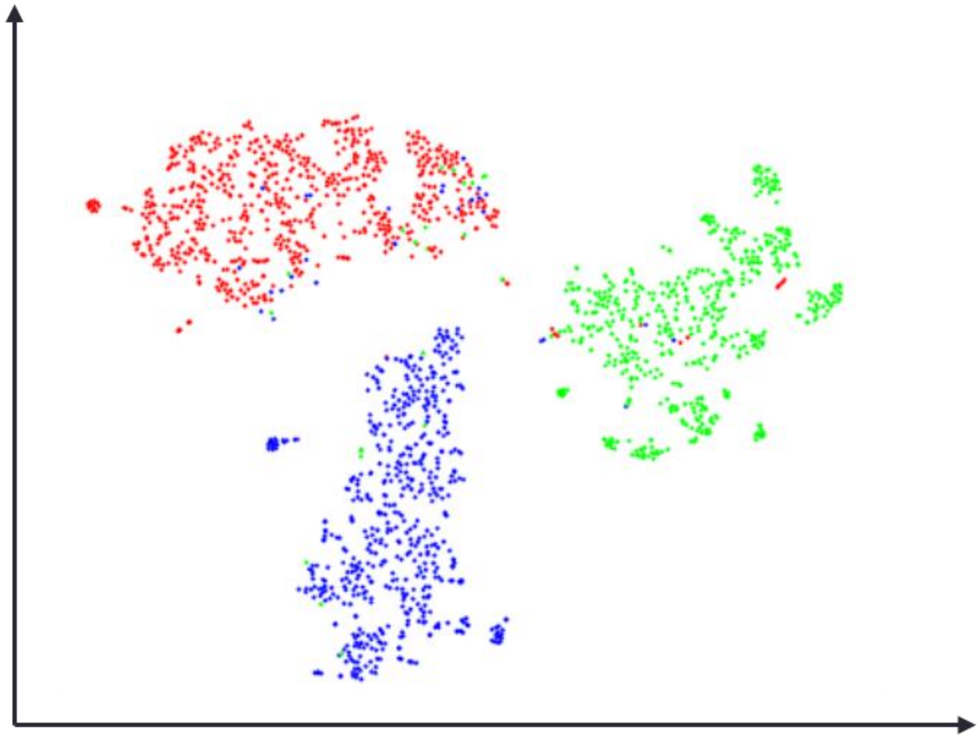
G = (V)

Vector Space



- ☐ Easy to parallel
- ☐ Can apply classical ML methods

The ultimate goal of network embedding



Network Inference

- Node importance
- Community detection
- Network distance
- Link prediction
- Node classification
- Network evolution
- …

in *Vector Space*

How?

The vector space should be able to…

Goal 1

Reconstruct the original network

Goal 2

Support network inference

↓

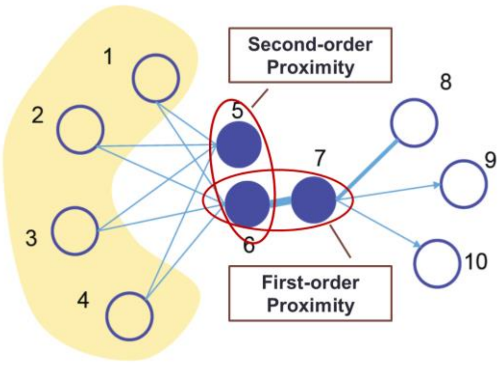
Network Embedding

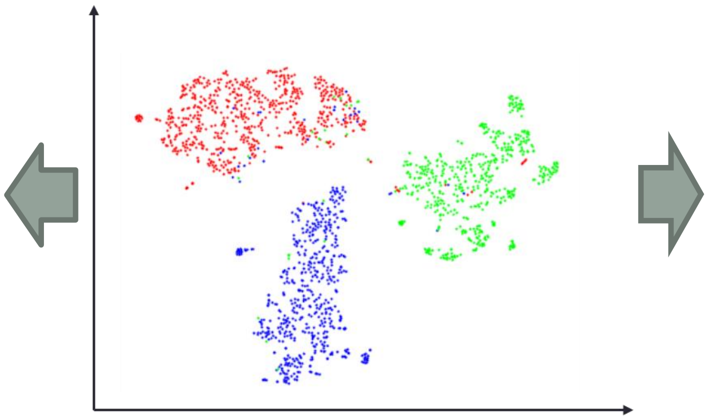
Network Embedding

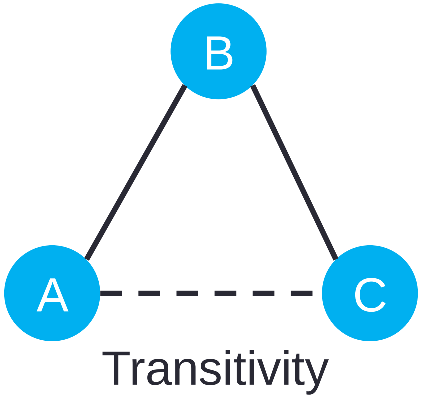
Goal 2 Support network inference

Reflect network structure

Maintain network Properties







A Survey on Network Embedding

IEEE TRANSACTIONS ON
KNOWLEDGE AND DATA ENGINEERING

A Survey on Network Embedding

Issue No. 01 - (preprint vol.)
ISSN: 1041-4347
pp: 1
DOI Bookmark: http://doi.ieeecomputersociety.org/10.1109/TKDE.2018.2849727

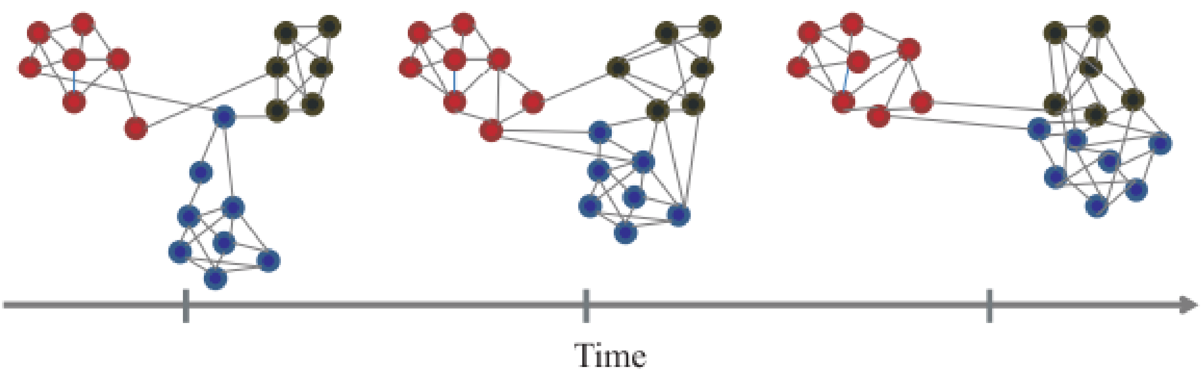
Peng Cui , Computer Science Department, Tsinghua University, Beijing, Beijing China (e-mail: cuip@tsinghua.edu.cn)
Xiao Wang , Computer Science, Tsinghua University, Beijing, Beijing China (e-mail: wangxiao007@mail.tsinghua.edu.cn)
Jian Pei , School of Computing Science, Simon Fraser Univeristy, Burnaby, British Columbia Canada (e-mail: jpei@cs.sfu.ca)
Wenwu Zhu , Department of Computer Science, Tsinghua Univerisity, Beijing, Beijing China (e-mail: wwzhu@tsinghua.edu.cn)

ABSTRACT
Network embedding assigns nodes in a network to low-dimensional representations and effectively preserves the network structure. Recently, a significant amount of progresses have been made toward this emerging network analysis paradigm. In this survey, we focus on categorizing and then reviewing the current development on network embedding methods, and point out its future research directions. We first summarize the motivation of network embedding. We discuss the classical graph embedding algorithms and their relationship with network embedding. Afterwards and primarily, we provide a comprehensive overview of a large number of network embedding methods in a systematic manner, covering the structure- and property-preserving network embedding methods, the network embedding methods with side information and the advanced information preserving network embedding methods. Moreover, several evaluation approaches for network embedding and some useful online resources, including the network data sets and softwares, are reviewed, too. Finally, we discuss the framework of exploiting these network embedding methods to build an effective system and point out some potential future directions.

Peng Cui, Xiao Wang, Jian Pei, Wenwu Zhu. **A Survey on Network Embedding**. *IEEE TKDE, 2018*.

Dynamic Networks

- **Networks are dynamic in nature**
  - **New (old) nodes are added (deleted)**
    - New users, products, etc.
  - **The edges between nodes evolve over time**
    - Users add or delete friends in social networks, or neurons establish new connections in brain networks.
- **How to efficiently incorporate the dynamic changes when networks evolve?**

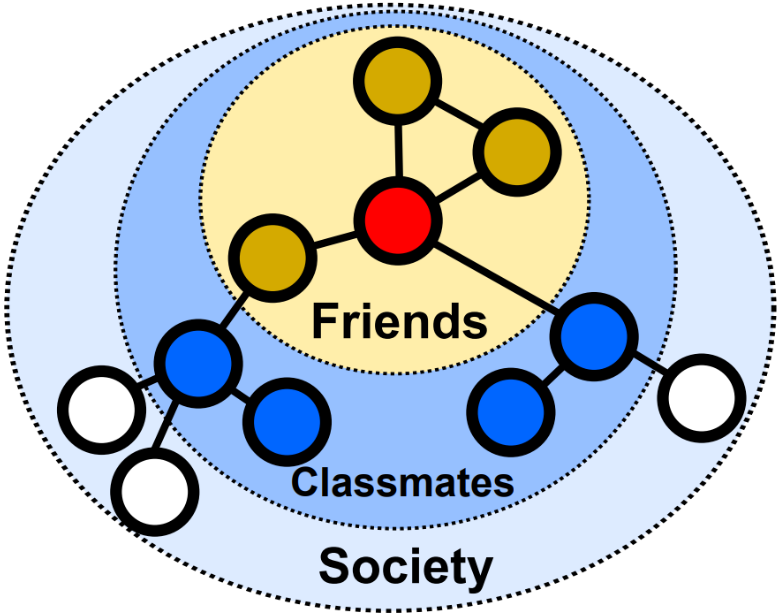


Key problems in dynamic network embedding

- **I : Out-of-sample nodes**
- **II : Incremental edges**
- **III: Aggregated error**
- **IV: Scalable optimization**

Challenge: High-order Proximity

- **High-order proximity**
  - Critical structural property of networks
  - Measure indirect relationship between nodes
  - Capture the structure of networks with different scales and sparsity



Network Embedding V.S. Traditional Graph Embedding

Challenge: High-order Proximity

I : Out-of-sample nodes

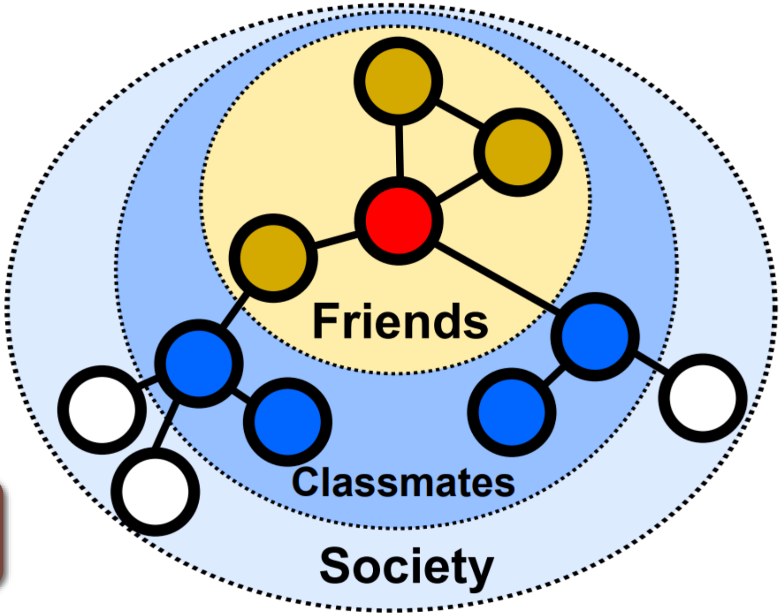
II : Incrementally updating

III: Aggregated error

IV: Scalable optimization

Preserve High-order Proximities

Local Change leads to Global Updating

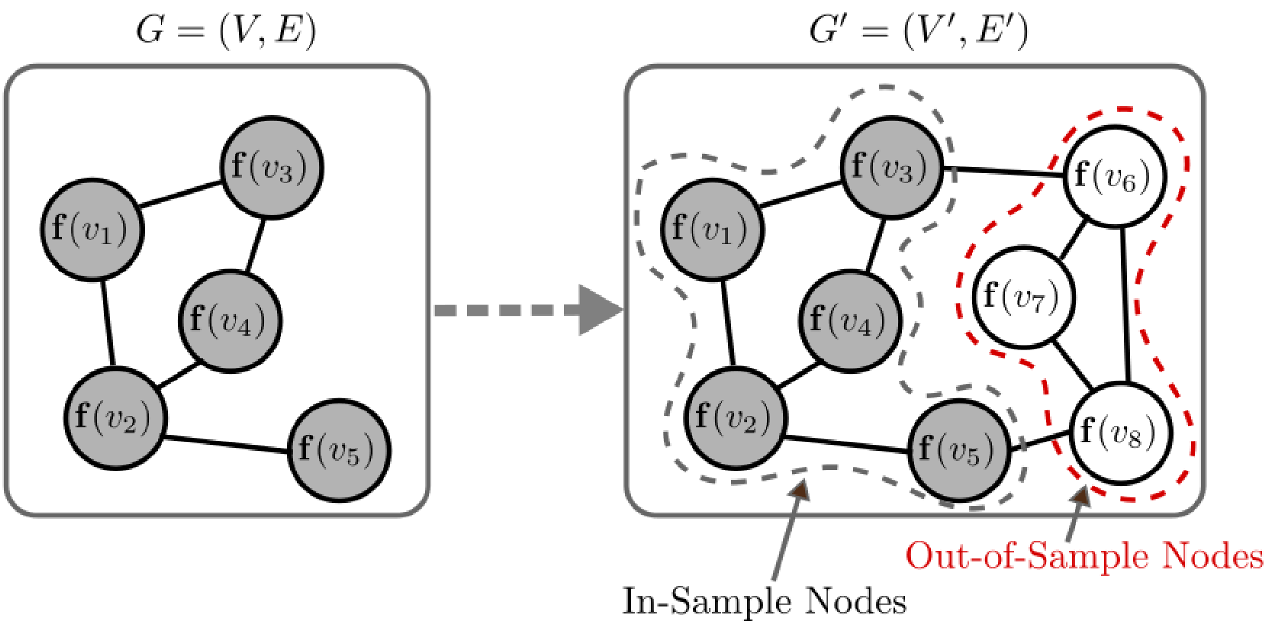


Key problems in dynamic network embedding

- **I : Out-of-sample nodes**
- **II : Incremental edges**
- **III: Aggregated error**
- **IV: Scalable optimization**

Problem

- To infer embeddings for out-of-sample nodes.



- *G=(V, E)* evolves into *G'=(V', E')*, where $V' = V \cup V^*$.
  - n old nodes: $V = \{v_1,\ldots,v_n\}$, m new nodes: $V^* = \{v_{n+1},\ldots,v_{n+m}\}$

- Network embedding: $\boldsymbol{f}: V \rightarrow R^d$
  - We know $\boldsymbol{f}(v)$ for old nodes, want to infer $\boldsymbol{f}(v)$ for new nodes.

Challenges

- Preserve network structures
  - e.g. high-order proximity
  - need to incorporate prior knowledge on networks
- Share similar characteristics with in-sample embeddings
  - e.g. magnitude, mean, variance
  - requires a model with great expressive power to fit the data well
- Low computational cost

Specific vs. General

- Specific
  - A new NE algorithm capable of handling OOS nodes.
- General
  - A solution that helps an arbitrary NE algorithm handle OOS nodes.

- We propose a **general** solution.
  - But it can be easily integrated into an existing NE algorithm (e.g. DeepWalk) to derive a **specific** algorithm (see the paper).

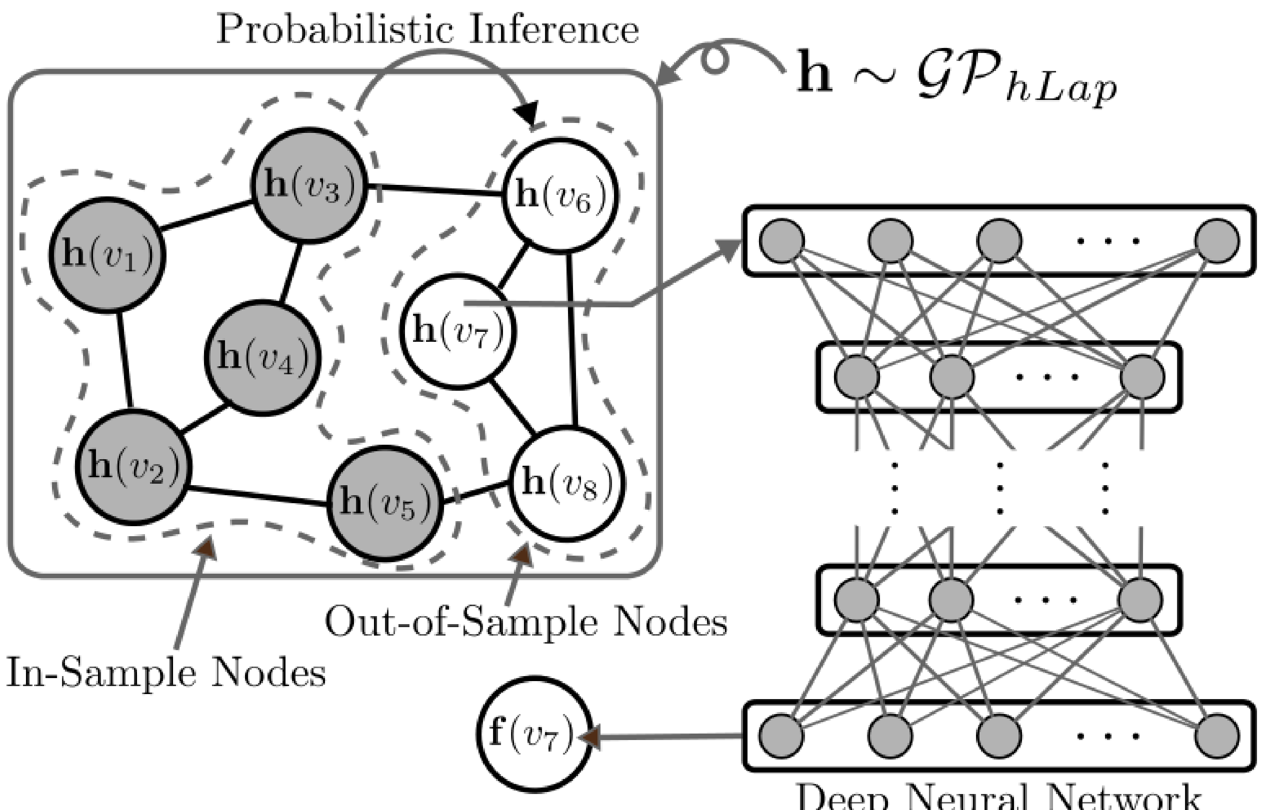
Iterative vs. Analytical

- Iterative
  - Some algorithms (e.g. DeepWalk) are trained iteratively. It is possible to incrementally learn embeddings for OOS nodes.
  - Disadvantage: Must find the proper number of iterations with cross validation. -> Slow and laborious.
- Analytical
  - …

- We want a fast **analytical** prediction procedure.
  - … so that we can handle OOS nodes without much effort.
  - The training procedure can be **iterative**, but it must be finished before new nodes even arrive.

DepthLGP

- Nonparametric probabilistic modeling + Deep Learning



Jianxin Ma, **Peng Cui**, Wenwu Zhu. DepthLGP: Learning Embeddings of Out-of-Sample Nodes in Dynamic Networks. *AAAI*, 2018.

Theories

- The model can fit arbitrary network embedding.
  - There exists such a set of parameters, such that: …

Theorem 1 (Expressive Power). *For any* $\epsilon > 0$, *any non-trivial* $G = (V, E)$ *and any* $\mathbf{f} : \mathcal{V} \to \mathbb{R}^d$, *there exists a parameter setting for DepthLGP, such that: for any* $v^* \in V$, *after deleting all information (except* G*) related with* v^*, *DepthLGP can still recover* $\mathbf{f}(v^*)$ *with error less than* ϵ, *by treating* v^* *as a new node and using Algorithm 1 on* G.

Task I: Classification

Metric	Embedding	Network	Baselines			This Work		Upper Bound
			LocalAvg	MRG	LabelProp	hLGP	DepthLGP	(rerunning)
Macro-F1(%)	LINE	DBLP	37.89	42.15	40.83	47.33	**48.25**	(49.07)
		PPI	10.52	10.02	12.42	13.42	**13.72**	(13.91)
		BlogCatalog	13.25	11.30	17.07	17.41	**18.03**	(18.90)
	GraRep	DBLP	50.61	55.79	55.02	57.43	**58.67**	(62.92)
		PPI	13.65	13.75	12.38	14.80	**14.84**	(15.33)
		BlogCatalog	14.76	14.80	14.71	15.94	**18.45**	(20.15)
	node2vec	DBLP	53.83	59.34	59.25	60.89	**62.63**	(64.87)
		PPI	15.05	13.43	13.78	15.85	**16.54**	(16.81)
		BlogCatalog	15.10	14.04	19.16	19.77	**20.32**	(20.82)
Micro-F1(%)	LINE	DBLP	49.58	50.49	50.88	54.01	**54.94**	(55.84)
		PPI	18.10	15.71	18.81	20.71	**21.42**	(21.43)
		BlogCatalog	27.40	23.21	30.79	31.36	**31.90**	(32.20)
	GraRep	DBLP	60.17	60.62	60.48	61.44	**62.29**	(65.44)
		PPI	20.23	20.35	20.23	20.79	**21.44**	(21.88)
		BlogCatalog	36.44	30.79	33.90	37.57	**38.14**	(38.37)
	node2vec	DBLP	60.54	62.29	62.52	62.83	**64.56**	(65.63)
		PPI	19.70	18.25	18.25	22.63	**23.11**	(23.41)
		BlogCatalog	34.83	25.82	36.94	37.96	**39.64**	(40.34)

Task II: Link Prediction

			Baselines			This Work		Upper Bound
Metric	Embedding	Network	LocalAvg	MRG	LabelProp	hLGP	DepthLGP	(rerunning)
AUC(%)	LINE	DBLP	72.87	72.87	77.39	80.63	**81.18**	(82.33)
		PPI	52.34	51.78	52.77	57.04	**60.45**	(60.57)
		BlogCatalog	55.51	51.01	54.71	54.74	**55.53**	(55.76)
	GraRep	DBLP	84.15	85.88	86.32	87.25	**87.40**	(91.95)
		PPI	62.80	68.55	66.48	67.60	**68.85**	(69.61)
		BlogCatalog	45.60	41.24	47.29	47.42	**48.11**	(48.26)
	node2vec	DBLP	68.49	76.90	77.98	81.36	**82.54**	(89.02)
		PPI	38.90	40.54	46.79	53.16	**55.37**	(59.74)
		BlogCatalog	54.65	38.41	55.40	55.43	**55.47**	(55.86)

Key problems in dynamic network embedding

- I : Out-of-sample nodes
- **II : Incremental edges**
- III: Aggregated error
- IV: Scalable optimization

Problem Formulation

- Suppose that we have learned the node embedding based on the edges appearing before time t.

- How to efficiently update these node embedding at time t+△t so that the changed network structure caused by the newly added/deleted edges during △t can be reflected by the updated node embedding?

The Static Model

- We aim to preserve high-order proximity in the embedding matrix with the following objective function：

$$\min ||\mathbf{S} - \mathbf{U}\mathbf{U}'^{\top}||_F^2$$

  - where S denotes the high-order proximity matrix of the network
  - U and U’ is the results of matrix decomposition of S.

- For undirected networks, U and U’ are highly correlated.
  - Without loss of generality, we choose U as the embedding matrix.

GSVD

- We choose Katz Index as S because it is one of the most widely used measures of high-order proximity. It can be formulated as:

$$\mathbf{S}^{Katz} = \mathbf{M}_a{}^{-1}\mathbf{M}_b$$
$$\mathbf{M}_a = (\mathbf{I} - \beta\mathbf{A})$$
$$\mathbf{M}_b = \beta\mathbf{A}$$

  - where β is a decay parameter, **I** is the identity matrix and **A** is the adjacency matrix

- According to HOPE, the original objective function can be solved by the generalized SVD (GSVD) method

Mingdong Ou, **Peng Cui**, Jian Pei, Wenwu Zhu. Asymmetric Transitivity Preserving Graph Embedding. *KDD*, 2016.

Problem Transformation

- For static model, we get the GSVD results of the high-order proximity matrix as the embedding of nodes.
- But it is difficult to incremental updating the GSVD results directly.
- Here we propose to transform the GSVD problem into generalized eigenvalue problem, so that the incremental updating is feasible.

Dingyuan Zhu, **Peng Cui**, Ziwei Zhang, Jian Pei, Wenwu Zhu. High-order Proximity Preserved Embedding For Dynamic Networks. *IEEE TKDE*, 2018.

GSVD → Generalized Eigen Problem

- Formally, GSVD can be transformed into the generalized eigenvalue problem as:

$$\mathbf{M}_a{}^{-1}\mathbf{M}_b\mathbf{X} = \mathbf{\Lambda}\mathbf{X}$$
$$\mathbf{\Lambda} = diag(\lambda_1, \lambda_2, ..., \lambda_N)$$
$$\lambda_i = \sigma_i \cdot sgn(\mathbf{v}_i^l \cdot \mathbf{v}_i^r)$$
$$\mathbf{X} = \mathbf{V}^l$$

  - where λ_i are the eigenvalues of **S** in descending order, and **X** is a matrix which contains the corresponding eigenvectors of λ_i and sgn() is the Sign function.

Dingyuan Zhu, **Peng Cui**, Ziwei Zhang, Jian Pei, Wenwu Zhu. High-order Proximity Preserved Embedding For Dynamic Networks. *IEEE TKDE*, 2018.

Generalized Eigen Problem → GSVD

- And the results of the generalized eigenvalue problem can also be transformed back into the results of GSVD problem:

$$\mathbf{v}_i^l = \mathbf{x}_i$$
$$\sigma_i = |\lambda_i|$$
$$\mathbf{v}_i^r = \mathbf{x}_i \cdot sgn(\lambda_i)$$

  - where $\mathbf{x}_i$ is the i-th column of the matrix $\mathbf{X}$, which represents the corresponding eigenvectors of λ_i.

- Based on the problem transformation, we can update the results of GSVD by updating the results of generalized eigenvalue problem.

Dingyuan Zhu, **Peng Cui**, Ziwei Zhang, Jian Pei, Wenwu Zhu. High-order Proximity Preserved Embedding For Dynamic Networks. *IEEE TKDE*, 2018.

Generalized Eigen Perturbation

- We propose generalized eigen perturbation to fulfill the task.
  - The goal of generalized eigen perturbation is to update $X^{(t)}$ to $X^{(t+1)}$

- Specifically, given the change of adjacency matrix △A between two consecutive time steps, the change of Ma and Mb can be represented as:

$$\Delta \mathbf{M}_a = -\beta \Delta \mathbf{A},\ and\ \Delta \mathbf{M}_b = \beta \Delta \mathbf{A}$$

Dingyuan Zhu, **Peng Cui**, Ziwei Zhang, Jian Pei, Wenwu Zhu. High-order Proximity Preserved Embedding For Dynamic Networks. *IEEE TKDE*, 2018.

Complexity Analysis

- Static Model
  - the time complexity of the static model is O(Md^2L), where M is the number of edges in the network and L is the iteration number.

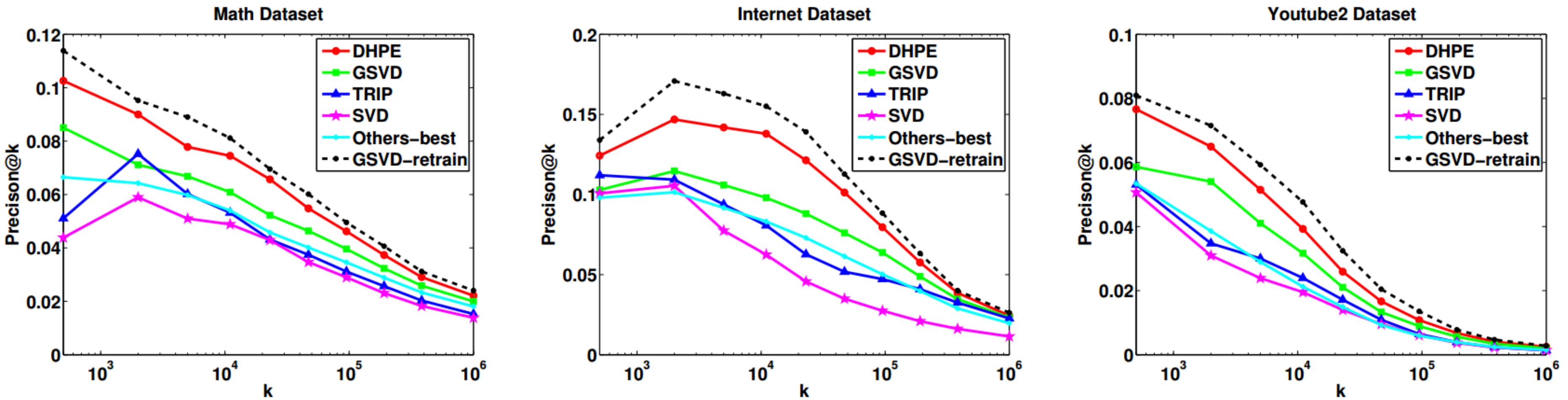
- Dynamic Model
  - Time complexity: $O(T((N+s)\ d\text{^}2+d\text{^}4\ ))$
    - where T is the time slice; N is the number of nodes; s is the number of changes and d is dimensionality of the embedding.

  - Linear with respect to the number of nodes in the network and the number of the newly edges.

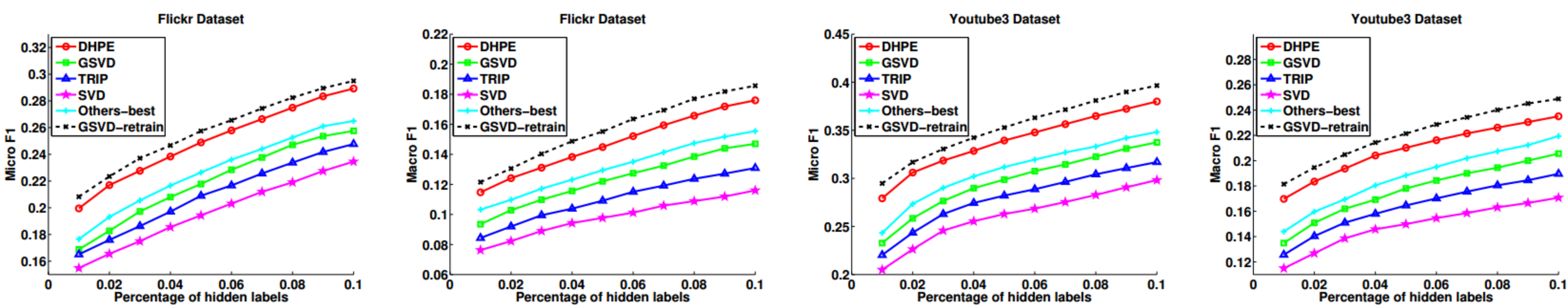
Dingyuan Zhu, **Peng Cui**, Ziwei Zhang, Jian Pei, Wenwu Zhu. High-order Proximity Preserved Embedding For Dynamic Networks. *IEEE TKDE*, 2018.

Experiments

☐ Link prediction



☐ Node Classification



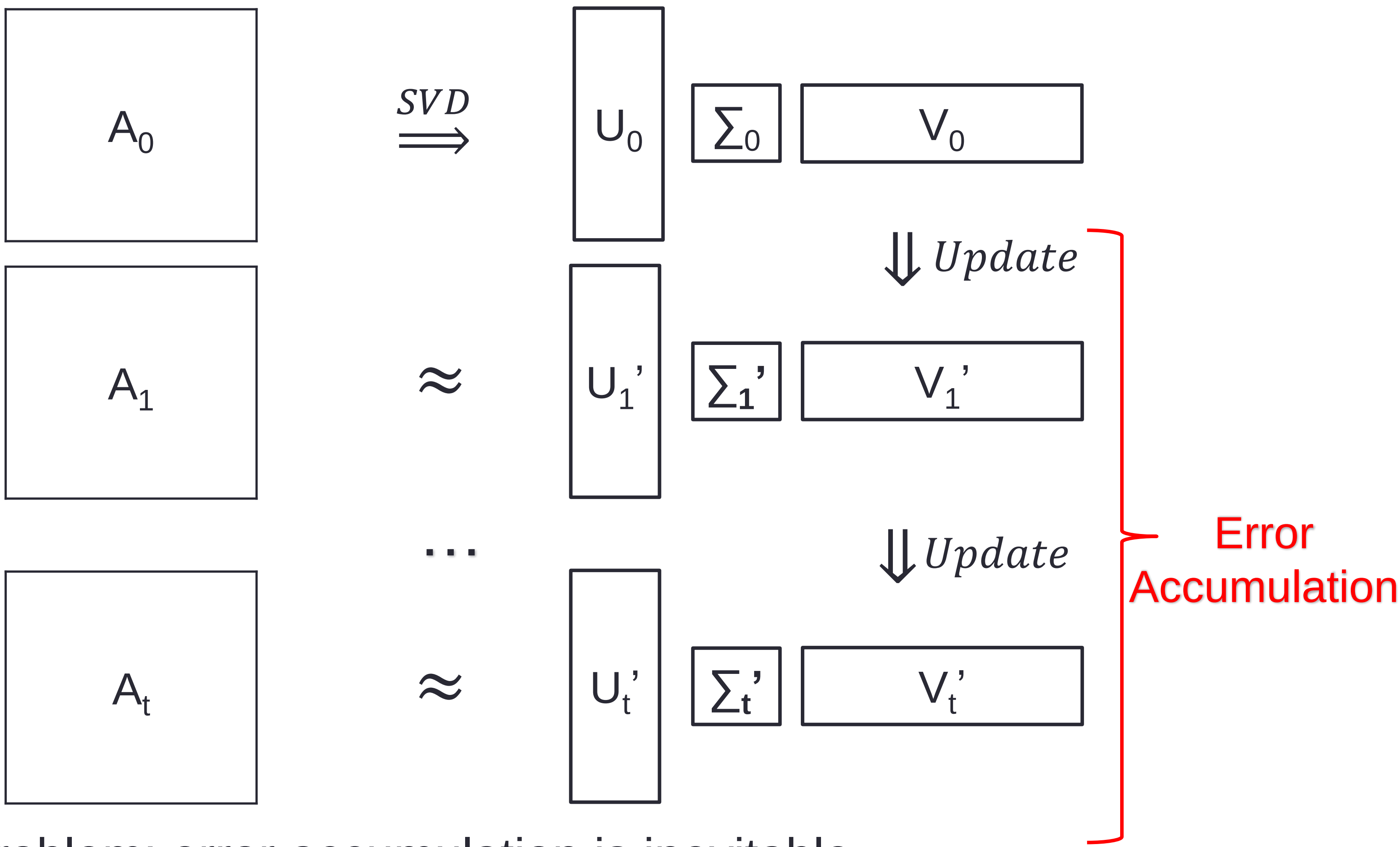
Dingyuan Zhu, **Peng Cui**, Ziwei Zhang, Jian Pei, Wenwu Zhu. High-order Proximity Preserved Embedding For Dynamic Networks. *IEEE TKDE*, 2018.

Key problems in dynamic network embedding

- I : Out-of-sample nodes
- II : Incremental edges
- III: Aggregated error
- IV: Scalable optimization

Problem: Error Accumulation

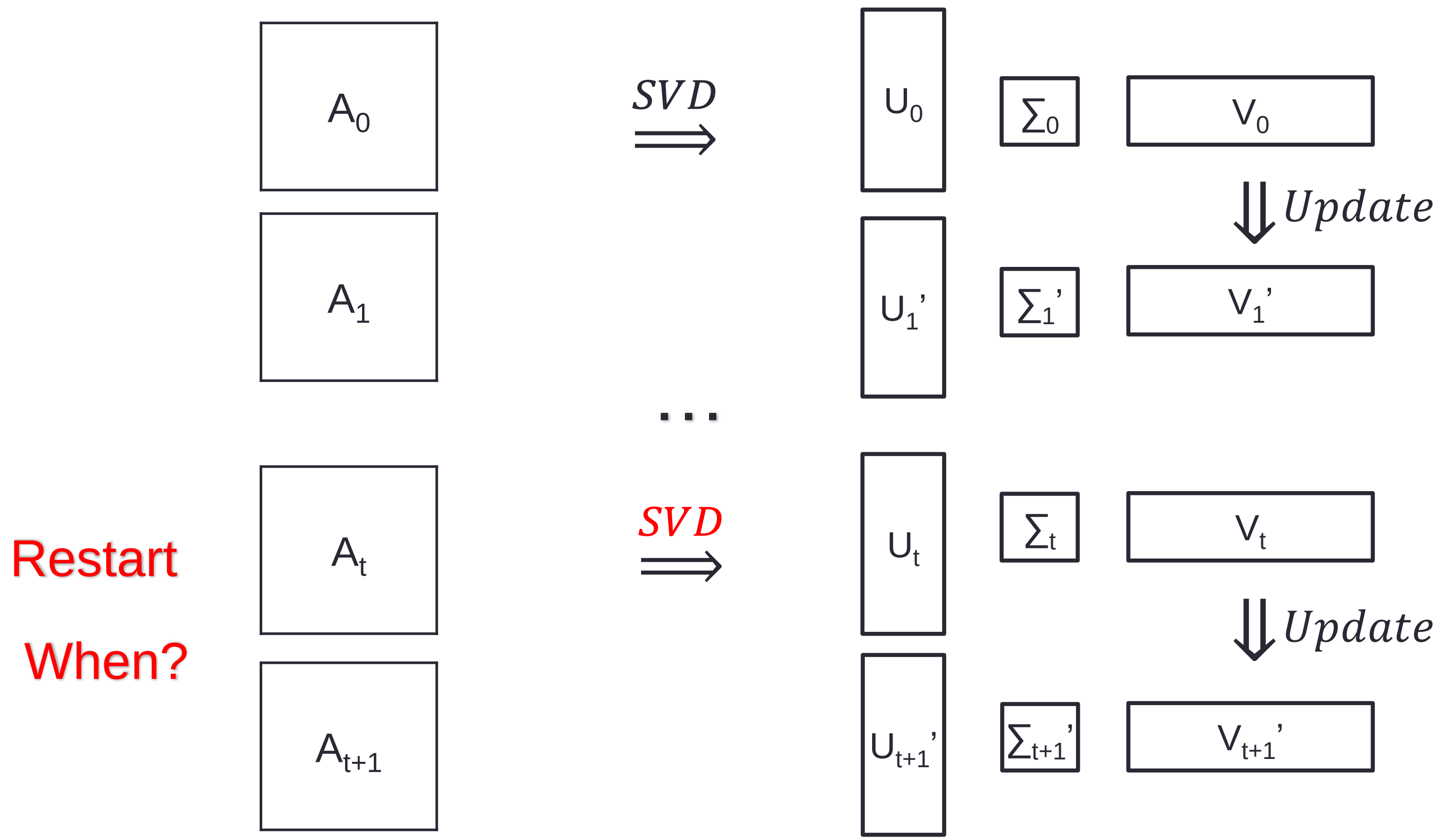
- Eigen perturbation is at the cost of inducing approximation



- Problem: error accumulation is inevitable

Solution: SVD Restarts

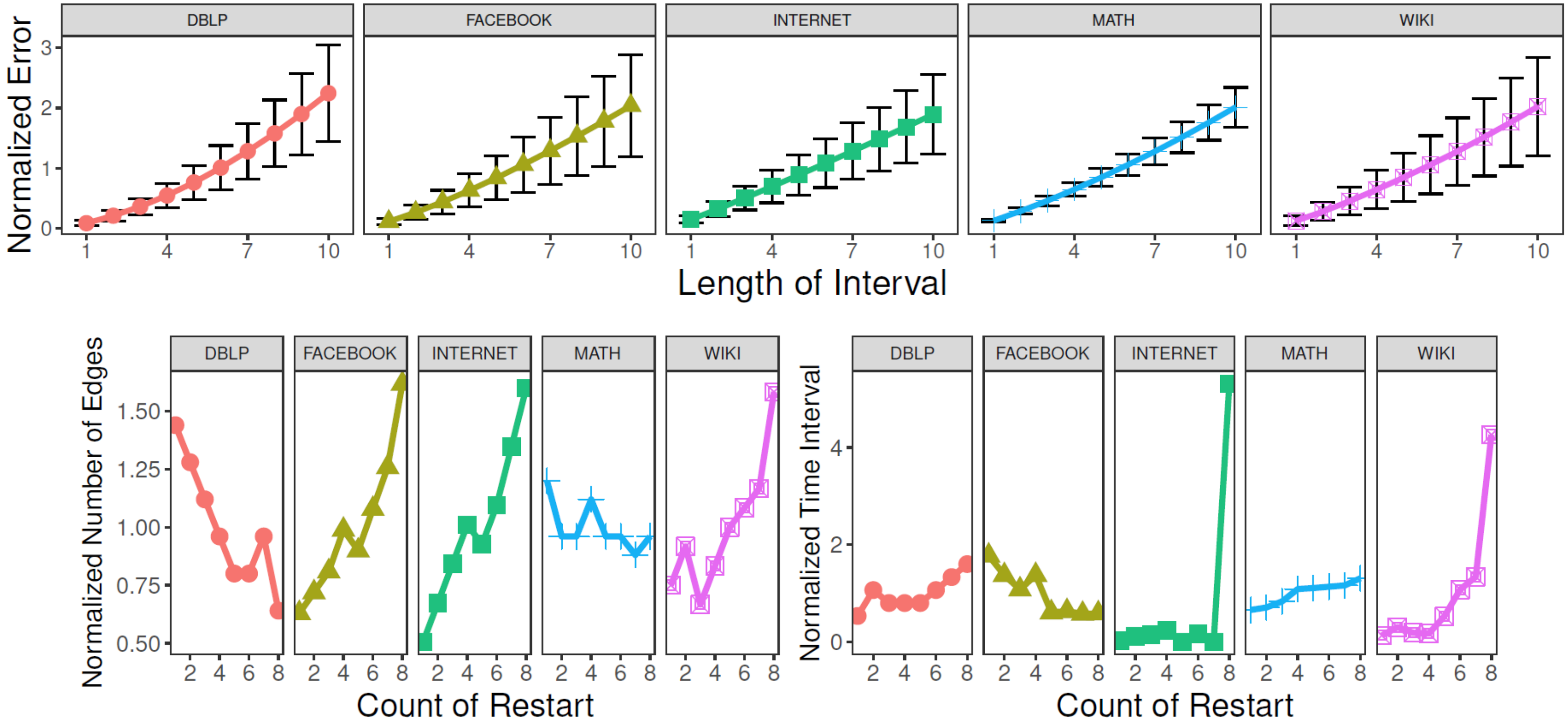
- Solution: restart SVD occasionally



- What are the appropriate time points?
  - Too early restarts: waste of computation resources
  - Too late restarts: serious error accumulation

Naïve Solution

- Naïve solution: fixed time interval or fixed number of changes
- Difficulty: error accumulation is not uniform
  - Validated by preliminary studies



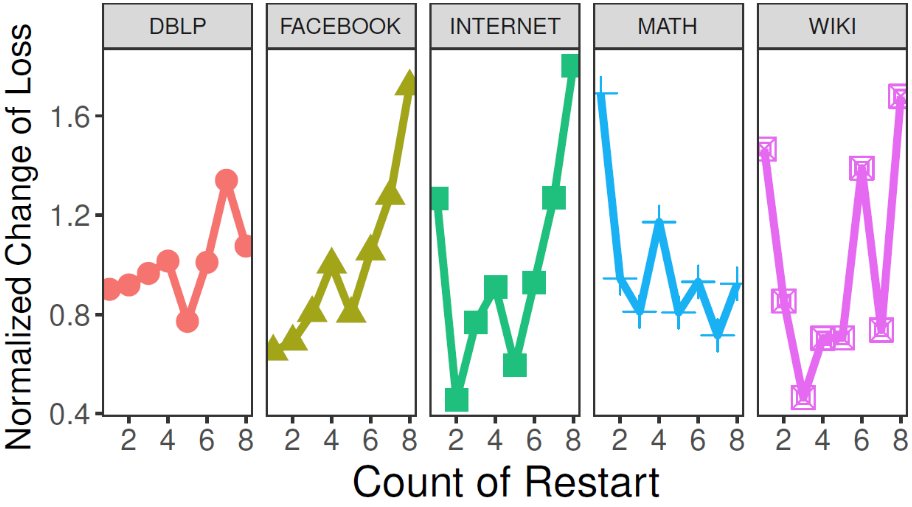
Existing Method

- Existing method: monitor loss (Chen and Candan, KDD 2014)
- Loss in SVD:

$$\mathcal{J} = \|S - U\Sigma V^T\|_F^2$$

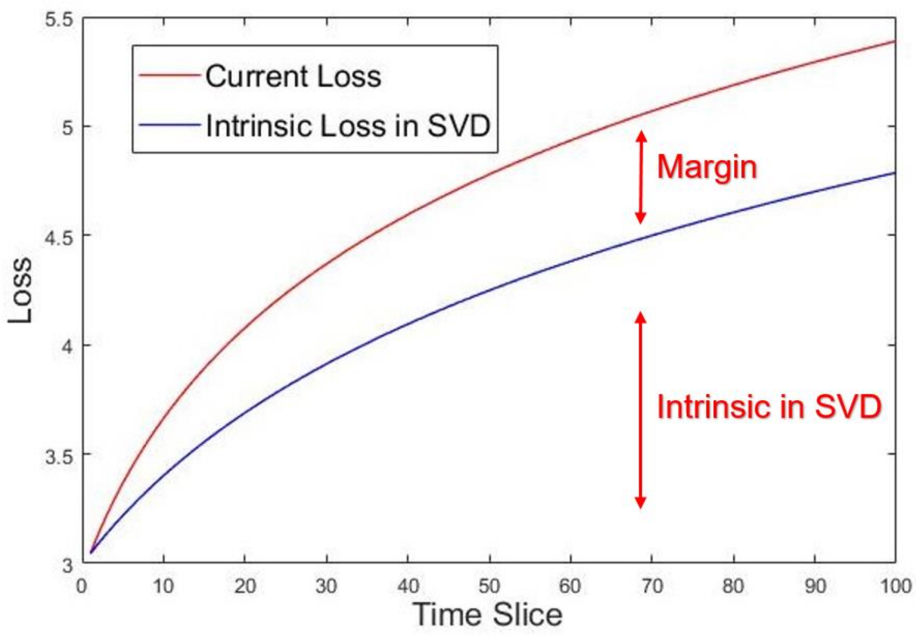
S: target matrix, $[U, \Sigma, V]$: results of SVD

- Problem: loss includes approximation error and intrinsic loss in SVD
- Preliminary study results:



Framework: Monitor Margin

- Observation: the margin between the current loss and intrinsic loss in SVD is the actual accumulated error
  - Current loss: $\mathcal{J} = \|S - U\Sigma V^T\|_F^2$
  - Intrinsic loss: $\mathcal{L}(S,k) = \min_{U^*,\Sigma^*,V^*} \left\|S - U^*\Sigma^* {V^*}^T\right\|_F^2, k: dimensionality$



Ziwei Zhang, **Peng Cui**, Xiao Wang, Jian Pei, Wenwu Zhu. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. *AAAI*, 2018.

Framework: Monitor Margin

- Framework: monitor the maximum margin
- Formulation: constrained optimization

$$\min_{c_1,\ldots,c_T} \sum_{t=1}^{T} c_t$$

$$s.t.\ \ \mathcal{G}\left(\mathbf{S}_0 \ldots \mathbf{S}_T, \left[\mathbf{U}_t, \mathbf{\Sigma}_t, \mathbf{V}_t\right], 1 \leq t \leq T\right) \leq \Theta$$

  - c_t: whether to restart; $\mathcal{G}$: evaluating the margin; Θ: threshold

$$\mathcal{G} = \max_{1 \leq t \leq T} \frac{\mathcal{J}(t) - \mathcal{L}(\mathbf{S}_t, k)}{\mathcal{L}(\mathbf{S}_t, k)}$$

  - Intuition: keep the maximum margin within a threshold while reducing the number of restarts

Ziwei Zhang, **Peng Cui**, Xiao Wang, Jian Pei, Wenwu Zhu. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. *AAAI*, 2018.

A Lower Bound of SVD Intrinsic Loss

- Idea: use matrix perturbation

Theorem 1 (A Lower Bound of SVD Intrinsic Loss)**.** *If* $\mathbf{S}$ *and* $\Delta\mathbf{S}$ *are symmetric matrices, then:*

$$\mathcal{L}(\mathbf{S}+\Delta\mathbf{S},k) \geq \mathcal{L}(\mathbf{S},k) + \Delta tr^2(\mathbf{S}+\Delta\mathbf{S},\mathbf{S}) - \sum_{l=1}^{k}\lambda_l, \quad (9)$$

where $\lambda_1 \geq \lambda_2 ... \geq \lambda_k$ *are the top-*k *eigenvalues of* $\nabla_{S^2} = \mathbf{S}\cdot\Delta\mathbf{S} + \Delta\mathbf{S}\cdot\mathbf{S} + \Delta\mathbf{S}\cdot\Delta\mathbf{S}$, *and*

$$\Delta tr^2(\mathbf{S}+\Delta\mathbf{S},\mathbf{S}) = tr\left((\mathbf{S}+\Delta\mathbf{S})\cdot(\mathbf{S}+\Delta\mathbf{S})\right) - tr(\mathbf{S}\cdot\mathbf{S}).$$

  - Intuition: treat changes as a perturbation to the original network

- Results: need to calculate top-k eigenvalues of $S\cdot\Delta S + \Delta S\cdot S + \Delta S\cdot\Delta S$

Ziwei Zhang, **Peng Cui**, Xiao Wang, Jian Pei, Wenwu Zhu. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. *AAAI*, 2018.

Time Complexity Analysis

Theorem 2. *The time complexity of calculating $B(t)$ in Eqn (13) is $O(M_S + M_L k + N_L k^2)$, where M_S is the number of the non-zero elements in $\Delta\mathbf{S}$, and N_L, M_L are the number of the non-zero rows and elements in ∇_{S^2} respectively.*

- If every node has a equal probability of adding new edges, we have: $M_L \approx 2d_{avg}M_S$, where d_{avg} is the average degree of the network .
- For Barabasi Albert model (Barabási and Albert 1999), a typical example of preferential attachment networks, we have: $M_L \approx \frac{12}{\pi^2}\left[log(d_{max}) + \gamma\right] M_S$, where d_{max} is the the maximum degree of the network and $\gamma \approx 0.58$ is a constant.

- Conclusion: the complexity is only linear to the local dynamic changes

Ziwei Zhang, **Peng Cui**, Xiao Wang, Jian Pei, Wenwu Zhu. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. *AAAI*, 2018.

Experimental Setting

- Baselines:
  - Heu-FL: restart SVD after a fixed number of edges changed
  - Heu-FT: restart SVD after a fixed amount of time passed
  - LWI2 (Chen and Candan, KDD 2014): restart SVD whenever the reconstruction loss exceeds a preset threshold
- Tasks:
  - Approximation error
    - Fixed number of restarts
    - Fixed maximum error
  - Applications
    - Link prediction
    - Eigenvalue tracking

Experimental Results: Approximation Error

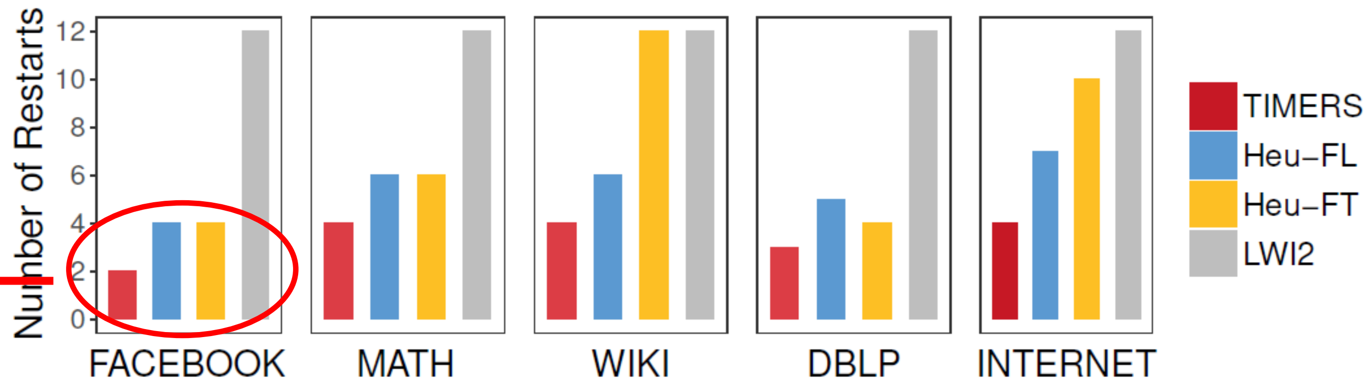
- Fixing number of restarts

Dataset	$avg(r)$				$max(r)$			
	TIMERS	LWI2	Heu-FL	Heu-FT	TIMERS	LWI2	Heu-FL	Heu-FT
FACEBOOK	**0.005**	0.020	0.009	0.011	**0.014**	0.038	0.025	0.023
MATH	**0.037**	0.057	0.044	0.051	**0.085**	0.226	0.117	0.179
WIKI	**0.053**	0.086	0.071	0.281	**0.139**	0.332	0.240	0.825
DBLP	**0.042**	0.110	0.053	0.064	**0.121**	0.386	0.198	0.238
INTERNET	**0.152**	0.218	0.196	0.961	**0.385**	0.806	0.647	1.897

27%~42% Improvement

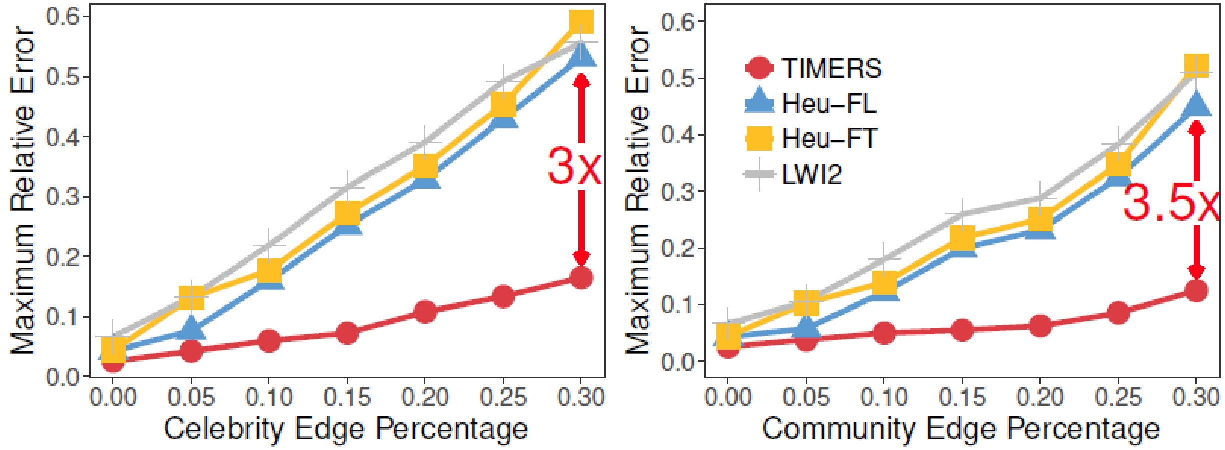
- Fixing maximum error

-50%



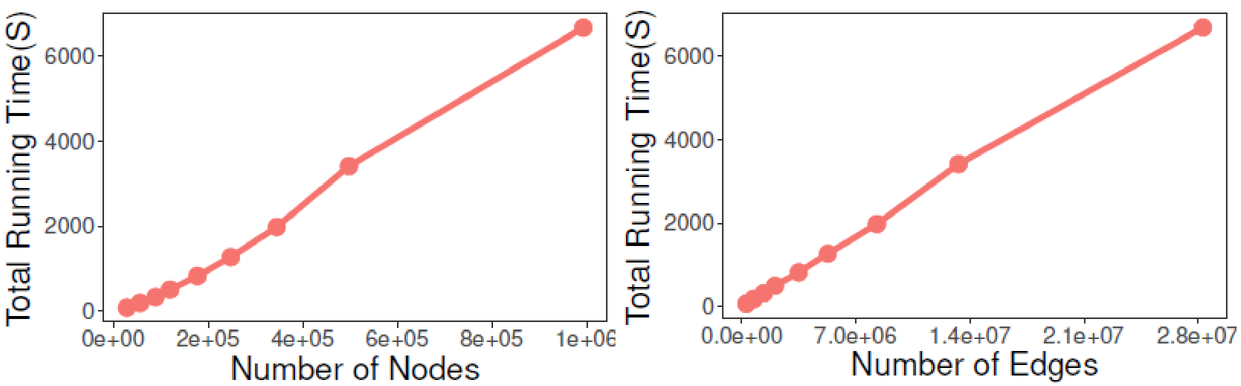
Experimental Results: Analysis

- Syntactic networks: simulate drastic changes in the network structure



  - Robust to sudden changes

- Linear scalability

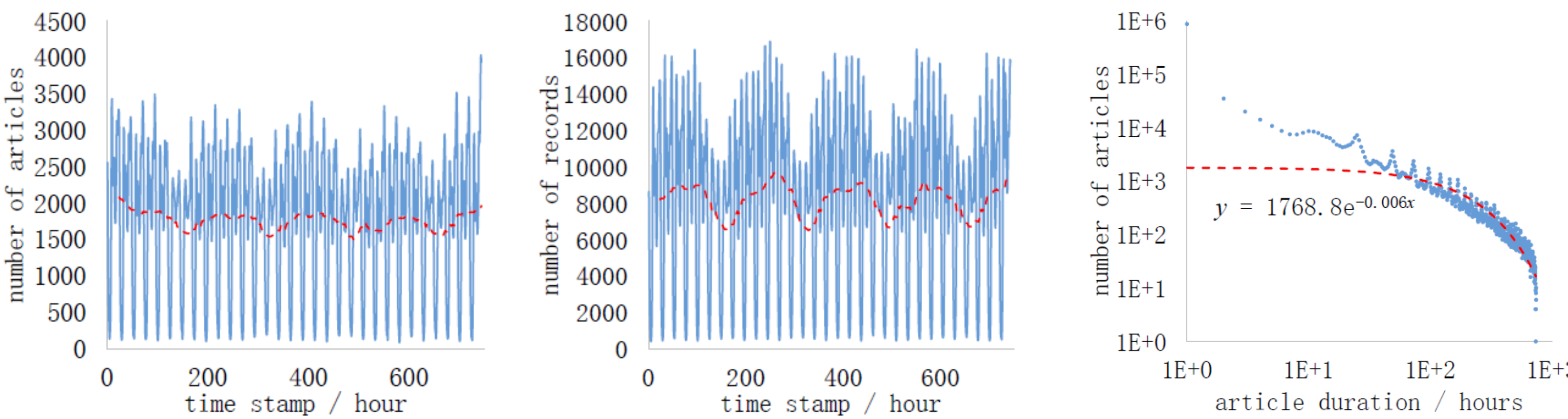


Key problems in dynamic network embedding

- I : Out-of-sample nodes
- II : Incremental edges
- III: Aggregated error
- IV: Scalable optimization

Highly-dynamic & Recency-sensitive Data

- WeChat article reading network is large and highly dynamic
  - 8.1 articles and 1400 reading records per second
- The network is recency-sensitive
  - >73% articles died less than 6 hours while no one read again
  - Obvious exponential decay for article duration length.



Xumin Chen, **Peng Cui**, Lingling Yi, Shiqiang Yang. Scalable Optimization for Embedding Highly-Dynamic and Recency-Sensitive Data. *KDD*, 2018.(Applied Data Science track)

Limited resources

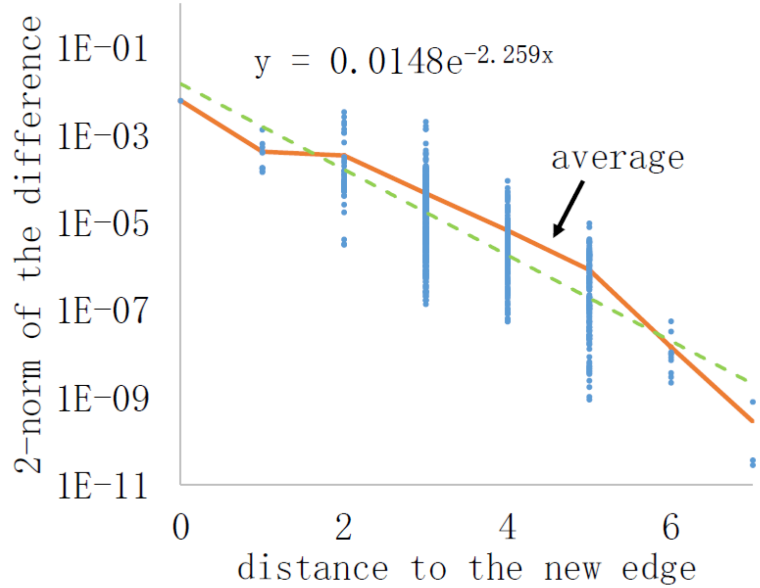
- We cannot guarantee convergence in-between every two timestamps.
- Just do it.

- How to do better?
- Non-uniform resource allocation.
- New edges and nodes worth more resources.

Xumin Chen, **Peng Cui**, Lingling Yi, Shiqiang Yang. Scalable Optimization for Embedding Highly-Dynamic and Recency-Sensitive Data. *KDD*, 2018.(Applied Data Science track)

Diffused SGD: Weight Diffusion Mechanism

- Difference of embedding vector is related to the distance of the changed edge
- Diffuse through **training step**
- For step r, if edge (i, j) is chosen by stochastic method



For edge (i, j), we have

$$p_{i,j}(r) \leftarrow \tau_e\left(i, p_{i,j}(r-1)\right);$$

for $(i, k) \in \mathbb{E} \wedge k \neq j$, we use

$$p_{i,k}(r) \leftarrow p_{i,k}(r-1) + \tau_n\left(i, p_{i,j}(r-1)\right);$$

and for other edges $(l, k) \in \mathbb{E} \wedge l \neq i$,

$$p_{l,k}(r) \leftarrow p_{l,k}(r-1);$$

Boundary and Convergence

- Convergence
  - The sub-function of loss function is convex and smooth if we fix $\mathbf{V}(r)$ and t, and almost always strongly convex

$$J_{i,j}^{(t)}\left(\mathbf{U},\mathbf{V}(r)\right) = w_{i,j}^{(t)}\left(\mathbf{u}_i^{\mathrm{T}}\cdot\mathbf{v}_j - a_{i,j}^{(t)}\right)^2$$

  - If we sample sub-functions with probability proportional to a weight function, the expecting step size r is bounded by optimal error, initial error and the Lipschitz constant of the strongly convex function
- Boundary
  - When edge (i,j) added, difference of optimal function is bounded

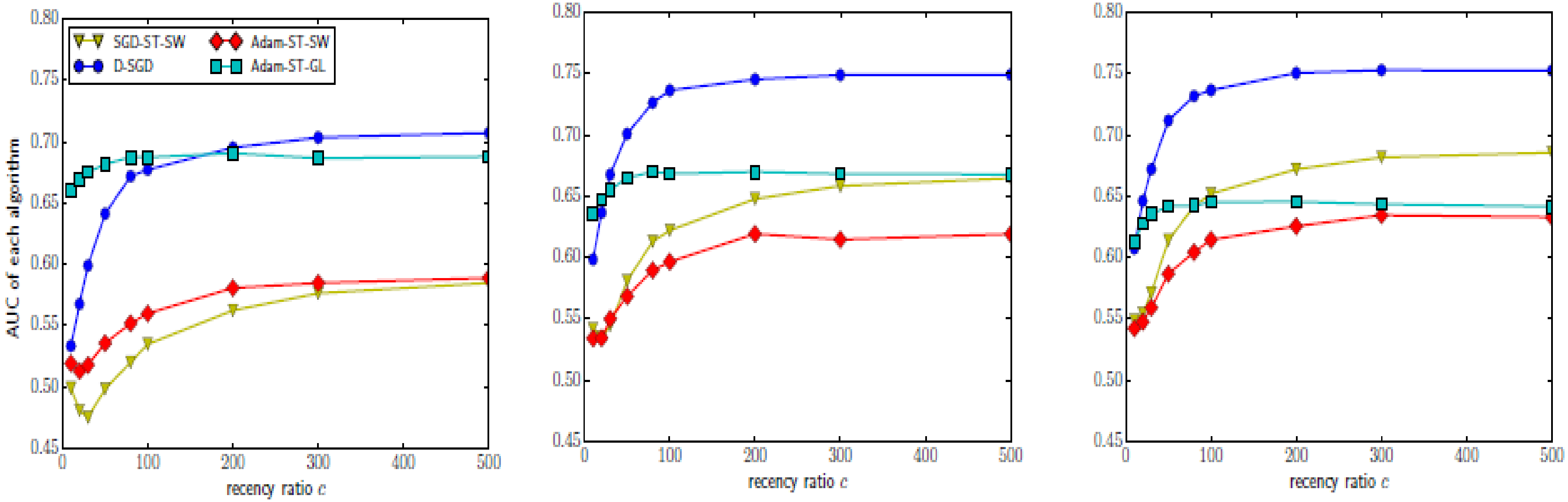
$$\frac{\mu}{2}\sum_{k\neq i}\left\|\mathbf{u}_{k*}^{(t+1)} - \mathbf{u}_{k*}^{(t)}\right\|_2^2 + \left\|\sqrt{\frac{\mu}{2}}\left(\mathbf{u}_{i*}^{(t+1)} - \mathbf{u}_{i*}^{(t)}\right) + \sqrt{\frac{2}{\mu}}w_{i,j}^{(t+1)^2}a_{i,j}^{(t)}\mathbf{v}_j\right\|_2^2$$
$$\leq \left(\left|w_{i,j}^{(t+1)^2} - w_{i,j}^{(t)^2}\right| + \frac{2}{\mu}w_{i,j}^{(t+1)^4}a_{i,j}^{(t)^2}\right)\mathbf{v}_j^{\mathrm{T}}\mathbf{v}_j$$

  - Similar boundary if $w_{i,j}^{(t)}(r)$ decreased to $w_{i,j}^{(t)}(r+1)$
- As the difference is bounded, the number optimal step is also bounded with the conclusion of convergence

Xumin Chen, **Peng Cui**, Lingling Yi, Shiqiang Yang. Scalable Optimization for Embedding Highly-Dynamic and Recency-Sensitive Data. *KDD*, 2018.(Applied Data Science track)

Experiment: AUC

- $\max r$: max iteration steps of each time stamp t
- c: the less the c is, the more recency-sensitive the dataset is



(a) AUC when max r = 3

(b) AUC when max r = 6

(c) AUC when max r = 9

Experiment: Running Time

- For each $G^{(t)}$, count the running time for every method to a similar AUC(0.73 in experiment)

timestamp t	D-SGD	D-SGD-WT	SGD-SW	Adam-SW	Adam-GL
1×10^0	0.0234	0.134	0.231	1.84	1.58
1×10^3	0.0307	0.147	0.238	1.79	1.54
3×10^3	0.0295	0.169	0.261	1.82	1.60
1×10^4	0.0347	0.255	0.279	1.88	1.82
3×10^4	0.0336	0.469	0.423	1.98	2.44
1×10^5	0.0441	1.20	0.388	2.15	4.63
3×10^5	0.0568	3.61	0.498	2.39	10.9
1×10^6	0.0739	15.1	0.668	2.77	32.6
3×10^6	0.0664	45.9	0.684	2.87	96.2

Summary

- **I : Out-of-sample nodes**
  - **DepthLGP = Non-parametric GP + DNN**
- **II : Incrementally updating**
  - **DHPE: Generalized Eigen Perturbation**
- **III: Aggregated error**
  - **TIMERS: A theoretically guaranteed SVD restart strategy**
- **IV: Scalable optimization**
  - **D-SGD: A iteration-wise weighted SGD for highly dynamic data**

References

- Peng Cui, Xiao Wang, Jian Pei, Wenwu Zhu. A Survey on Network Embedding. ***IEEE TKDE***, 2018.
- Dingyuan Zhu, Peng Cui, Daixin Wang and Wenwu Zhu. Deep Variational Network Embedding in Wasserstein Space. ***KDD***, 2018.
- Ziwei Zhang, Peng Cui, Xiao Wang, Jian Pei, Xuanrong Yao and Wenwu Zhu. Arbitrary-Order Proximity Preserved Network Embedding. ***KDD***, 2018.
- Ke Tu, Peng Cui, Xiao Wang, Philip S. Yu and Wenwu Zhu. Deep Recursive Network Embedding with Regular Equivalence. ***KDD***, 2018.
- Jianxin Ma, Peng Cui, Xiao Wang and Wenwu Zhu. Hierarchical Taxonomy Aware Network Embedding. ***KDD***, 2018.
- Xumin Chen, Peng Cui, Lingling Yi, Shiqiang Yang. Scalable Optimization for Embedding Highly-Dynamic and Recency-Sensitive Data. ***KDD***, 2018.(Applied Data Science track)
- Dingyuan Zhu, Peng Cui, Ziwei Zhang, Jian Pei, Wenwu Zhu. High-order Proximity Preserved Embedding For Dynamic Networks. ***IEEE TKDE***, 2018.

References

- Ziwei Zhang, Peng Cui, Xiao Wang, Jian Pei, Wenwu Zhu. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. ***AAAI***, 2018.
- Ke Tu, Peng Cui, Xiao Wang, Fei Wang, Wenwu Zhu. Structural Deep Embedding for Hyper-Networks. ***AAAI***, 2018.
- Jianxin Ma, Peng Cui, Wenwu Zhu. DepthLGP: Learning Embeddings of Out-of-Sample Nodes in Dynamic Networks. ***AAAI***, 2018.
- Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, Shiqiang Yang. Community Preserving Network Embedding. ***AAAI***, 2017.
- Daixin Wang, Peng Cui, Wenwu Zhu. Structural Deep Network Embedding. ***KDD***, 2016. *(Full Paper, Oral)*
- Mingdong Ou, Peng Cui, Jian Pei, Wenwu Zhu. Asymmetric Transitivity Preserving Graph Embedding. ***KDD***, 2016. *(Full Paper, Oral)*
- Mingdong Ou, Peng Cui, Fei Wang, Jun Wang, Wenwu Zhu. Non-transitive Hashing with Latent Similarity Components. ***KDD***, 2015.(Full Paper, Oral)
- Mingdong Ou, Peng Cui, Fei Wang, Jun Wang, Wenwu Zhu, Shiqiang Yang. Comparing Apples to Oranges: A Scalable Solution with Heterogeneous Hashing. ***KDD***, 2013.*(Full Paper, Oral)*

Thanks!

Email: cuip@tsinghua.edu.cn

NRL page: http://nrl.thumedialab.com/

My homepage: http://pengcui.thumedialab.com