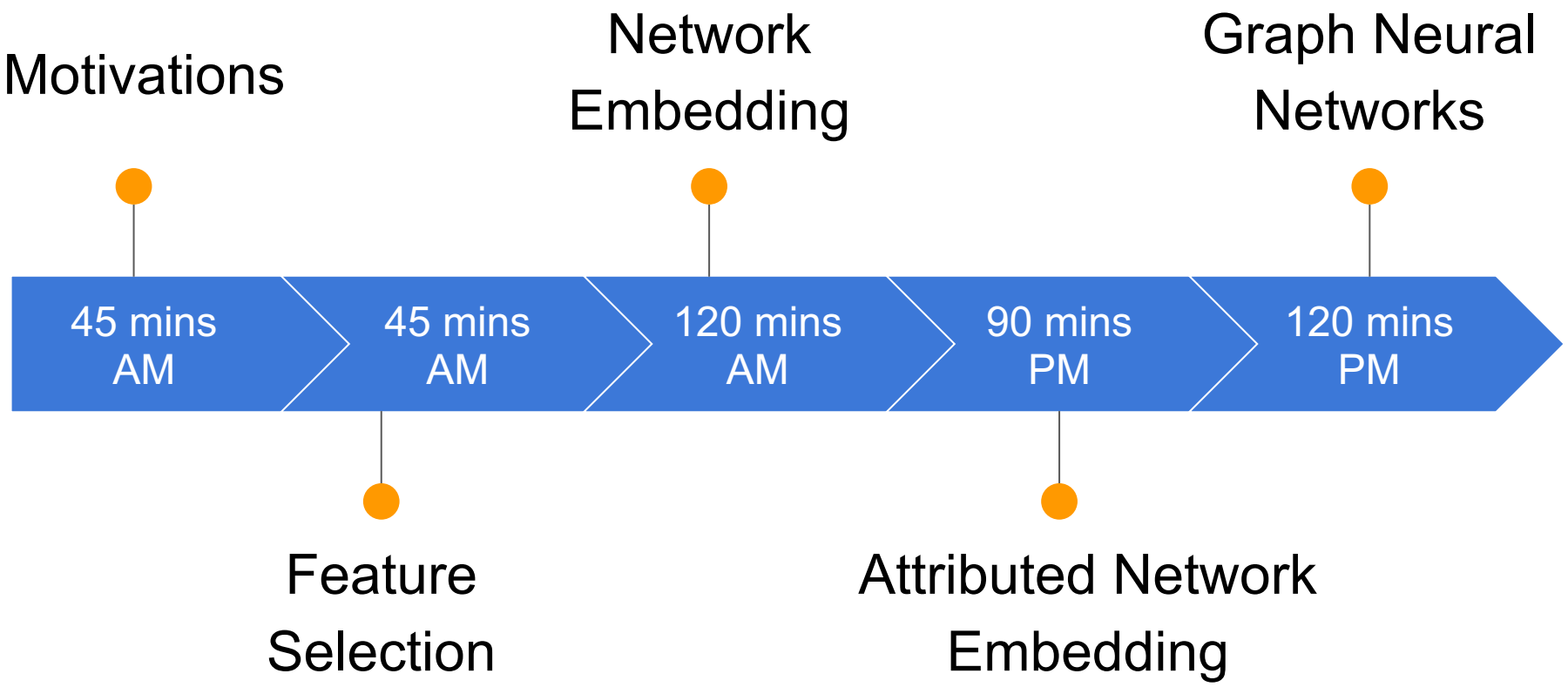
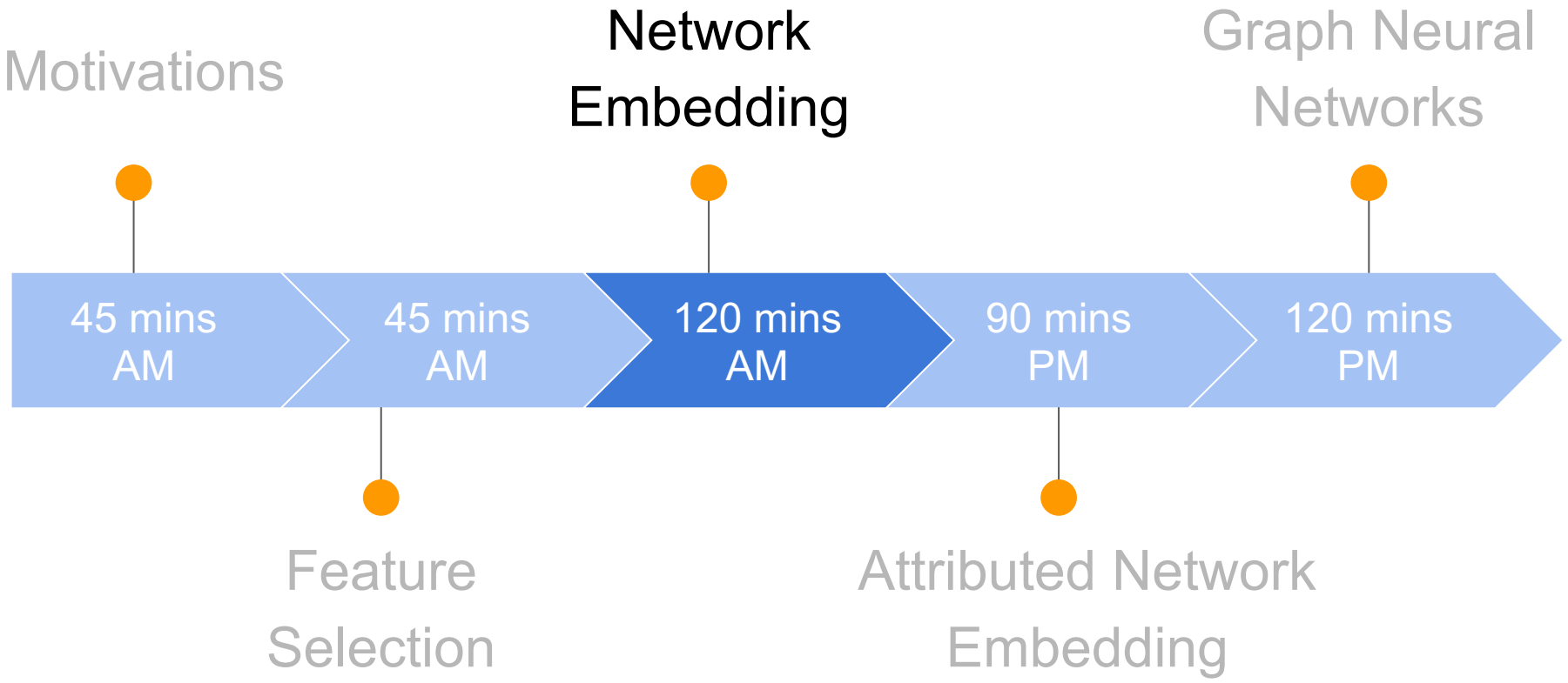
Learning From Networks

——*Algorithms, Theory, & Applications*

Xiao Huang, Peng Cui, Yuxiao Dong, Jundong Li, Huan Liu, Jian Pei, Le Song,
Jie Tang, Fei Wang, Hongxia Yang, Wenwu Zhu

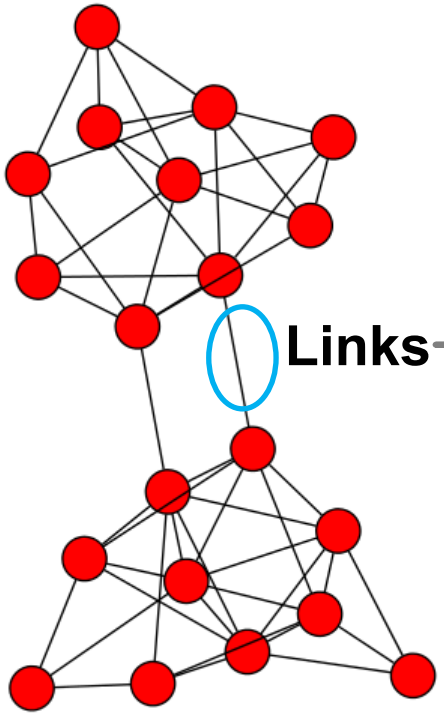
xhuang@tamu.edu; cuip@tsinghua.edu.cn; yuxdong@microsoft.com; jundongl@asu.edu;
huan.liu@asu.edu; jpei@cs.sfu.ca; le.song@antfin.com; jietang@tsinghua.edu.cn;
few2001@med.cornell.edu; yang.yhx@alibaba-inc.com; wwzhu@tsinghua.edu.cn;


Motivations
Network Embedding
Graph Neural Networks
45 mins AM
45 mins AM
120 mins AM
90 mins PM
120 mins PM
Feature Selection
Attributed Network Embedding


Motivations
Network Embedding
Graph Neural Networks
45 mins AM
45 mins AM
120 mins AM
90 mins PM
120 mins PM
Feature Selection
Attributed Network Embedding

The problem of network representation

G = (V, E)



Links

Iterative &
Combinatorial

Coupling

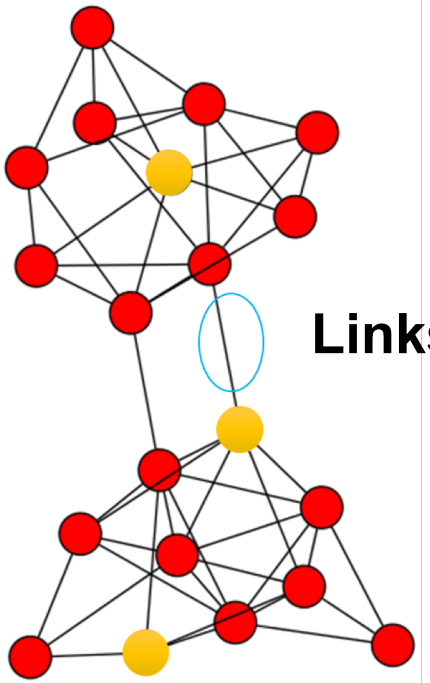
Complexity

Computability

Parallelizability

The problem of network representation

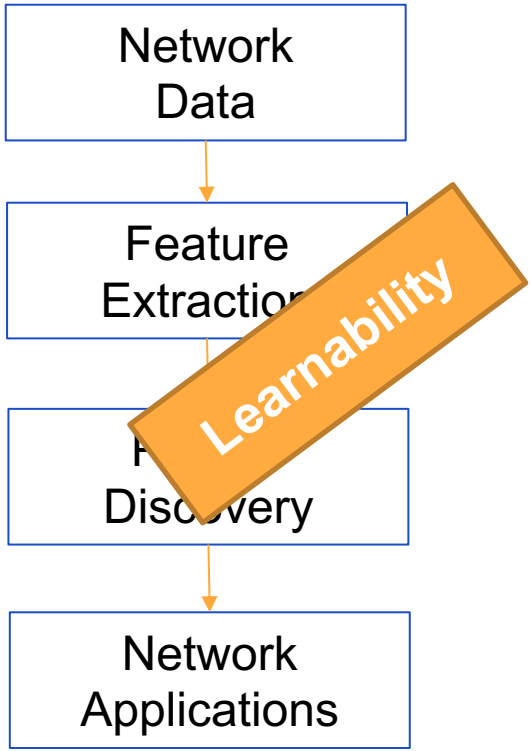
G = (V, E)



Links ⇨ Topology

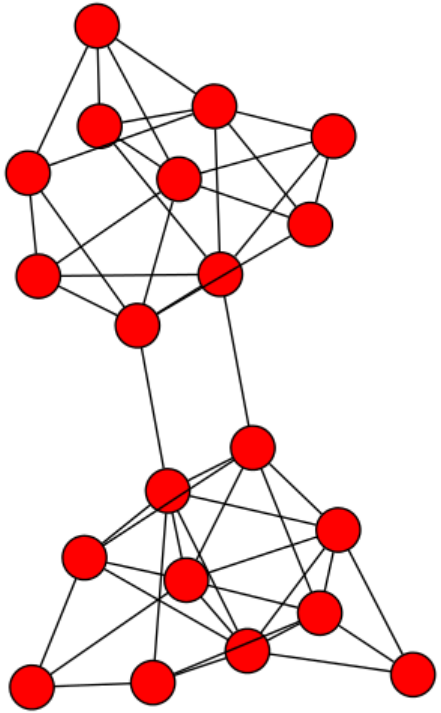
Inapplicability of ML methods

Pipeline for network analysis



Revisit network representation

G = (V, E)

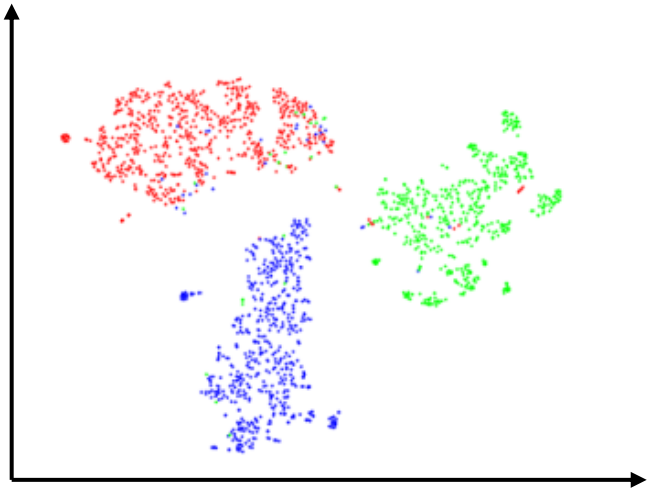


generate

embed

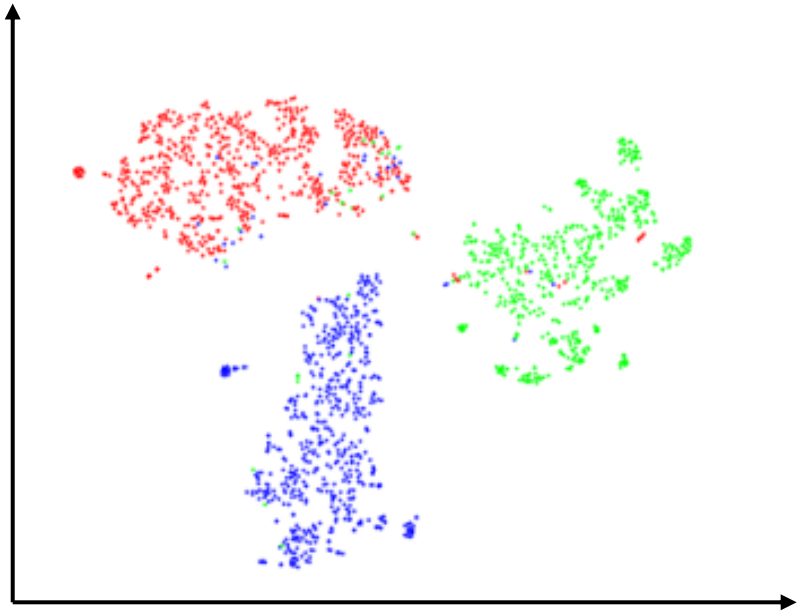
G = (V)

Vector Space



- Easy to parallel
- Can apply classical ML methods

The ultimate goal

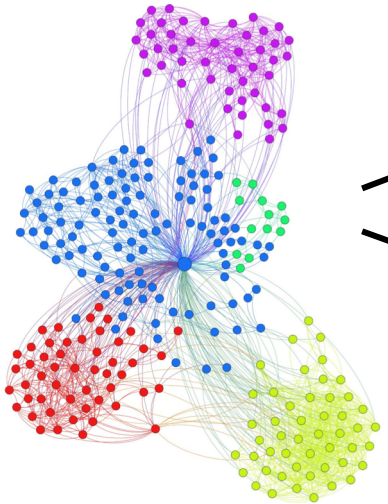


Network Inference

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

in *Vector Space*

The information encoded in networks



Topological Information

Semantic Information

Nodes Attributes

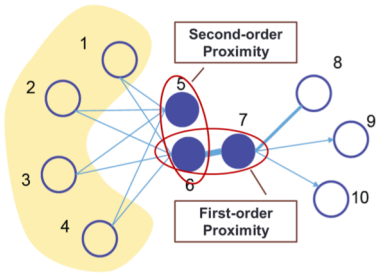
Topology Vectorization is the key problem.

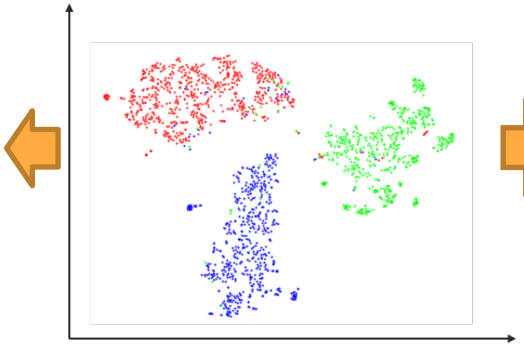
Network Embedding

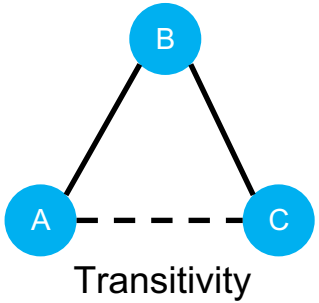
Goal Support network inference in vector space

Reflect network structure

Maintain network properties







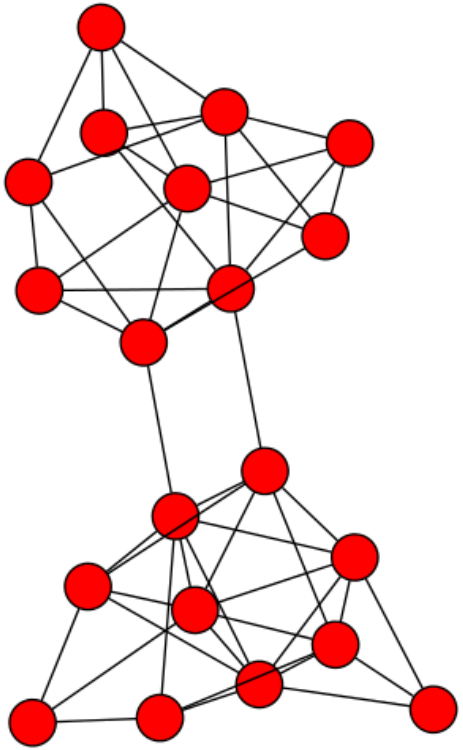
Outline

- Structure-preserved network embedding
- Property-preserved network embedding
- Dynamic network embedding
- Robustness, Explainability and Applicability
- Network embedding for biomedical applications

Outline

- **Structure-preserved network embedding**
- **Property-preserved network embedding**
- **Dynamic network embedding**
- **Robustness, Explainability and Applicability**
- **Network embedding for biomedical applications**

Network Structures



Nodes & Links

Pair-wise Proximity

Community Structures

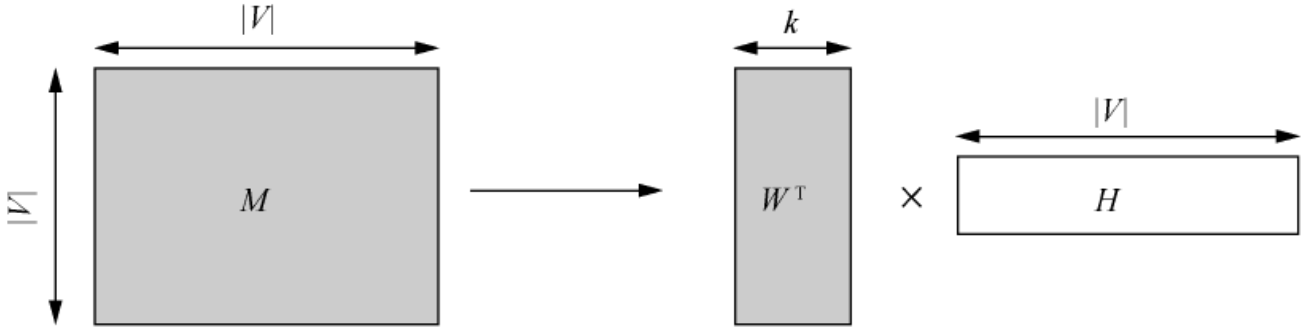
Hyper Edges

Global Structure

Nodes & Links

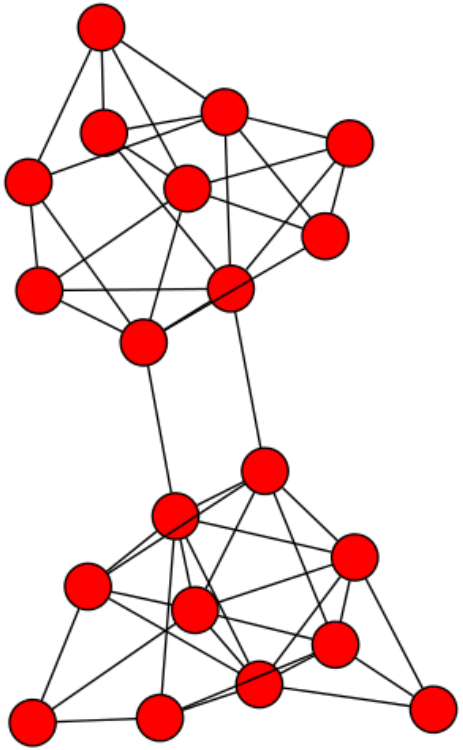
Reconstruct the original network

Matrix Factorization



Reconstruct all the links? May cause overfitting.
The network inference ability is seriously limited.

Network Structures



Nodes & Links

Pair-wise Proximity

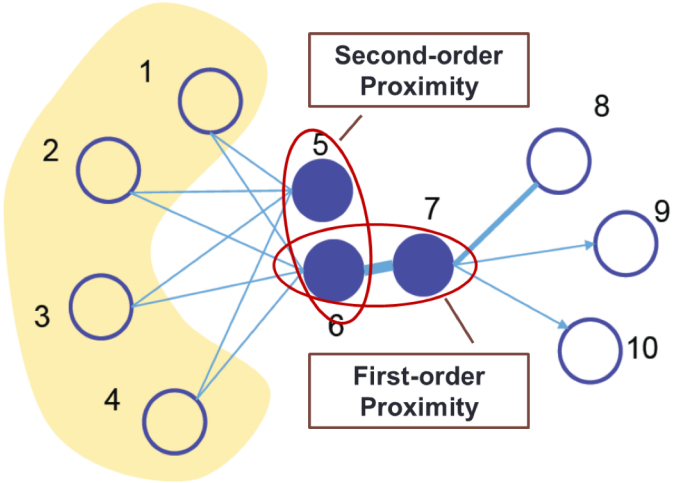
Community Structures

Hyper Edges

Global Structure

High-Order Proximity

- Capturing the underlying structure of networks



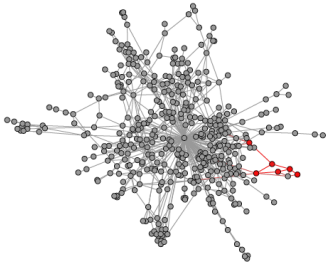
- Advantages:
  - Solve the sparsity problem of network connections
  - Measure indirect relationship between nodes

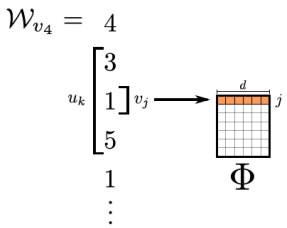
Deepwalk

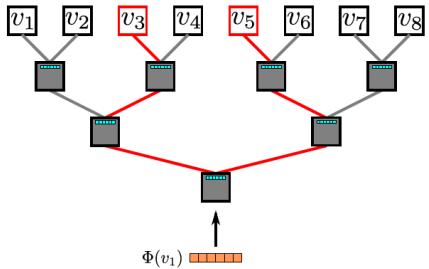
- Exploit truncated random walk to define neighborhood of a node.

Random Walks on Graph

- $V_{26} - V_{25} - V_{32} - V_3 - V_{10} \cdots$
- $V_5 - V_7 - V_{17} - V_6 - V_{11} \ldots$
- $V_{31} - V_{33} - V_{21} - V_{33} - V_{15}$



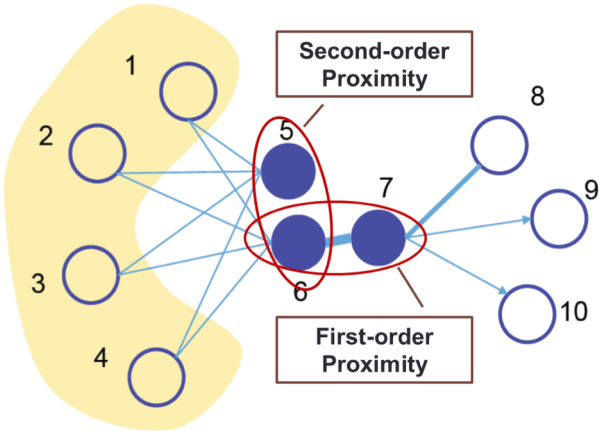




(a) Random walk generation. (b) Representation mapping. (c) Hierarchical Softmax.

B. Perozzi et al. Deepwalk: Online learning of social representations. *KDD 2014.*

LINE



LINE with First-order Proximity:
local pairwise

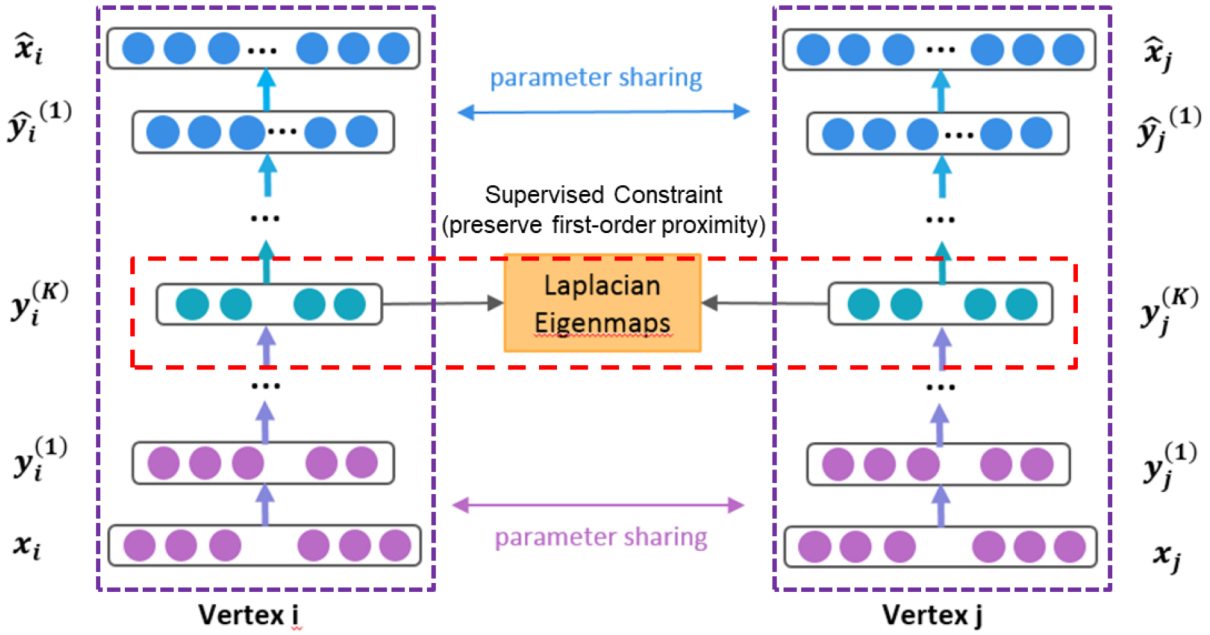
$$O_1 = -\sum_{(i,j)\in E} w_{ij} \log p_1(v_i, v_j)$$

LINE with Second-order Proximity:
neighborhood structures

$$O_2 = \sum_{i\in V} \lambda_i d(\hat{p}_2(\cdot|v_i), p_2(\cdot|v_i))$$

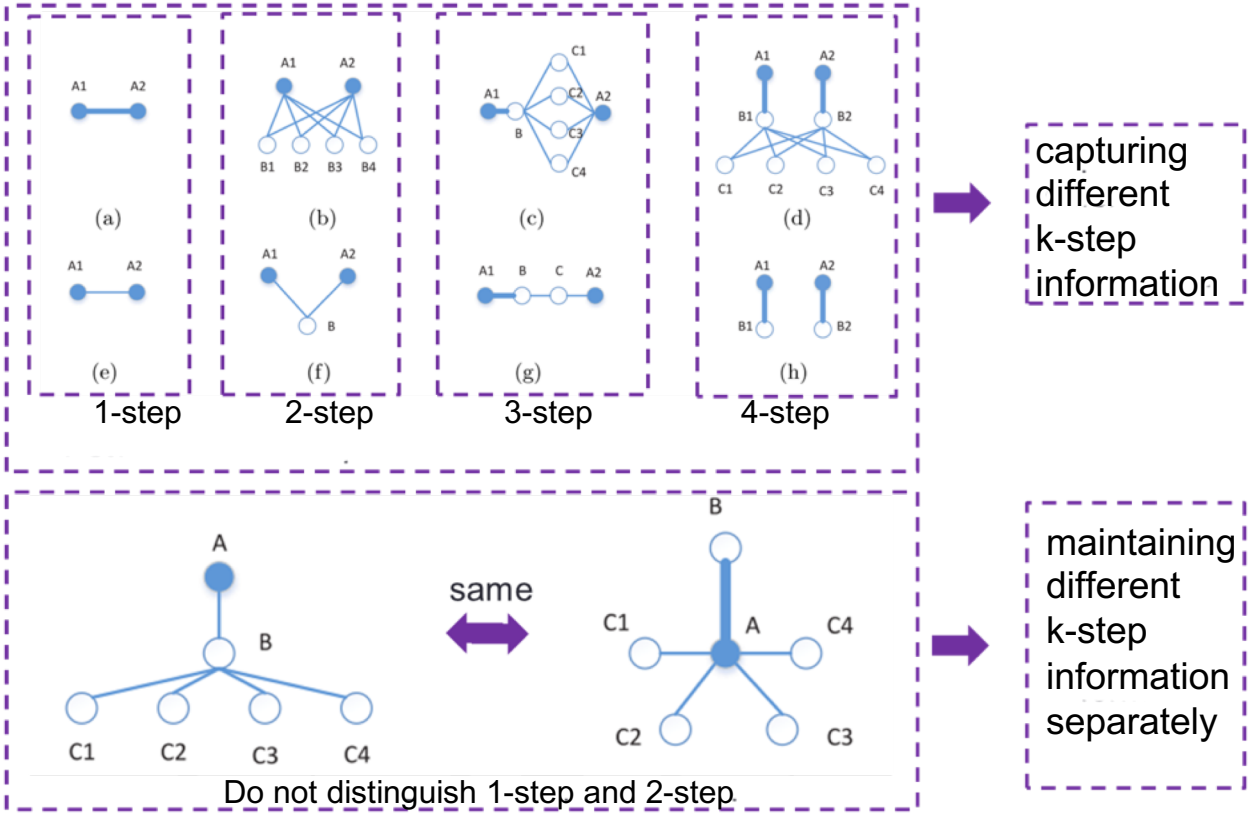
Jian Tang et al. LINE: Large-scale Information Network Embedding. *WWW 2015*.

SDNE – Structural Deep Network Embedding



Daixin Wang et al. Structural Deep Network Embedding. *KDD*, 2016.

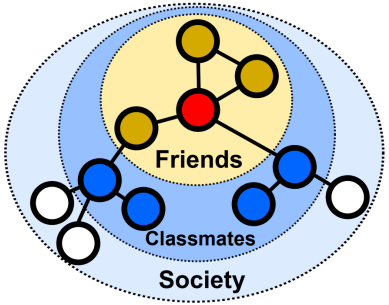
GraRep



Shaosheng Cao et al. GraRep: Learning Graph Representations with Global Structural Information. *CIKM 2015*.

What is the *right* order?

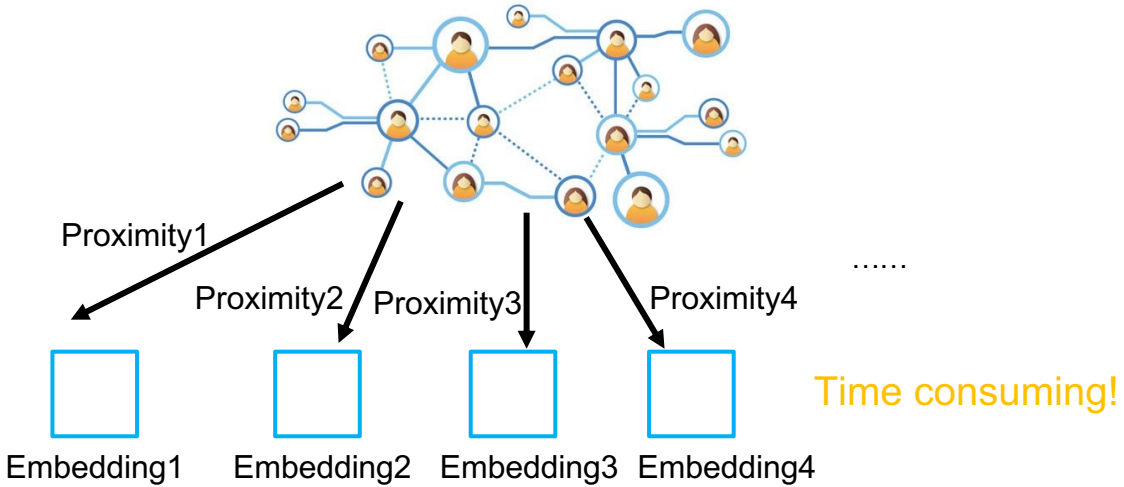
- Different networks/tasks require different high-order proximities
  - E.g., multi-scale classification (Bryan Perozzi, et al, 2017)



  - 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)

What is the *right* order?

- Existing methods can only preserve one fixed high-order proximity
  - Different high-order proximities are calculated separately



- -> How to preserve arbitrary-order proximity while guaranteeing accuracy and efficiency?

Problem Formulation

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

$$S = f(A) = w_1 A^1 + w_2 A^2 + \cdots + w_q A^q$$

  - 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

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

Z. Zhang, et al. Arbitrary-Order Proximity Preserved Network Embedding. *KDD, 2018.*

Problem Transformation

- The equivalence between top-d SVD and eigen-decomposition
  - $[U, \Sigma, V]$: top-d SVD . $[\Lambda, \mathrm{X}]$: top-d eigen-decomposition

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

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

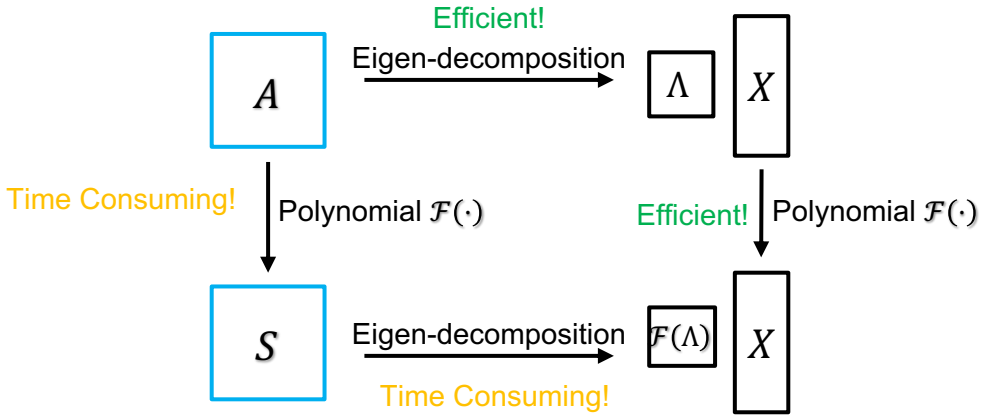
  - How to solve $[\Lambda, \mathrm{X}]$ for $S = f(A) = w_1 A^1 + w_2 A^2 + \cdots + w_q A^q$

Z. Zhang, et al. Arbitrary-Order Proximity Preserved Network Embedding. *KDD, 2018.*

Eigen-decomposition Reweighting

- Eigen-decomposition reweighting

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



- **Insights**: high-order proximity is simply re-weighting dimensions!

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

Z. Zhang, et al. Arbitrary-Order Proximity Preserved Network Embedding. *KDD, 2018.*

Eigen-decomposition Reweighting

- Re-ordering of dimensions
  - As Λ is not necessarily positive, the top-d $\mathcal{F}(\Lambda)$ may not correspond to the top-d Λ
  - How many eigen pairs of A do we need for top-d eigen pairs of $S = \mathcal{F}(A)$?

THEOREM 4.3. *l satisfies that the top l eigenvalues of* **A** *have d positive, i.e.*

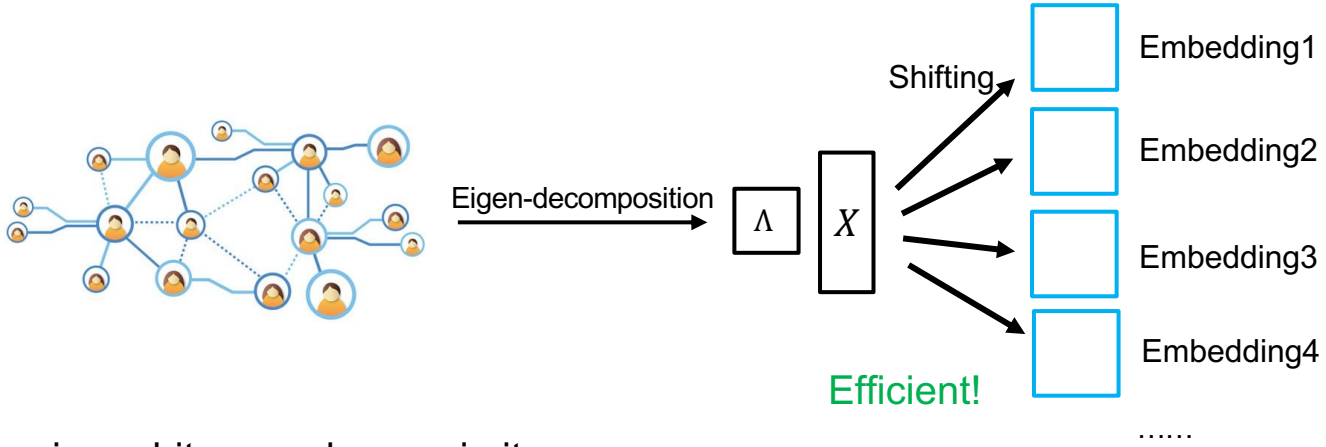
$$l = \mathcal{L}(\mathbf{A}, d) = \min\ l' \quad s.t.\ \sum_{j=1}^{l'} \mathbb{I}\left(\lambda_j > 0\right) = d,$$

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

Z. Zhang, et al. Arbitrary-Order Proximity Preserved Network Embedding. *KDD, 2018.*

Preserving Arbitrary-Order Proximity

- Shifting across different orders/weights:

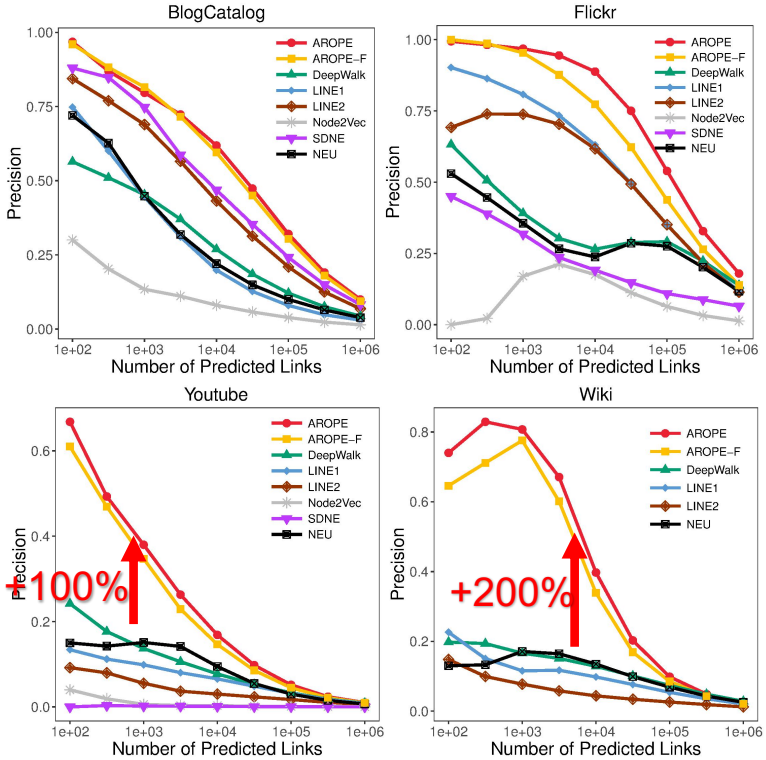


- Preserving arbitrary-order proximity
- Low marginal cost
- Accurate and efficient

Z. Zhang, et al. Arbitrary-Order Proximity Preserved Network Embedding. *KDD, 2018.*

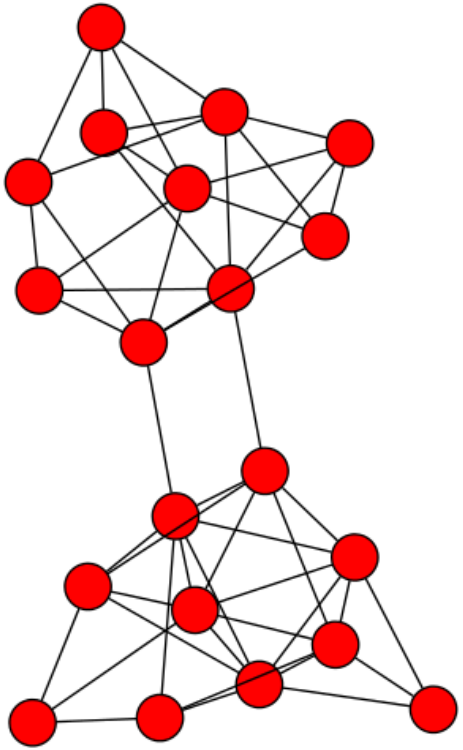
Experimental Results

- Link Prediction



Z. Zhang, et al. Arbitrary-Order Proximity Preserved Network Embedding. *KDD, 2018.*

Network Structures



Nodes & Links

Pair-wise Proximity

Community Structures

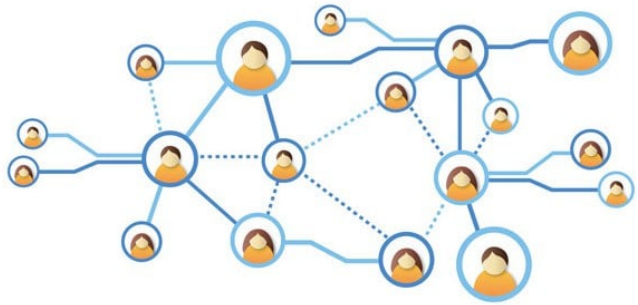
Hyper Edges

Global Structure

Motivation

- Vertexes in different parts of the network may have similar roles(global position)

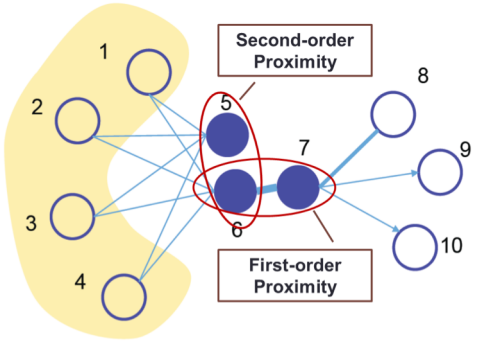
- Example :
  - Managers in the social network of a company
  - Outliers in a network in the task of anomaly detection

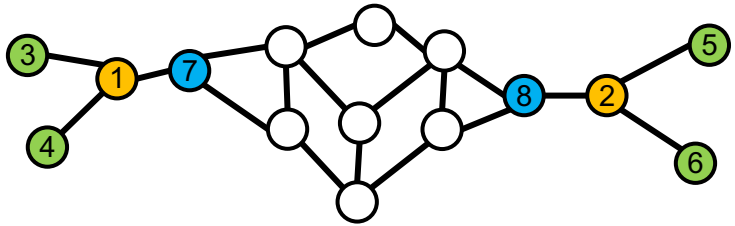


Social network with different position

How to reflect the role or importance of a vertex in embedding space?

Existing embedding methods

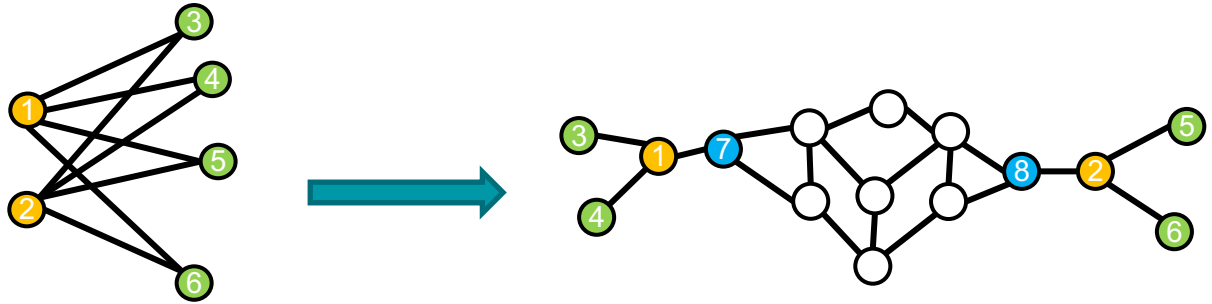




- They can only preserve local proximity(Structural equivalence), can not reflect the global position
  - Embeddings of node 5,6 in left network will be similar but embeddings of node 1, 2 in right network will not be similar.

Regular Equivalence

Two nodes are regularly equivalent if their network neighbors are themselves similar (i.e. regularly equivalent).



- Structural equivalence s
  - $N(u) = N(v)$
  - Direct way
  - Common neighbors

- Regular equivalence r
  - $\{r(i)|i \in N(u)\} = \{r(j)|j \in N(v)\}$
  - Recursive way
  - Similar global position

Regular equivalence is largely ignored in network embedding

Naïve Solutions

- Basis: two regularly equivalent nodes should have similar embeddings

1. Explicitly calculate the regular equivalence of all vertex pairs
   - infeasible for large-scale networks due to the high complexity of calculating regular equivalence
2. Replace regular equivalence into simpler graph theoretic metrics
   - centrality measures
   - one centrality can only capture a specific aspect of network role
   - some centrality measures also bear high computational complexity

Ke Tu, et al. Deep Recursive Network Embedding with Regular Equivalence. ***KDD***, 2018.

Deep Recursive Network Embedding

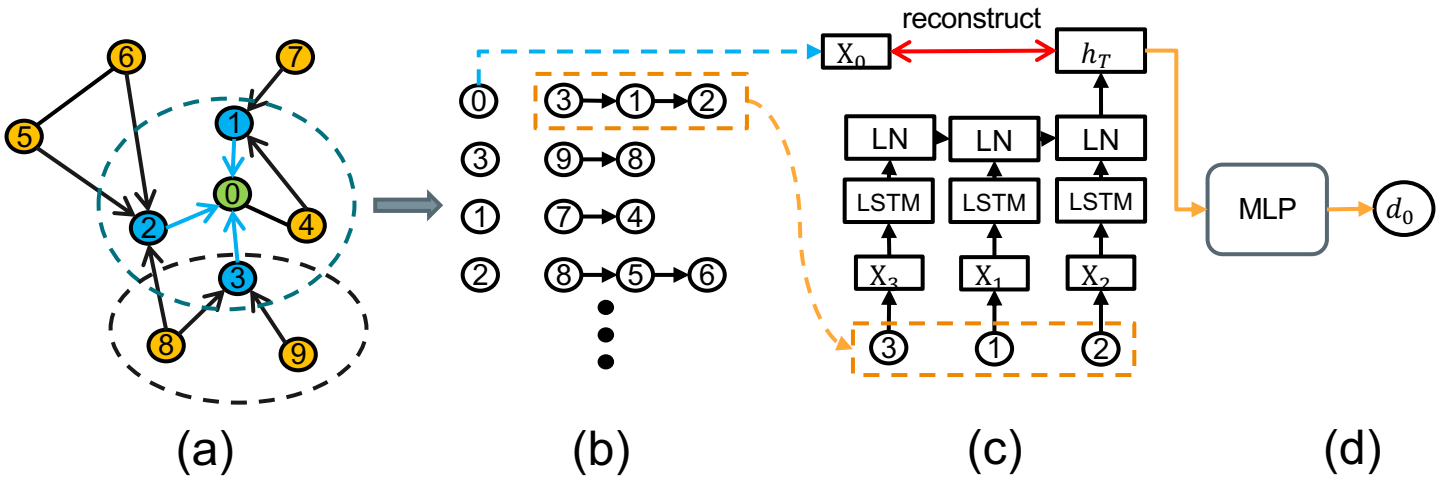
- The definition of regular equivalence is recursive
  - Aggregating neighbors' information in a **recursive** way

$$\mathcal{L}_1 = \sum_{v \in V} ||\mathrm{X}_v - Agg(\{\mathrm{X}_u | u \in \mathcal{N}(v)\})||_F^2,$$

- How to design the aggregating function
  - Variable length of neighbors
  - Highly nonlinear
  - → Layer-normalized LSTM

Ke Tu, et al. Deep Recursive Network Embedding with Regular Equivalence. ***KDD***, 2018.

Deep Recursive Network Embedding



(a) Sampling neighborhoods
(b) Sorting neighborhoods by their degree
(c) Aggregate neighbors
(d) A Weakly guided regularizer

Ke Tu, et al. Deep Recursive Network Embedding with Regular Equivalence. ***KDD***, 2018.

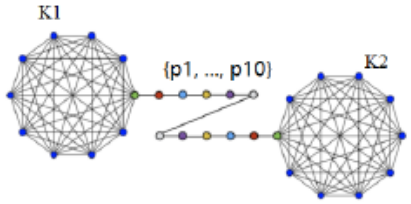
Theoretical Analysis

THEOREM 3.5. *If the centrality $C(v)$ of node v satisfies that $C(v) = \sum_{u \in N(v)} F(u)C(u)$ and $F(v) = f(\{F(u), u \in N(v)\})$ where f is any computable function, then $C(v)$ is one of the optimal solutions of our model.*

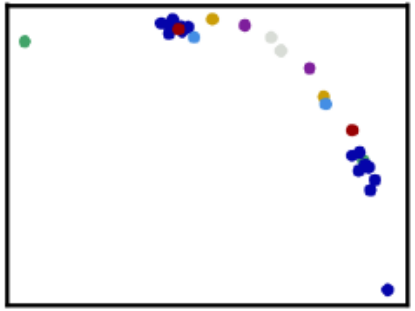
Centrality	Definition $C(v)$	$F(v)$	$f(\{x_i\})$
Degree	$d_v = \sum_{u \in N(v)} I(d_u)$	$1/d_v$	$1/(\sum I(x_i))$
Eigenvector	$1/\lambda * \sum_{u \in N(v)} C(u)$	$1/\lambda$	mean
PageRank	$\sum_{u \in N(v)} 1/d_u * C(u)$	$1/d_v$	$1/(\sum I(x_i))$

Ke Tu, et al. Deep Recursive Network Embedding with Regular Equivalence. ***KDD***, 2018.

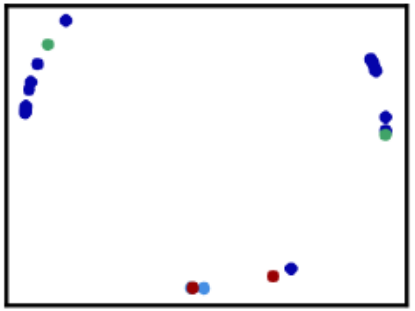
Experiment --- Network Visualization



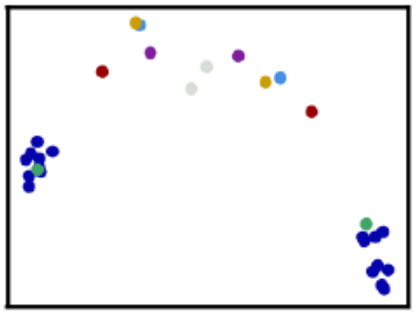
(a) barbell graph



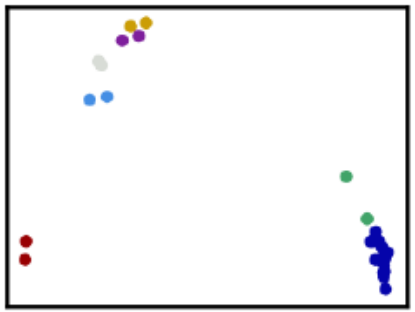
(b) DeepWalk



(c) LINE



(d) node2vec



(e) struc2vec

(f) DRNE

Experiment --- predict centrality

centrality	closeness	betweenness	eignvector	k-core
DeepWalk	0.6016	3.7188	2.1543	13.2755
LINE	0.5153	4.3919	1.5072	15.8179
node2vec	1.0489	3.4065	3.9436	39.2156
struc2vec	0.2365	0.25371	1.0544	9.0858
DRNE	**0.1909**	**0.1261**	**0.5267**	**5.5683**

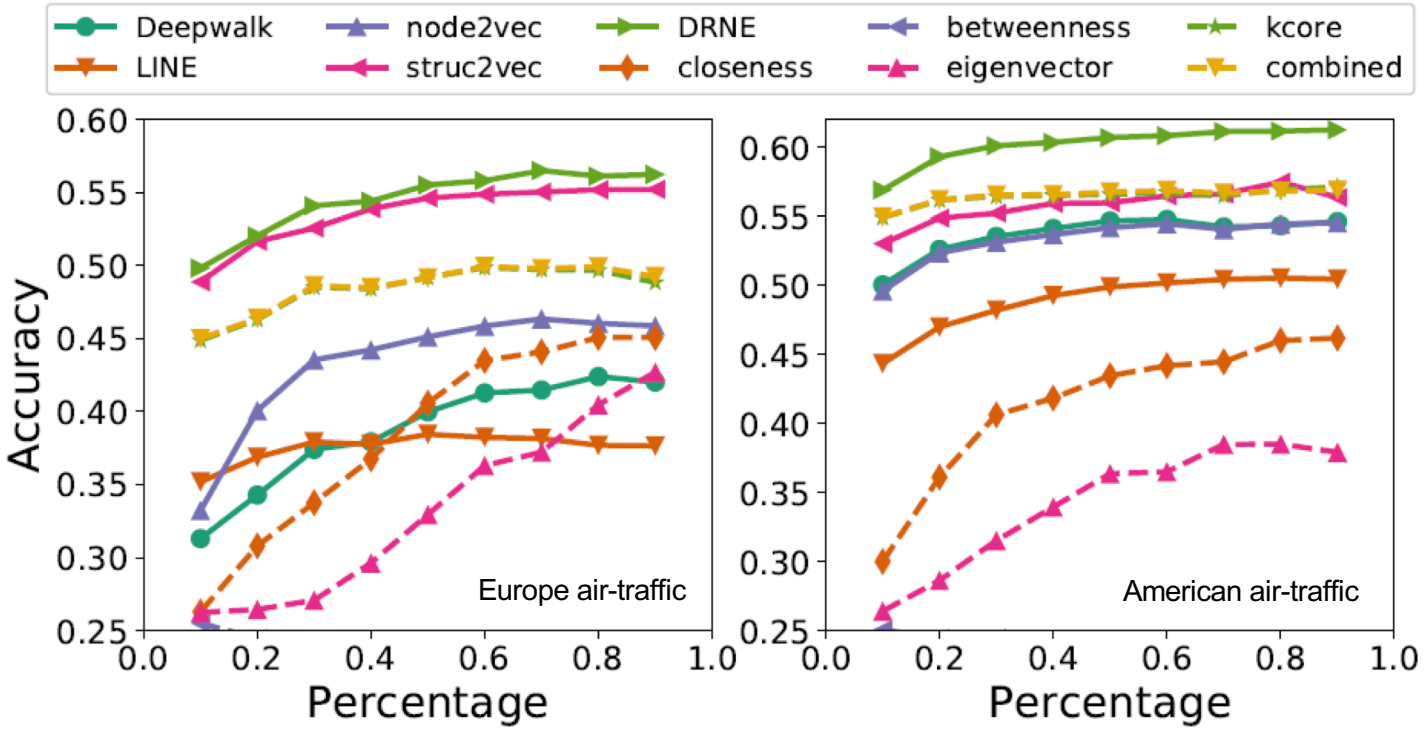
The MSE value of predicting centralities on Jazz dataset (∗10−2)

centrality	closeness	betweenness	eignvector	k-core
DeepWalk	0.2982	1.7836	1.1194	19.7016
LINE	0.3979	1.8425	1.5167	34.9079
node2vec	0.3573	1.6958	1.1432	24.1704
struc2vec	0.2947	1.6018	1.0445	25.3047
DRNE	**0.1101**	**0.6676**	**0.3108**	**7.7210**

The MSE value of predicting centralities on BlogCatalog dataset (∗10−2)

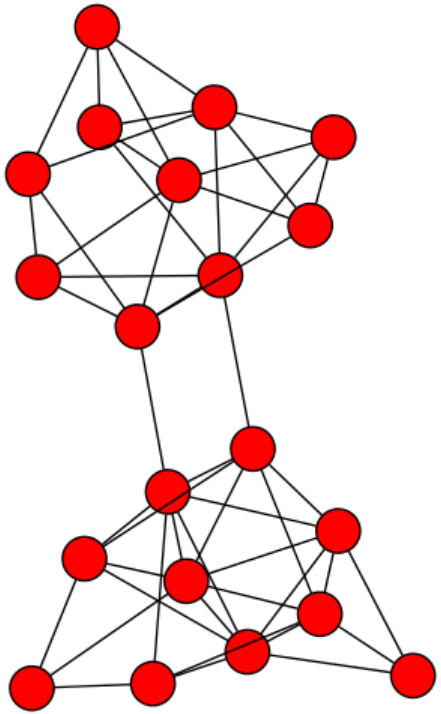
Ke Tu, et al. Deep Recursive Network Embedding with Regular Equivalence. ***KDD***, 2018.

Experiment - Structural Role Classification



Ke Tu, et al. Deep Recursive Network Embedding with Regular Equivalence. ***KDD***, 2018.

Section Summary



Nodes & Links

Node Neighborhood

Pair-wise Proximity

Community Structures

Hyper Edges

Global Structure

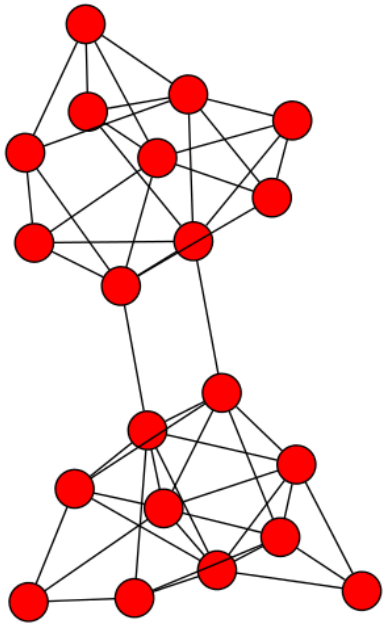
Network Characteristics

Application Characteristics

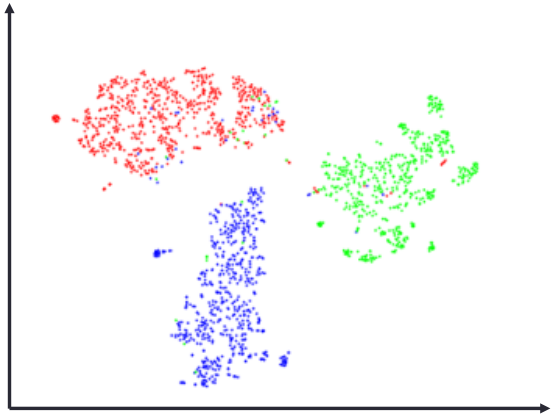
Outline

- Structure-preserved network embedding
- Property-preserved network embedding
- Dynamic network embedding
- Robustness, Explainability and Applicability
- Network embedding for biomedical applications

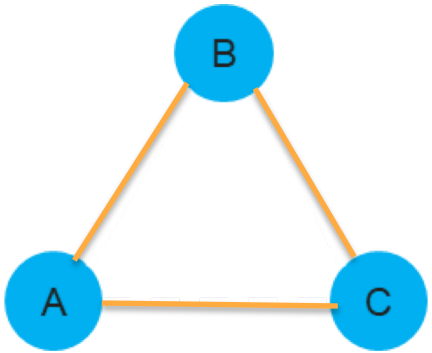
Why preserve network properties?



Heterogeneity



Two important network properties



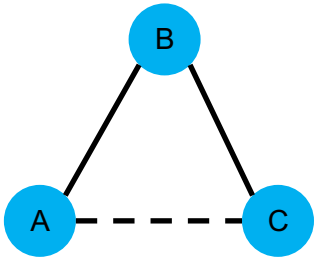
Transitivity

Uncertainty

Transitivity

The Transitivity Phenomenon

Network



Embedding Space

Triangle Inequality: $D(A,B) + D(B,C) > D(A,C)$

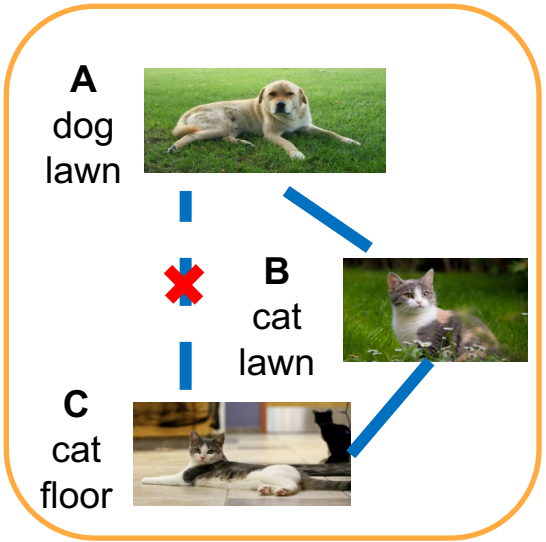
A close to ***B***, ***B*** close to ***C***, $\longrightarrow$ ***A*** relatively close to ***C***

However, real network data is complex…

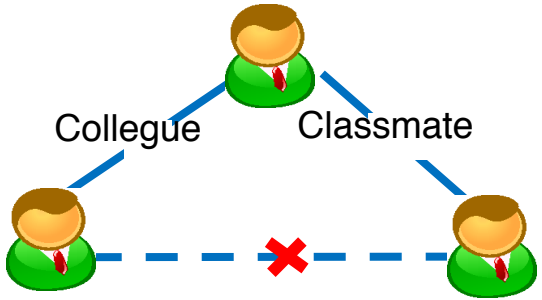
Non-transitivity

The Co-existence of *Transitivity* and *Non-transitivity*

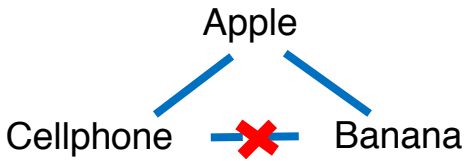
Image network



Social network



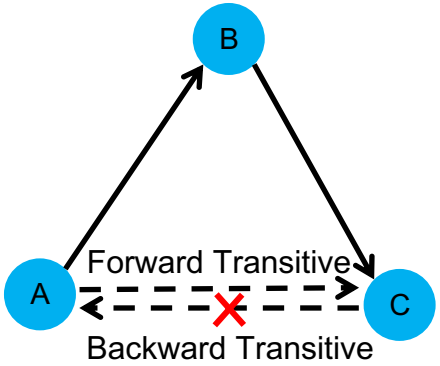
Word network



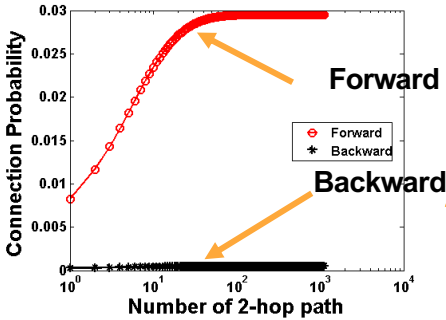
How to incorporate non-transitivity in embedding space?

Asymmetric Transitivity

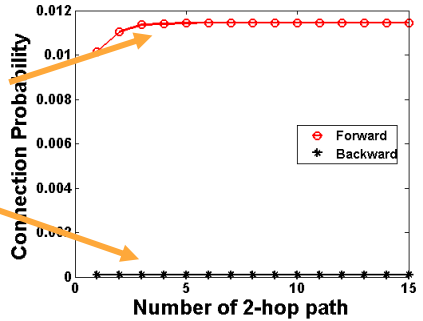
Directed Network



A→B, B → C => A → C, but not C →A



Tencent Microblog



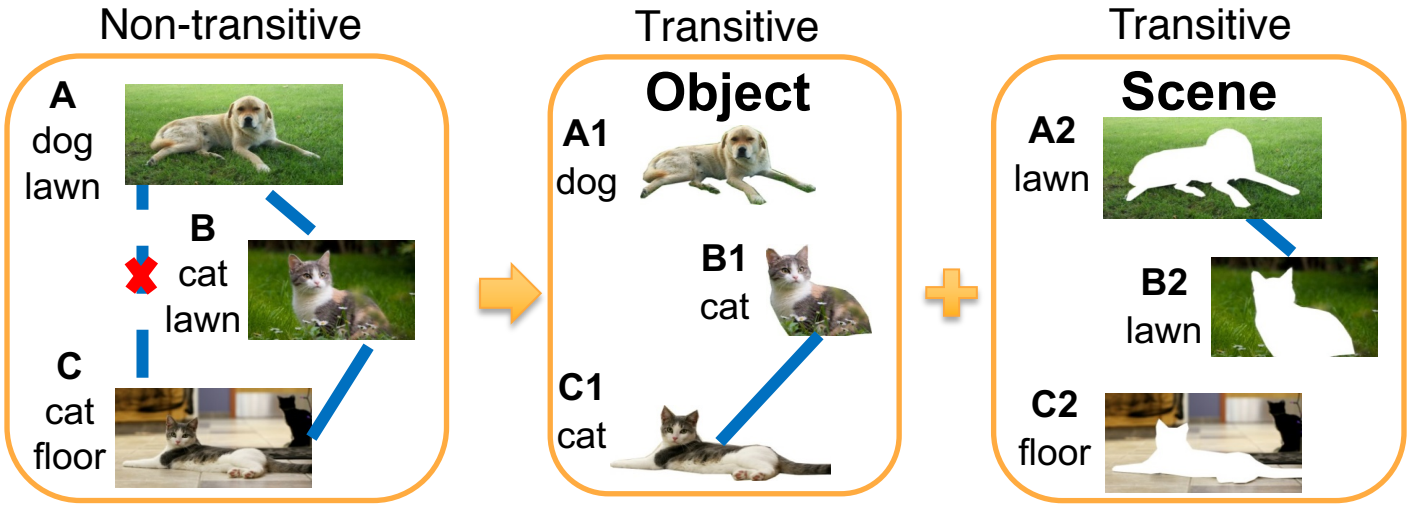
Twitter

Distance metric in embedding space is symmetric.
How to incorporate *Asymmetric Transitivity*?

Non-transitivity

The source of non-transitivity:

Each node has multiple similarity components

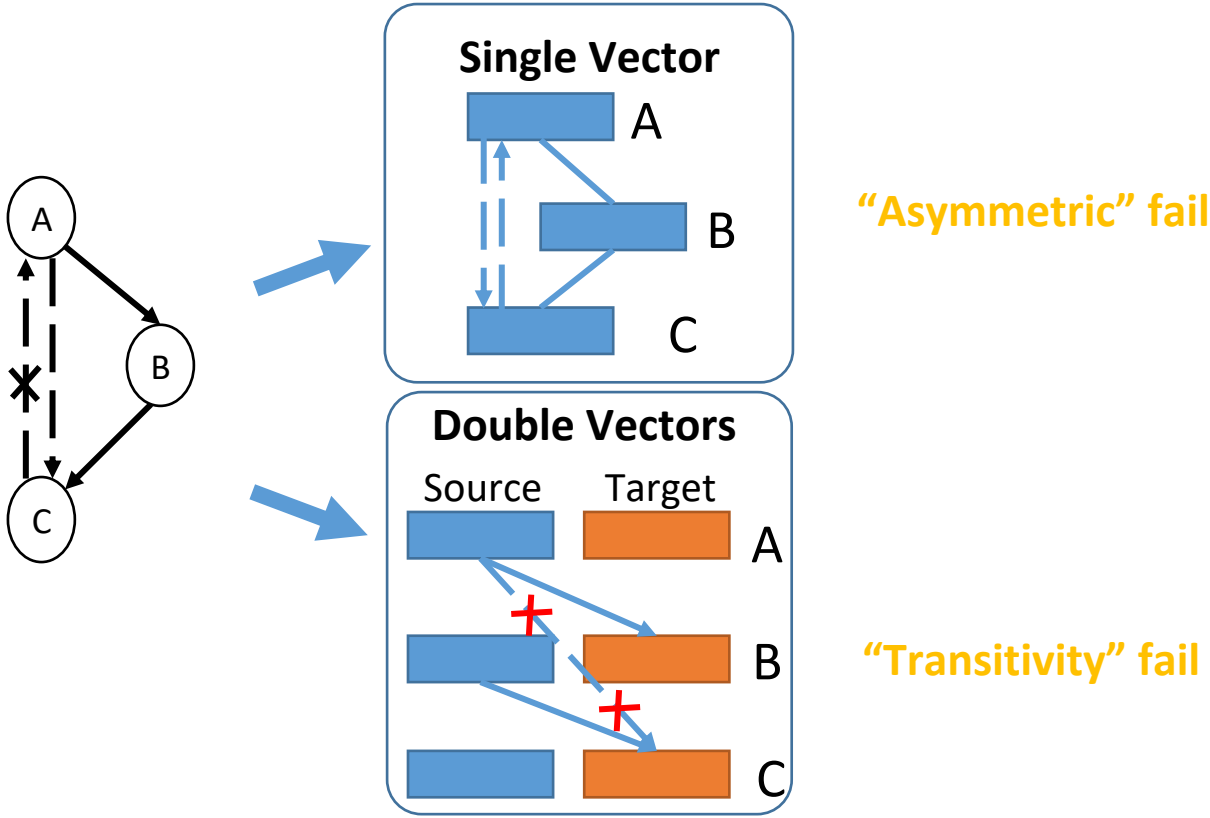


Non-transitive Embedding: represent non-transitive data with multiple latent similarity components

M. Ou, et al. Non-transitive Hashing with Latent Similarity Components. *KDD*, 2015.

Asymmetric Transitivity

All existing methods fail..

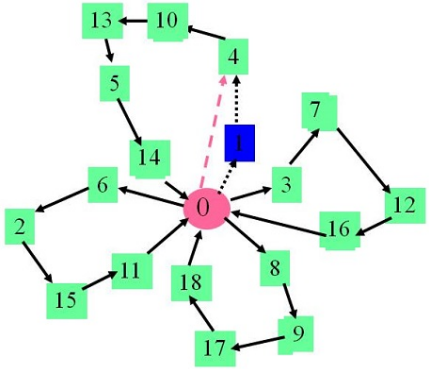


"Asymmetric" fail

"Transitivity" fail

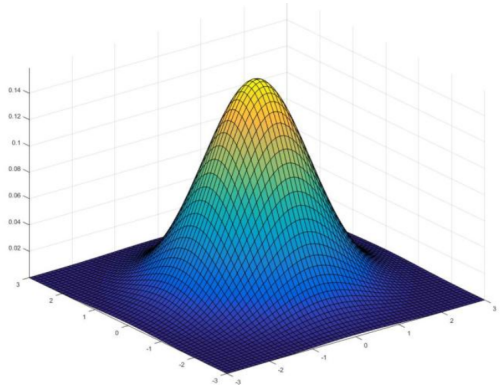
Uncertainties in Networks

- The formation and evolution of real-world networks are full of uncertainties
  - E.g., for the nodes with low degree, they contain less information and thus their representations bear more uncertainties than others.
  - E.g., for the nodes across multiple communities, the possible contradiction between their neighboring nodes may also be large and thus cause the uncertainty.



Uncertainties in Networks

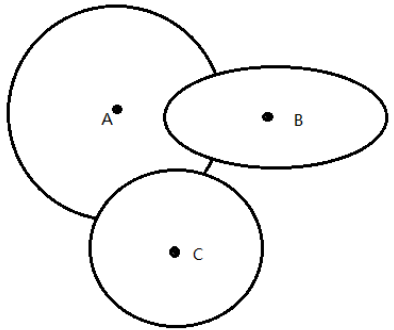
- Learning an embedding as a distribution rather than a point-vector allows us to capture uncertainty of the nodes
  - The mean vectors reflect the position of the nodes and variance terms should contain the uncertainty of the nodes.
  - Gaussian distribution innately represents the uncertainty property[1].



[1] Luke Vilnis and Andrew McCallum. 2014. Word representations via Gaussian embedding. arXiv preprint arXiv:1412.6623 (2014).

Basic requirements

- **Structural Proximities**
- **Transitivity**
- **Uncertainty**
  - The mean vectors should reflect the position of the nodes.
  - The variance terms should contain the uncertainty of the nodes.



DVNE for Structure and Uncertainty

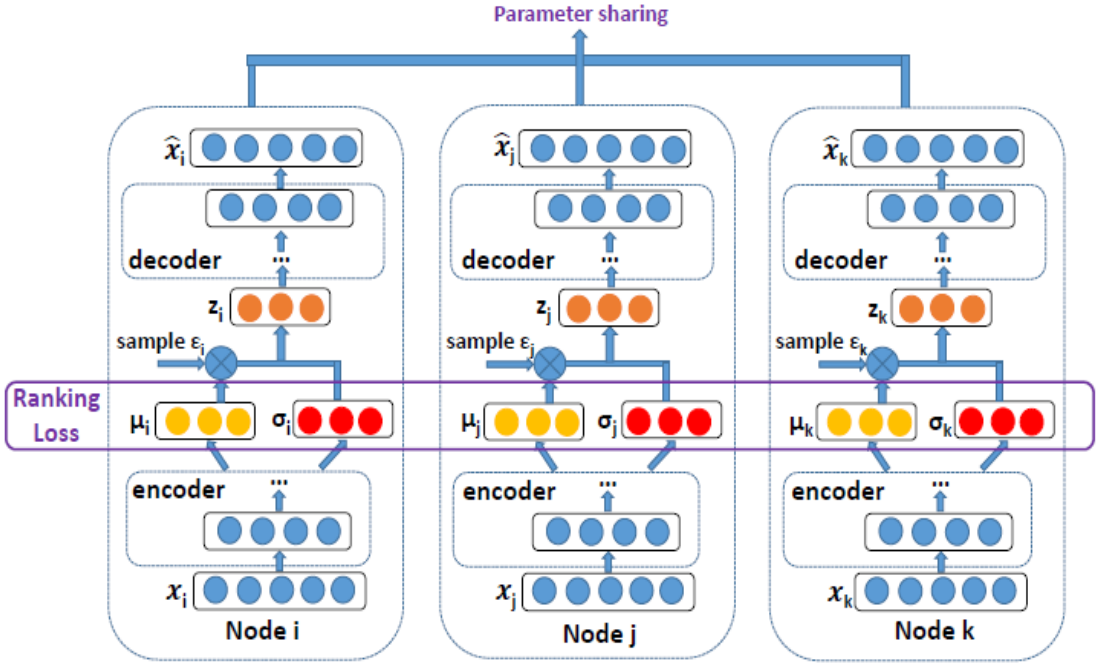


Figure 1: The framework of DVNE.

Dingyuan Zhu, et al. Deep Variational Network Embedding in Wasserstein Space. ***KDD***, 2018.

Section Summary

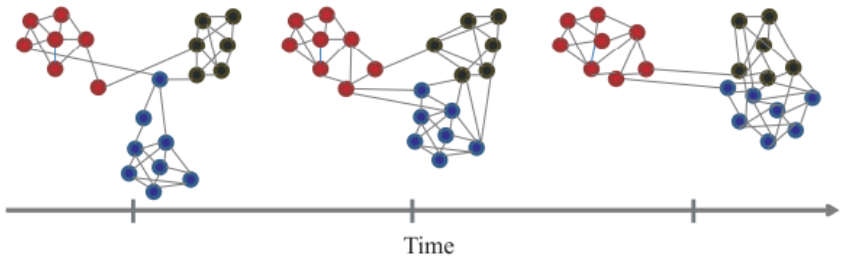
- Compared with network structures, **network properties** have large space to explore in network embedding.
- Transitivity is important for network inference.
- Uncertainty provides evidence in making network inference.
- Many other property issues:
  - The right embedding space: Euclidean space?
  - Power-law distribution
  - …

Outline

- Structure-preserved network embedding
- Property-preserved network embedding
- Dynamic network embedding
- Robustness, Explainability and Applicability
- Network embedding for biomedical applications

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?**

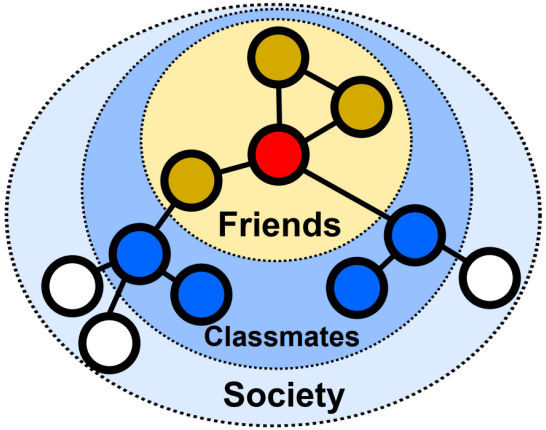


Key problems in dynamic network embedding

- **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

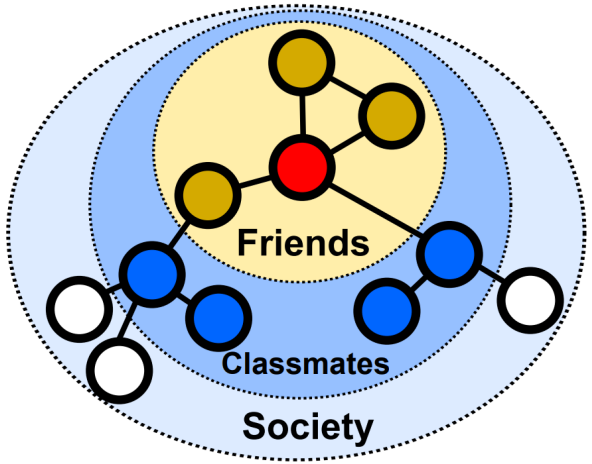
Challenge: High-order Proximity

I : Out-of-sample nodes

II : Incremental edges

III: Aggregated error

IV: Scalable optimization



Preserve High-order Proximities

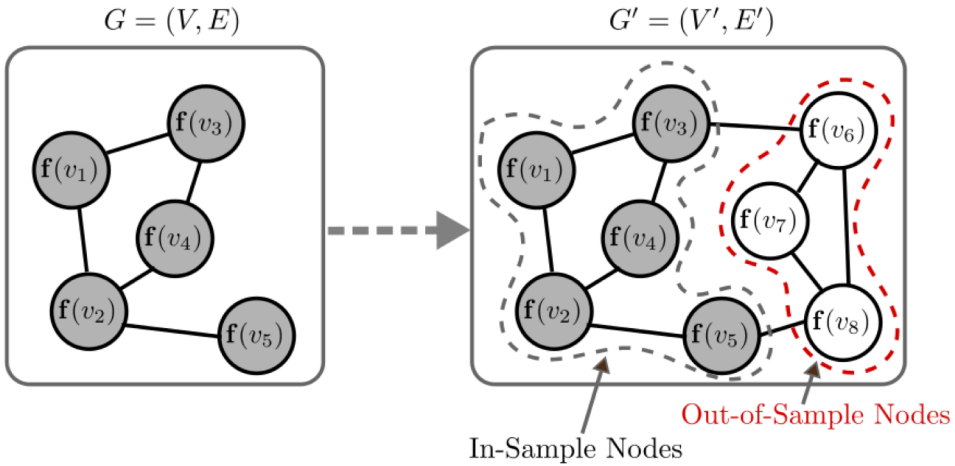
Local Change leads to Global Updating

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 ∪ V*.
- n old nodes: V = {v1,…,vn}, m new nodes: V* = {vn+1,…,vn+m}
- Network embedding: f: V→Rd
- We know f(v) for old nodes, want to infer 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

Jianxin Ma, et al. DepthLGP: Learning Embeddings of Out-of-Sample Nodes in Dynamic Networks. ***AAAI***, 2018.

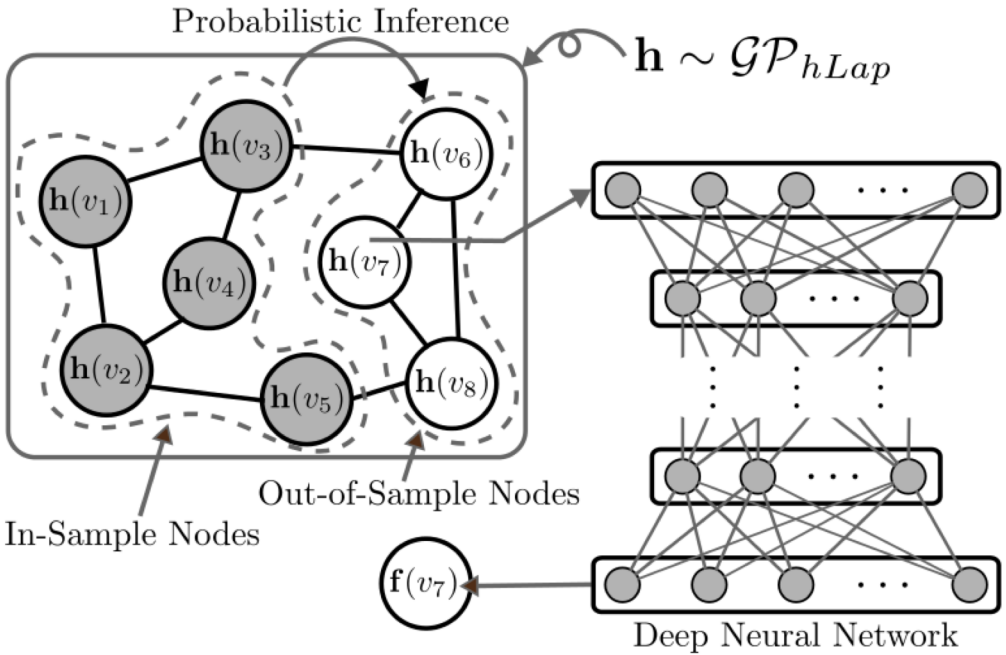
Specific vs. General

- 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).

Jianxin Ma, et al. DepthLGP: Learning Embeddings of Out-of-Sample Nodes in Dynamic Networks. ***AAAI***, 2018.

DepthLGP

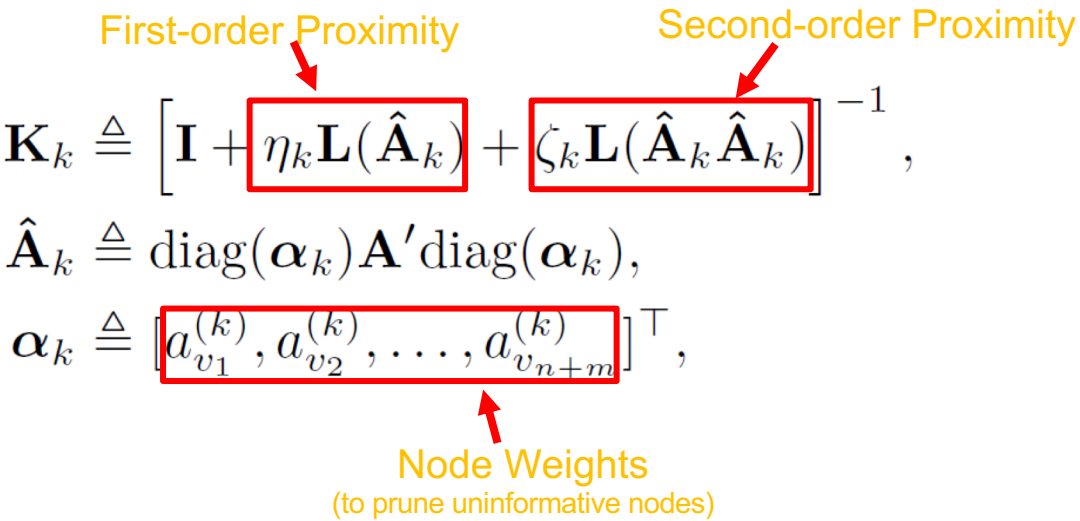
- Nonparametric probabilistic modeling + Deep Learning



Jianxin Ma, et al. DepthLGP: Learning Embeddings of Out-of-Sample Nodes in Dynamic Networks. ***AAAI***, 2018.

DepthLGP

- Design a kernel for the *k*th (*k=1,…,s*) dimension of $h(\cdot)$



[1]The matrix inversion can be bypassed without approximation.

[2]$a^{(k)}_v$ indicates how much attention we pay to a node. It is learned for an in-sample node, but fixed to one for an OOS node, as we are always interested in OOS nodes.

Task I: Classification

			Baselines			This Work		Upper Bound
Metric	Embedding	Network	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)

Key problems in dynamic network embedding

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

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.

Dingyuan Zhu, et al. High-order Proximity Preserved Embedding For Dynamic Networks. ***IEEE TKDE***, 2018.

GSVD

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

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

  - 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

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 ΔA between two consecutive time steps, the change of Ma and Mb can be represented as:

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

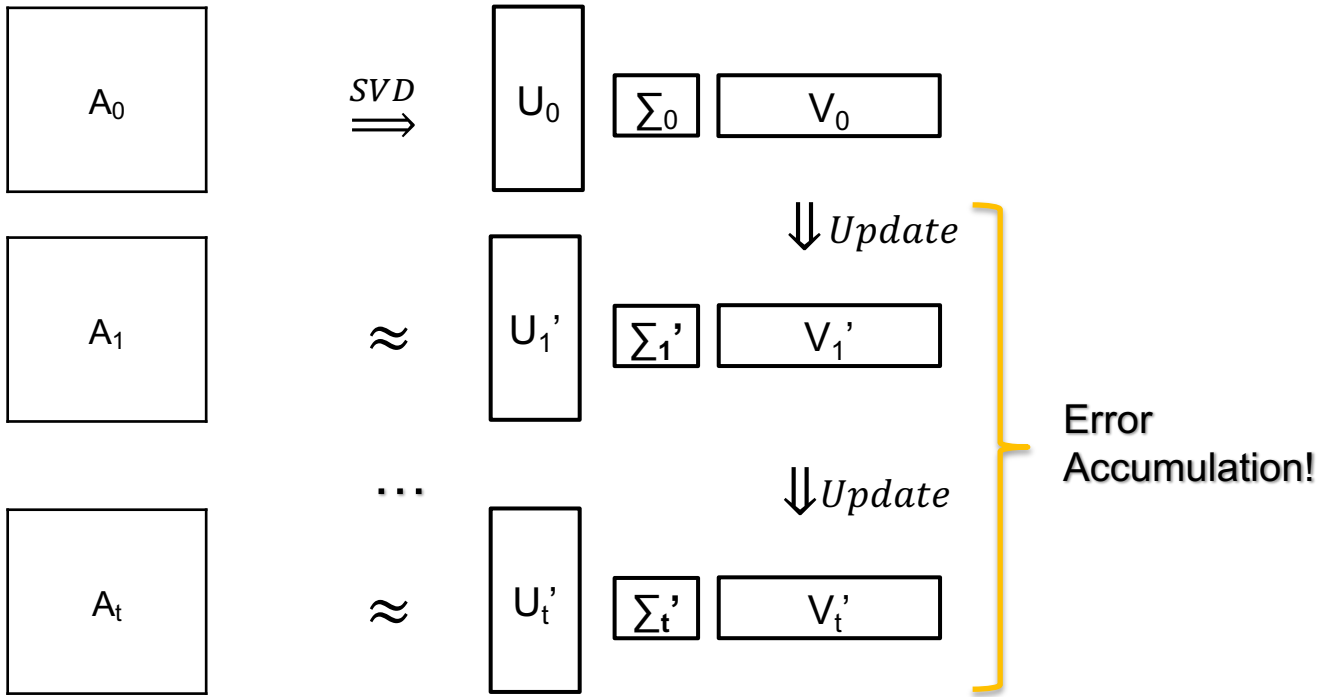
Dingyuan Zhu, et al. High-order Proximity Preserved Embedding For Dynamic Networks. ***IEEE TKDE***, 2018.

Key problems in dynamic network embedding

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

Problem: Error Accumulation

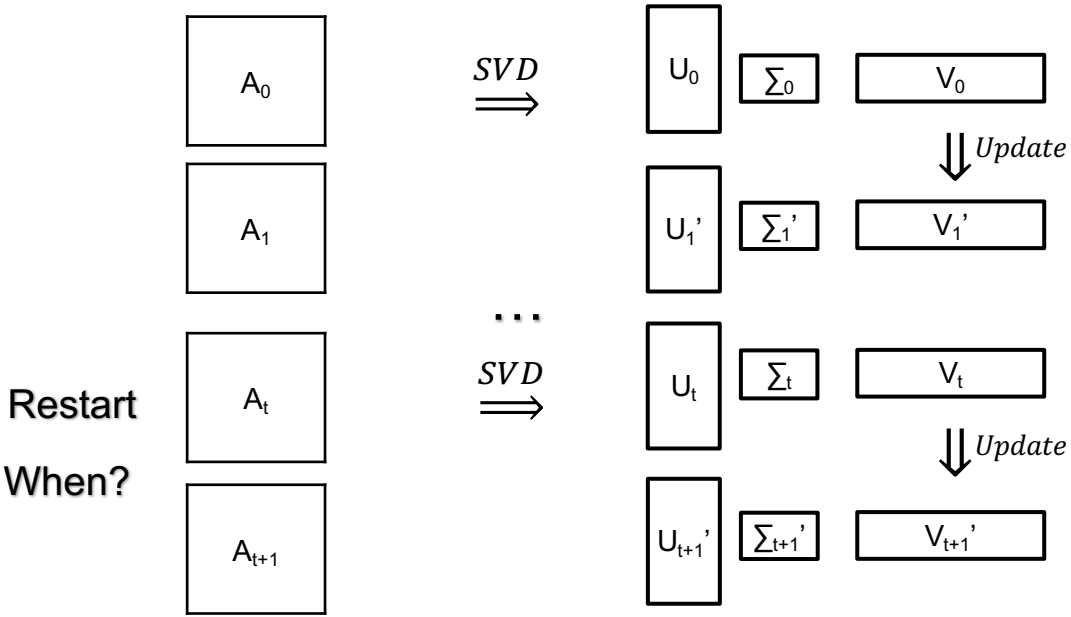
- Eigen perturbation is at the cost of inducing approximation



- Problem: error accumulation is inevitable

Solution: SVD Restarts

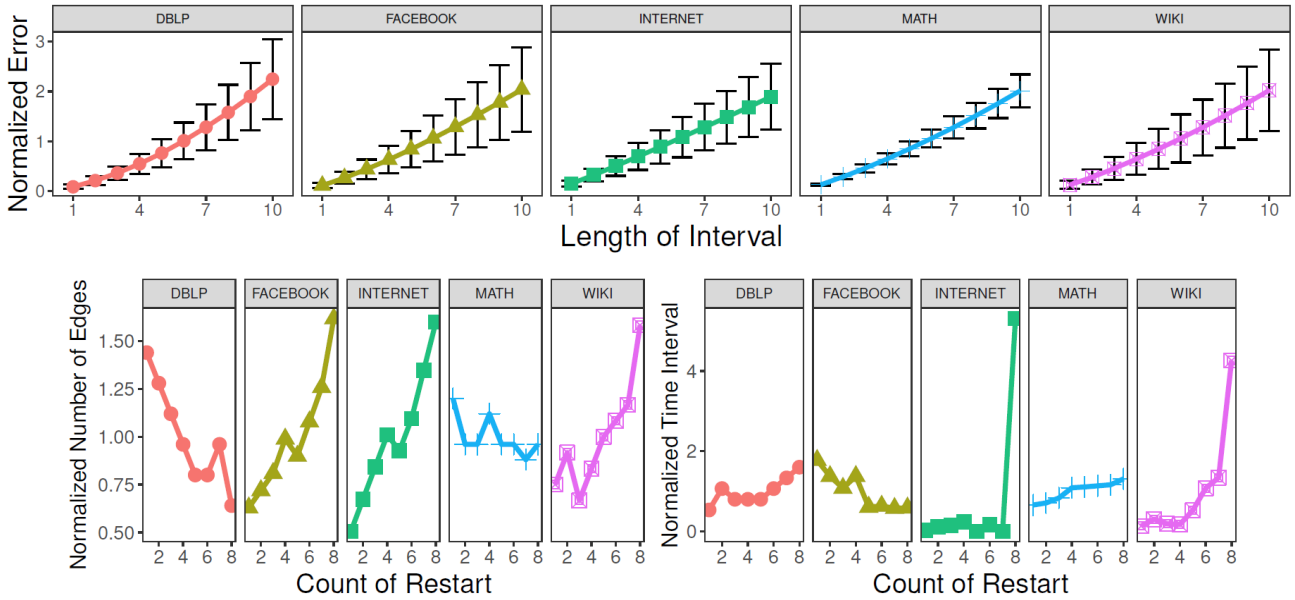
- Solution: restart SVD occasionally



- 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



Ziwei Zhang, et al. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. ***AAAI***, 2018.

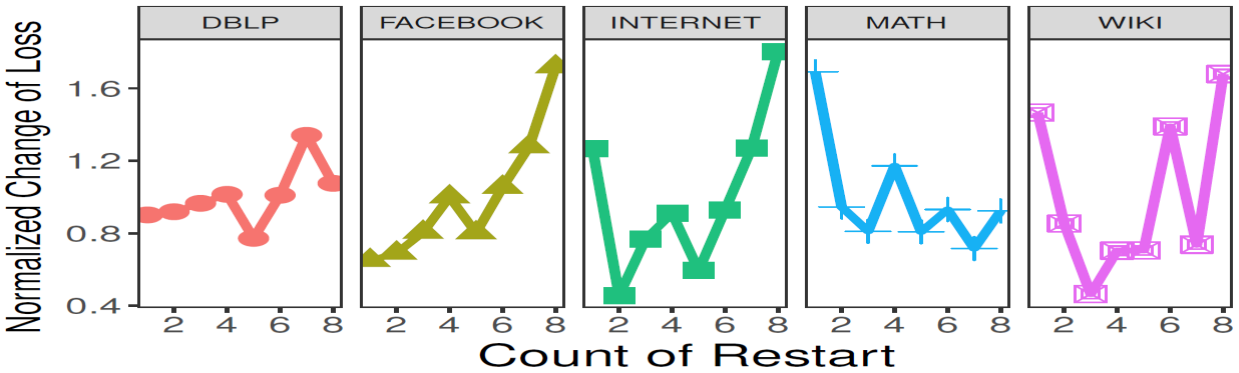
Existing Method

- 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

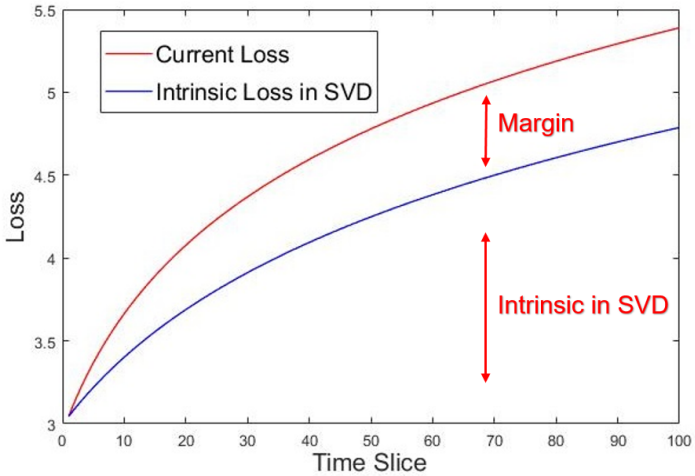
- Problem: loss includes approximation error and intrinsic loss in SVD



Ziwei Zhang, et al. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. ***AAAI***, 2018.

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, et al. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. ***AAAI***, 2018.

Solution: Lazy Restarts

- Lazy restarts: restart only when the margin exceeds the threshold
- Problem: intrinsic loss is hard to compute
  - Direct calculation has the same time complexity as SVD
- Relaxation: an upper bound on margin
  - A lower bound on intrinsic loss $\mathcal{L}(S,k)$

$$\mathcal{L}(\mathbf{S}_t, k) \geq B(t) \Rightarrow \frac{\mathcal{J}(t) - \mathcal{L}(\mathbf{S}_t, k)}{\mathcal{L}(\mathbf{S}_t, k)} \leq \frac{\mathcal{J}(t) - B(t)}{B(t)}.$$

$\mathcal{J}(t)$: current loss; $\mathcal{L}(S\_t,k)$: intrinsic loss; $B(t)$: bound of intrinsic loss

Ziwei Zhang, et al. 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

Ziwei Zhang, et al. 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 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, et al. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. ***AAAI***, 2018.

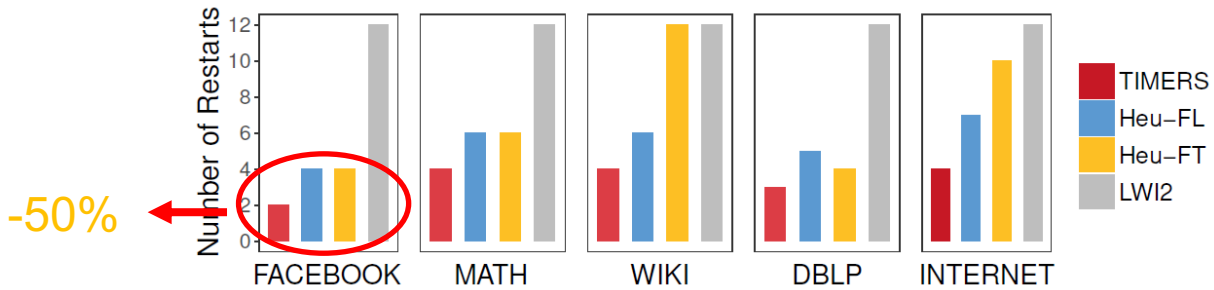
Experimental Results: Approximation Error

- 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

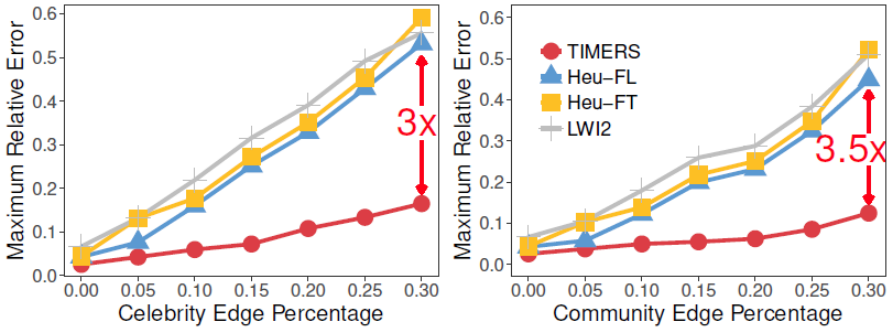
- Fixing maximum error



Ziwei Zhang, et al. TIMERS: Error-Bounded SVD Restart on Dynamic Networks. ***AAAI***, 2018.

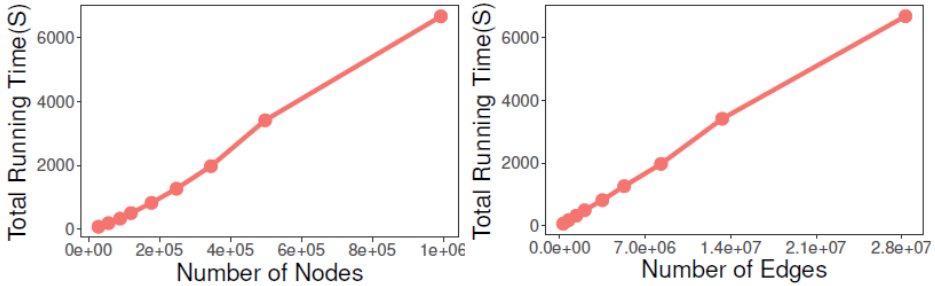
Experimental Results: Analysis

- Syntactic networks: simulate drastic changes in the network structure



  - Robust to sudden changes

- Linear scalability

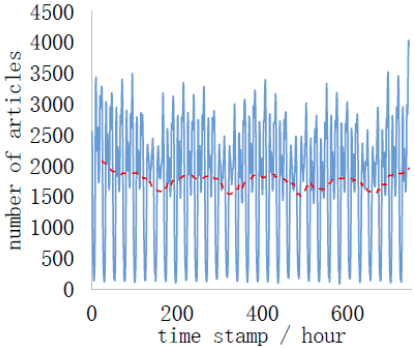


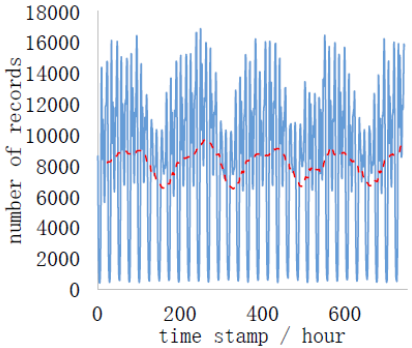
Key problems in dynamic network embedding

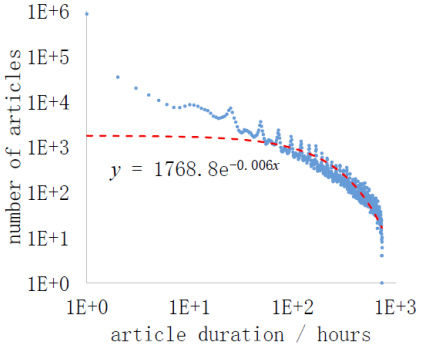
- I : Out-of-sample nodes
- II : Incremental edges
- III: Aggregated error
- IV: Scalable optimization

Highly-dynamic & Recency-sensitive Data

- News recommendation applications: a bipartite graph
- WeChat news recommendation network is highly dynamic
  - 81 articles and 1400 reading records per second
- The network is also recency-sensitive
  - >73% articles died less than 6 hours while no one read again
  - Obvious exponential decay for article duration length.







Xumin Chen, et al. Scalable Optimization for Embedding Highly-Dynamic and Recency-Sensitive Data. ***KDD***, 2018.(Applied)

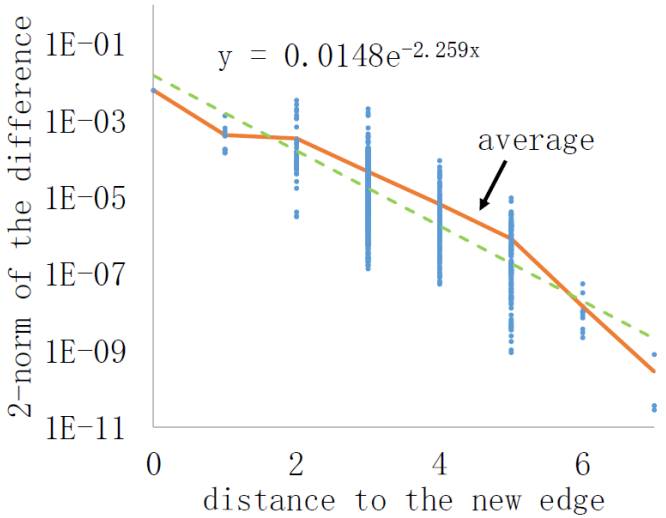
Limited resources

- 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.

Diffused SGD: Step-wise Weight Diffusion Mechanism

- The Change of a node embedding vector depends on its distance to the changed edge.
- Diffuse across **training steps**
- 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 $(l, k) \in \mathbb{E} \wedge l \neq i$,

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

Xumin Chen, et al. Scalable Optimization for Embedding Highly-Dynamic and Recency-Sensitive Data. ***KDD***, 2018.(Applied)

Section Summary

- **I : Out-of-sample nodes**
  - **DepthLGP = Non-parametric GP + DNN**
- **II : Incremental edges**
  - **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**

Outline

- Structure-preserved network embedding
- Property-preserved network embedding
- Dynamic network embedding
- Robustness, Explainability and Applicability
- Network embedding for biomedical applications

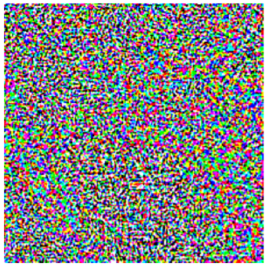
Technical challenges in real applications

Research → Application

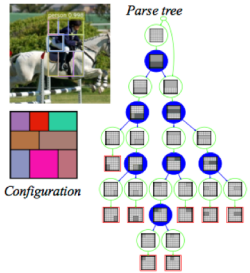
- Robustness
- Interpretability
- Applicability

Hot directions in computer vision:

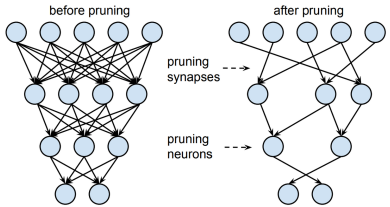
Adversarial



Explainable

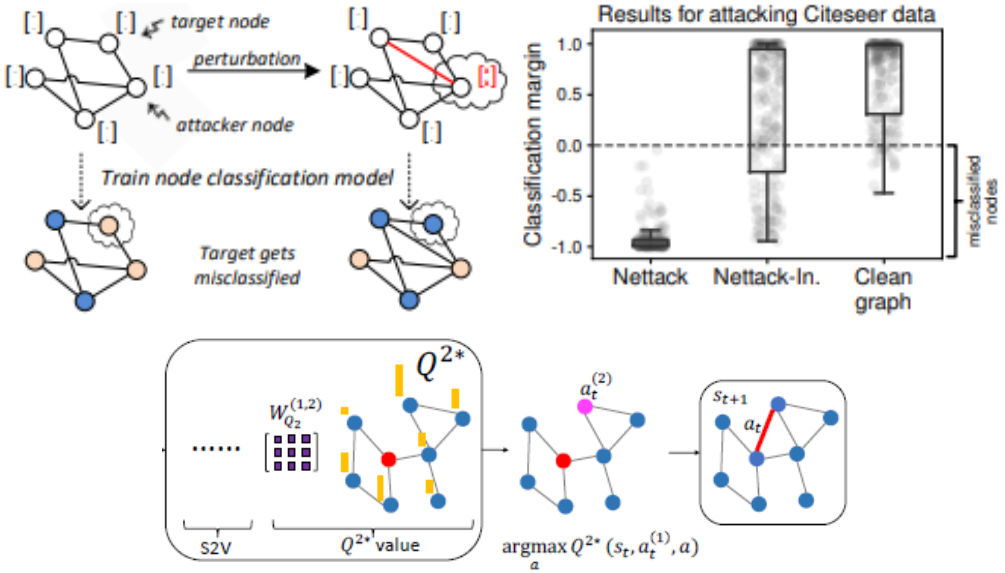


Scalable



Robustness in network embedding

- **Adversarial attacks**
  - small perturbations in graph structures and node attributes
  - great challenges for applying GCNs to node classification

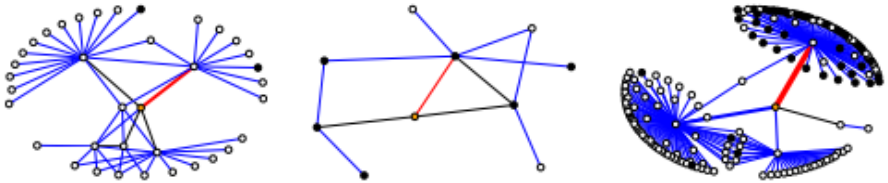


Adversarial Attacks on GCNs

- **Categories**
  - Targeted vs Non-targeted
    - Targeted: the attacker focus on misclassifying some target nodes
    - Non-targeted: the attacker aims to reduce the overall model performance
  - Direct vs Influence
    - Direct: the attacker can directly manipulate the edges or features of the target nodes
    - Influence: the attacker can only manipulate other nodes except the targets
- How to enhance the robustness of GCNs against adversarial attacks?

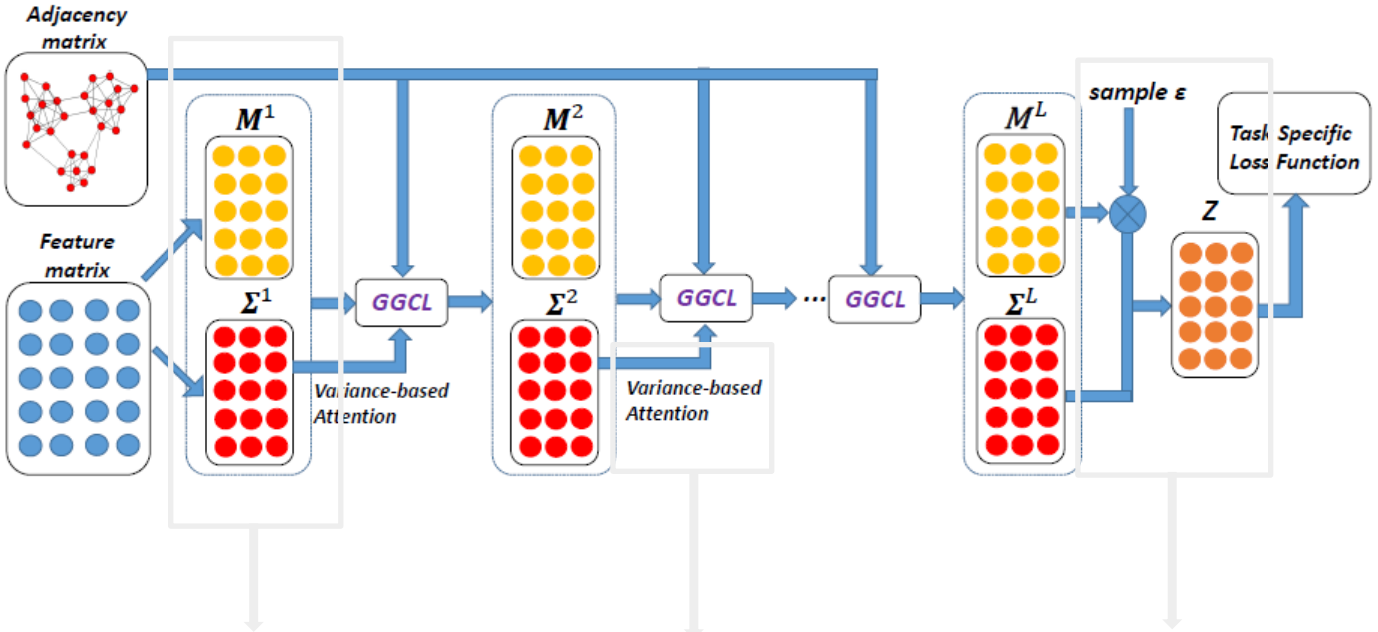
Robust Graph Convolutional Networks

- Adversarial attacks in node classification
  - Connect nodes from different communities to confuse the classifier



- Distribution V.S. plain vectors
  - Plain vectors cannot adapt to such changes
  - Variances can help to absorb the effects of adversarial changes
  - Gaussian distributions -> Hidden representations of nodes

The Framework of RGCN



Gaussian Based hidden representations:

Variance terms absorb the effects of adversarial attacks

Attention mechanism:

Remedy the propagation of adversarial attacks

Sampling process:

Explicitly considers mathematical relevance between means and variances

Experimental Results

☐ Node Classification on Clean Datasets

	Cora	Citeseer	Pubmed
GCN	81.5	70.9	79.0
GAT	83.0	72.5	79.0
RGCN	**83.1**	71.3	**79.2**

☐ Against Non-targeted Adversarial Attacks

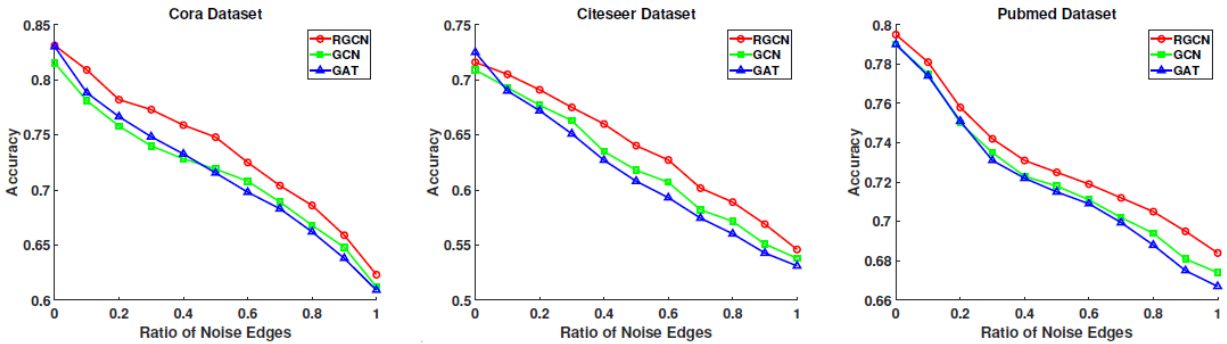
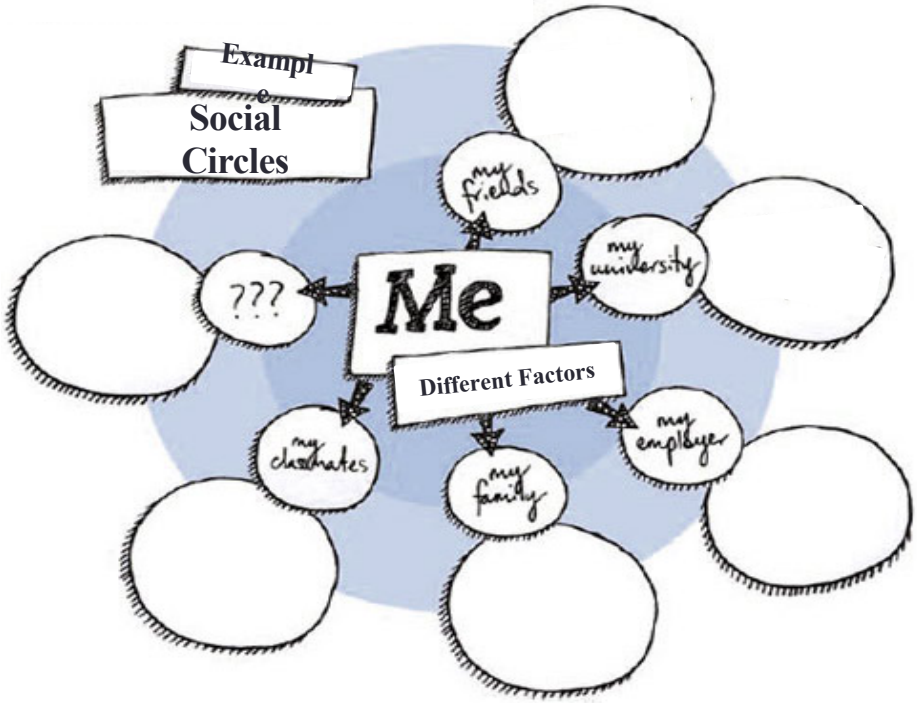


Figure 2: Results of different methods when adopting Random Attack as the attack method.

Interpretability of network embedding

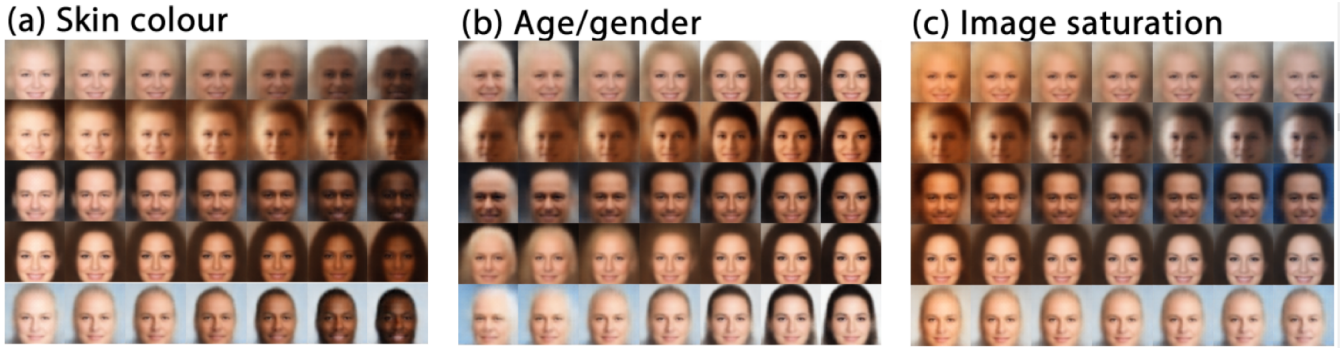
- **A real-world graph is typically formed due to *many* latent factors.**



- Existing GNNs/GCNs:
  - A holistic approach, that takes in the *whole* neighborhood to produce a *single* node representation.
- We suggest:
  - To disentangle the latent factors. (By segmenting the heterogeneous parts, and learning multiple factor-specific representations for a node.)
  - Robustness (e.g., not overreact to an irrelevant factor) & Interpretability.

Disentangled Representation Learning

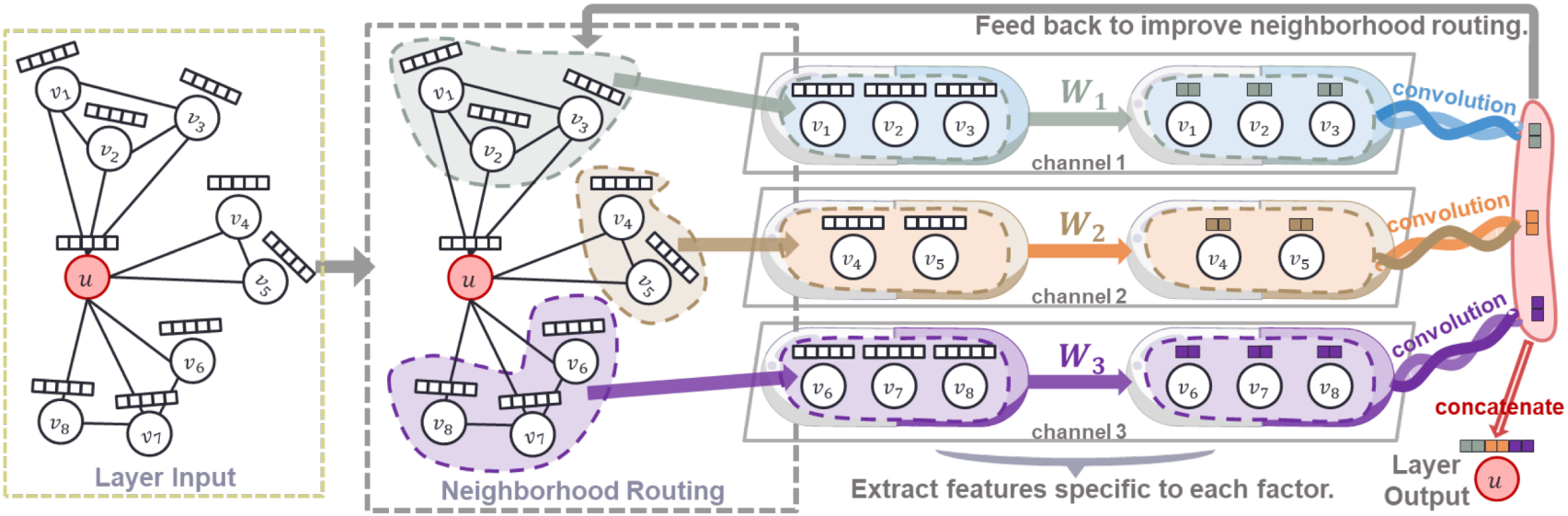
- That is, we aim to learn disentangled node representation,
  - A representation that contains independent components, that describes different aspects (caused by different latent factors) of the observation.
- The topic is well research in the field of computer vision.
  - But largely unexplored in the literature of GNNs.



Example: Three dimensions that are related skin color, age/gender, and saturation, respectively.

Method Overview

- We present DisenGCN, the *disentangled* graph convolutional network.
  - DisenConv, a disentangled multichannel convolutional layer (figure below).
  - Each channel convolutes features related with a single latent factor.



DisenConv: Multiple Channels

- The DisenConv layer $f(\cdot)$ takes in the neighborhood of a node, and outputs the node's convoluted representation.

$$\mathbf{y}_u = f\left(\mathbf{x}_u, \{\mathbf{x}_v : (u, v) \in G\}\right)$$

- It has K channels. The output is concatenated from the outputs of the channels.

$$\mathbf{y}_u = \left[\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\right]$$

- Each channel has its own projection matrix for extracting aspect-specific features. Given a node i, the extracted feature regarding aspect k is:

$$\mathbf{z}_{i,k} = \frac{\sigma(\mathbf{W}_k^\top \mathbf{x}_i + \mathbf{b}_k)}{\left\|\sigma(\mathbf{W}_k^\top \mathbf{x}_i + \mathbf{b}_k)\right\|_2}$$

Neighborhood Routing: Hypothesis I

- A neighbor is *patched to channel k* (for further in-channel graph convolution), if the edge between the neighbor and the center node is *caused by factor k*.
- But the actual causes are unknown. *Neighborhood routing* is therefore proposed to infer the latent causes, based on two hypothesis.

- The first is analogous to the second-order proximity.

> **Hypothesis 1.** Factor k is likely to be the reason why node u connects with a certain subset of its neighbors, if the subset is large and the neighbors in the subset are similar w.r.t. aspect k, i.e., they form a cluster in the k^{th} subspace.

- It inspires us to search for the biggest cluster in each of the K subspaces.

Neighborhood Routing: Hypothesis II

- The second hypothesis is analogous to the first-order proximity.

> **Hypothesis 2.** Factor k is likely to be the reason why node u and neighbor v are connected, if the two are similar in terms of aspect k.

- Hypothesis 2 is not robust if either $\boldsymbol{x}_u$ or $\boldsymbol{x}_v$ misses features about aspect k, and therefore must be combined with Hypothesis 1. But it can provide a fast guess.

Neighborhood Routing: Hypothesis I + Hypothesis II

- Let $p_{v,k}$ be the probability that factor k is the reason why node u reaches neighbor v, which should satisfies $p_{v,k} \geq 0$ and $\sum_{k'=1}^{K} p_{v,k'} = 1$.

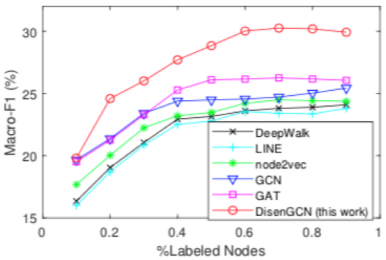
- Initialization based on Hypothesis 1 (for fast convergence):

$$p_{v,k}^{(1)} \propto \exp(\mathbf{z}_{v,k}{}^{\top} \mathbf{z}_{u,k})$$

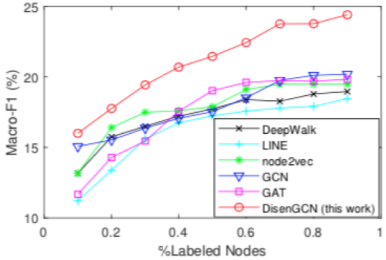
- Iterate for T steps, based mainly on Hypothesis 2:

$$\mathbf{c}_k^{(t)} = \frac{\mathbf{z}_{u,k} + \sum_{v:(u,v)\in G} p_{v,k}^{(t-1)} \mathbf{z}_{v,k}}{\left\| \mathbf{z}_{u,k} + \sum_{v:(u,v)\in G} p_{v,k}^{(t-1)} \mathbf{z}_{v,k} \right\|_2}, \qquad p_{v,k}^{(t)} = \frac{\exp(\mathbf{z}_{v,k}{}^{\top} \mathbf{c}_k^{(t)})}{\sum_{k'=1}^{K} \exp(\mathbf{z}_{v,k'}{}^{\top} \mathbf{c}_{k'}^{(t)})},$$

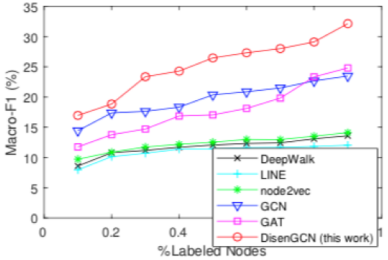
Results: Multi-label Classification



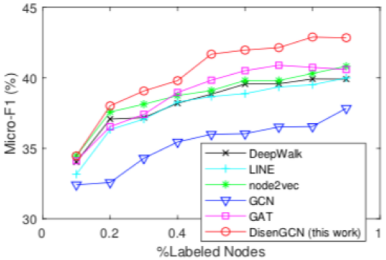
(a) Macro-F1(%), BlogCatalog.



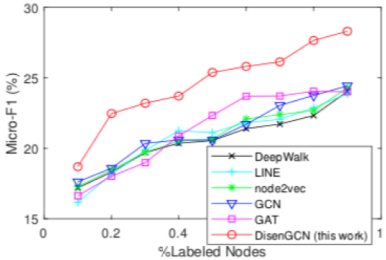
(c) Macro-F1(%), PPI.



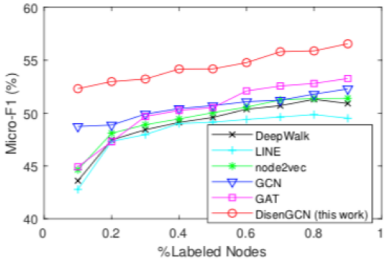
(e) Macro-F1(%), POS.



(b) Micro-F1(%), BlogCatalog.



(d) Micro-F1(%), PPI.



(f) Micro-F1(%), POS.

Figure 2. Macro-F1 and Micro-F1 scores on the multi-label classification tasks. Our approach consistently outperforms the best performing baselines by a large margin, reaching 10% to 20% relative improvement in most cases.

*Results on Cora/Citeseer/Pubmed are only slightly better than GAT. See the paper for explanations.

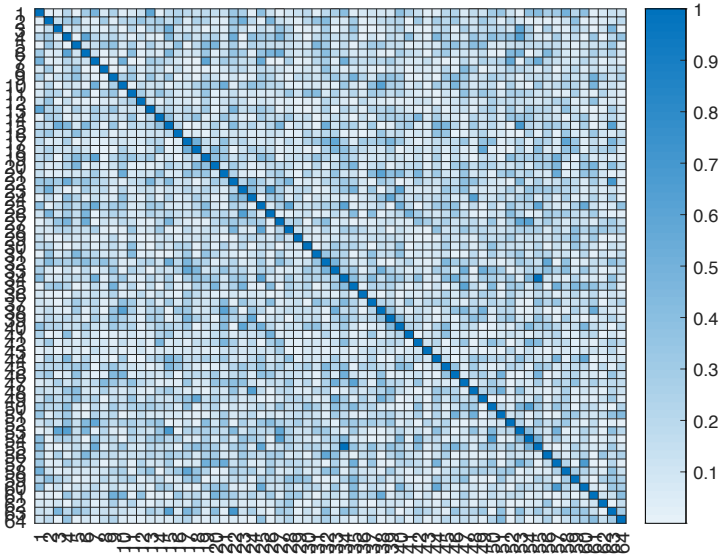
Results: On Synthetic Graphs

Table 3. Micro-F1 scores on synthetic graphs generated with different numbers of latent factors.

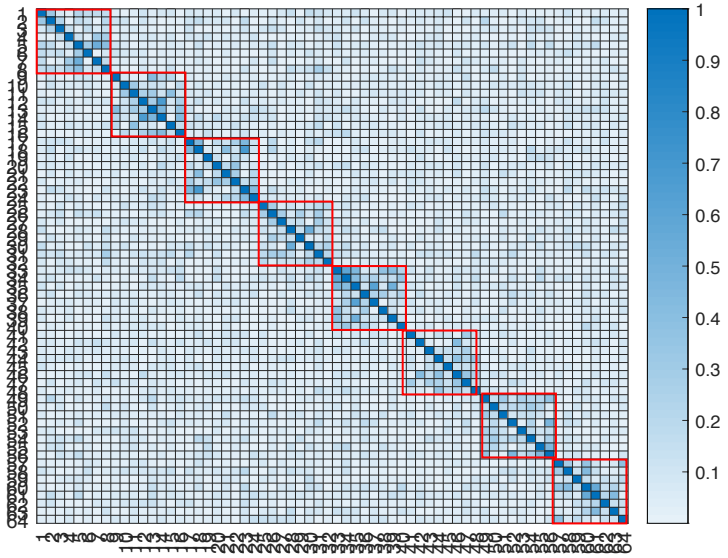
	Number of latent factors						
Method	4	6	8	10	12	14	16
GCN	78.78 ± 1.52	65.73 ± 1.94	46.55 ± 1.55	37.37 ± 1.52	24.49 ± 1.03	18.14 ± 1.50	16.43 ± 0.92
GAT	83.77 ± 2.32	60.89 ± 3.75	45.88 ± 3.79	36.72 ± 3.58	24.77 ± 3.47	20.89 ± 3.57	19.53 ± 3.97
DisenGCN (this work)	**93.84** ± 1.12	**74.68** ± 1.92	**54.57** ± 1.79	**43.96** ± 1.45	**28.17** ± 1.22	**23.57** ± 1.28	**21.99** ± 1.34
Relative improvement	+12.02%	+13.62%	+17.23%	+17.63%	+13.73%	+12.83%	+12.6%

- Improvement is larger when #factors is relatively large (around 8).
- However, all methods are bad when #factors is extremely large.

Results: Correlations between the Neurons

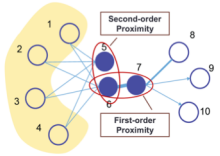


(a) GCN.

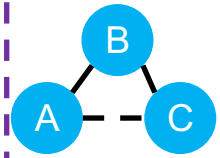


(b) DisenGCN (this work).

Applicability of network embedding



High-order



Transitivity

Global position

Various network properties

- ☐ Link Prediction
- ☐ Community Detection
- ☐ Node Classification
- ☐ Network Distance
- ☐ Node Importance
- ☐ …

Various applications

- Leading to a large number of hyperparameters
- Must be carefully tuned

→ AutoML

AutoML

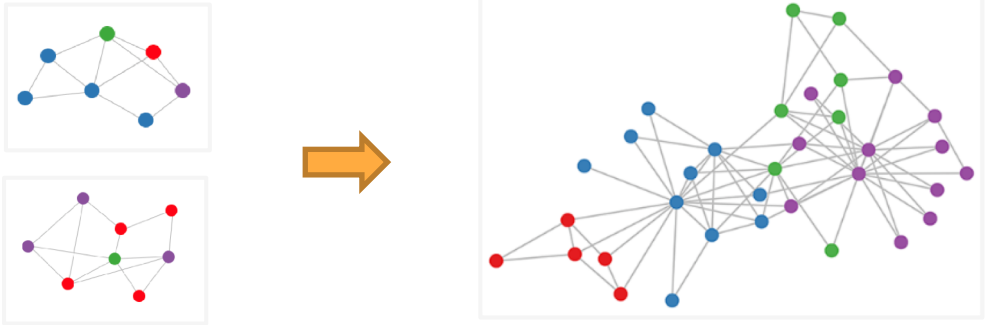
- Ease the adoption of machine learning and reduce the reliance on human experts
  - e.g., hyperparameter optimization
- Largely unexplored on network data

- Large scale issue:
  - Complexity of Network Embedding is usually at least $O(E)$
    - E is the number of edges (can be 10 billion)
  - Total complexity: $O(EL)$, T is the times searching for optimal hyperparameter

How to incorporate AutoML into massive network embedding efficiently? (reduce E and T)

AutoML for network embedding

- A straightforward way: configuration selection on sampled sub-networks



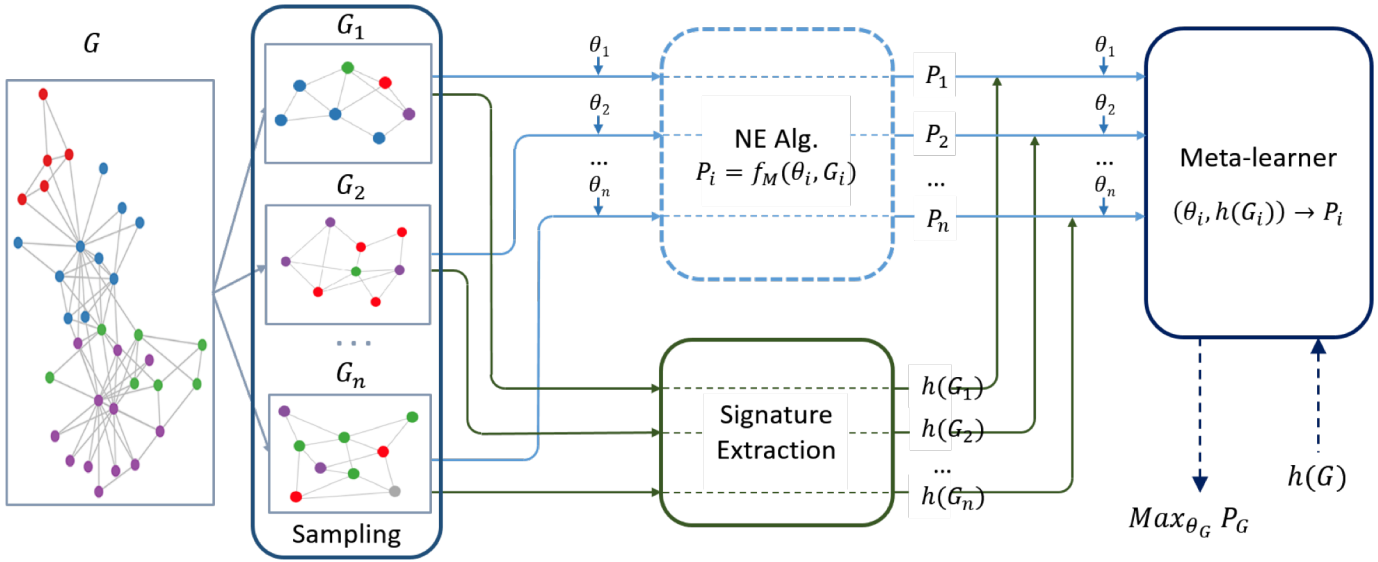
Sampled sub-network
Optimal configuration θ

Origin massive network

- Transferability
  - $\theta \neq$ optimal configuration on origin network
- Heterogeneity
  - several highly heterogeneous components => carefully designed sampling

Ke Tu, Jianxin Ma, **Peng Cui**, Jian Pei, Wenwu Zhu. AutoNRL: Hyperparameter Optimization for Massive Network Representation Learning. *KDD*, 2019.

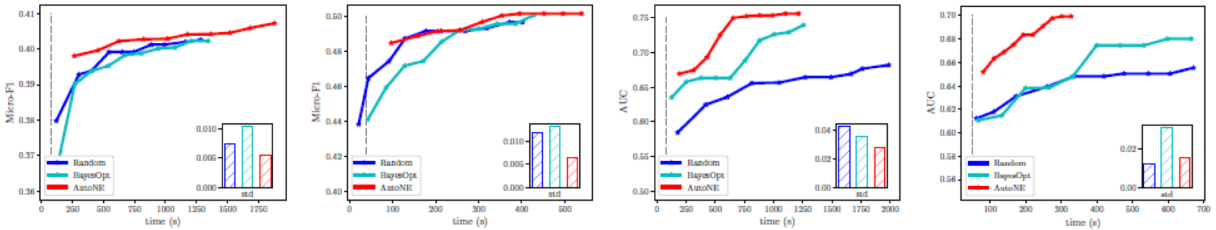
AutoNE



Transfer the knowledge about optimal hyperparameters from the sub-networks to the original massive network

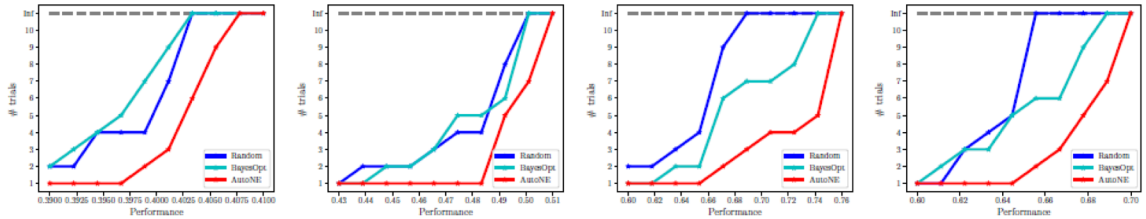
Ke Tu, Jianxin Ma, **Peng Cui**, Jian Pei, Wenwu Zhu. AutoNRL: Hyperparameter Optimization for Massive Network Representation Learning. *KDD*, 2019.

Experiment --- Sampling-Based NE



(a) Classification on BlogCatalog (b) Classification on Wikipedia (c) Link prediction on BlogCatalog (d) Link prediction on Wikipedia

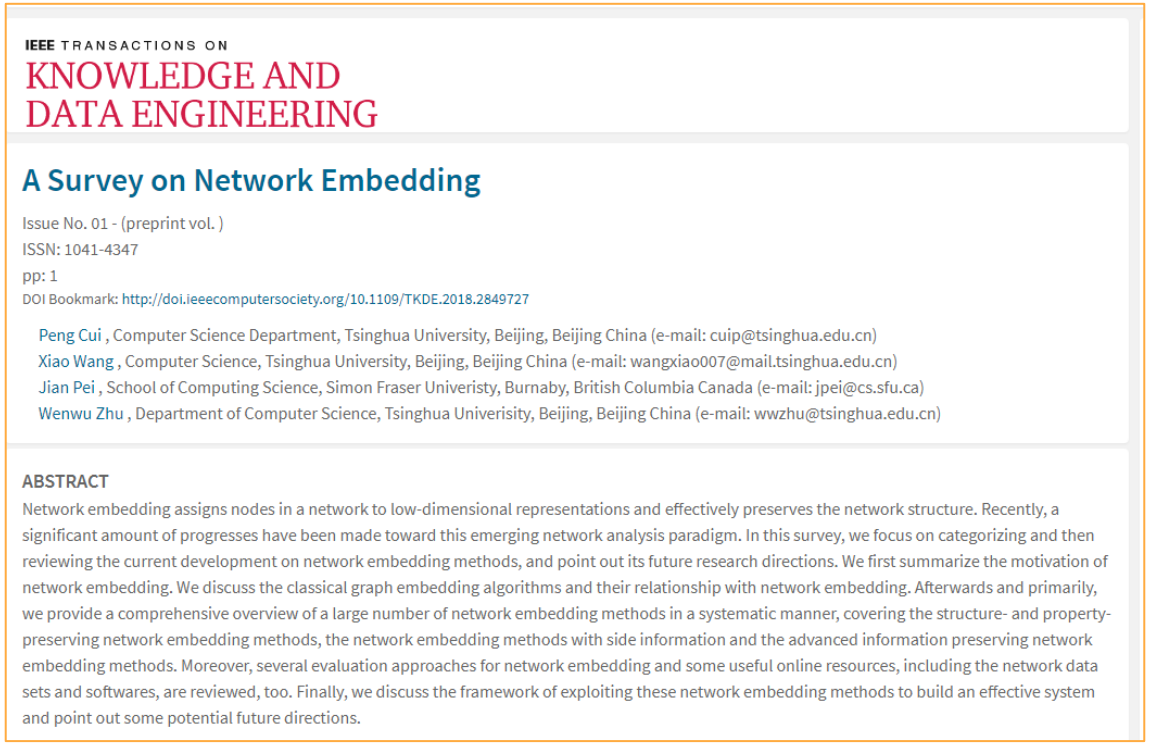
The performance achieved within various time thresholds.



(a) Classification on BlogCatalog (b) Classification on Wikipedia (c) Link prediction on BlogCatalog (d) Link prediction on Wikipedia

The number of trials to reach a certain performance threshold

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*.

Deep Learning on Graphs: A Survey

Deep Learning on Graphs: A Survey

Ziwei Zhang, Peng Cui and Wenwu Zhu

Abstract—Deep learning has been shown successful in a number of domains, ranging from acoustics, images to natural language processing. However, applying deep learning to the ubiquitous graph data is non-trivial because of the unique characteristics of graphs. Recently, a significant amount of research efforts have been devoted to this area, greatly advancing graph analyzing techniques. In this survey, we comprehensively review different kinds of deep learning methods applied to graphs. We divide existing methods into three main categories: semi-supervised methods including Graph Neural Networks and Graph Convolutional Networks, unsupervised methods including Graph Autoencoders, and recent advancements including Graph Recurrent Neural Networks and Graph Reinforcement Learning. We then provide a comprehensive overview of these methods in a systematic manner following their history of developments. We also analyze the differences of these methods and how to composite different architectures. Finally, we briefly outline their applications and discuss potential future directions.

Index Terms—Graph Data, Deep Learning, Graph Neural Network, Graph Convolutional Network, Graph Autoencoder.

1 Introduction

In the last decade, deep learning has been a "crown jewel" in artificial intelligence and machine learning [1], showing superior performance in acoustics [2], images [3] and natural language processing [4]. The expressive power of deep learning to extract complex patterns underlying data has been well recognized. On the other hand, graphs[1] are ubiquitous in the real world, repre-

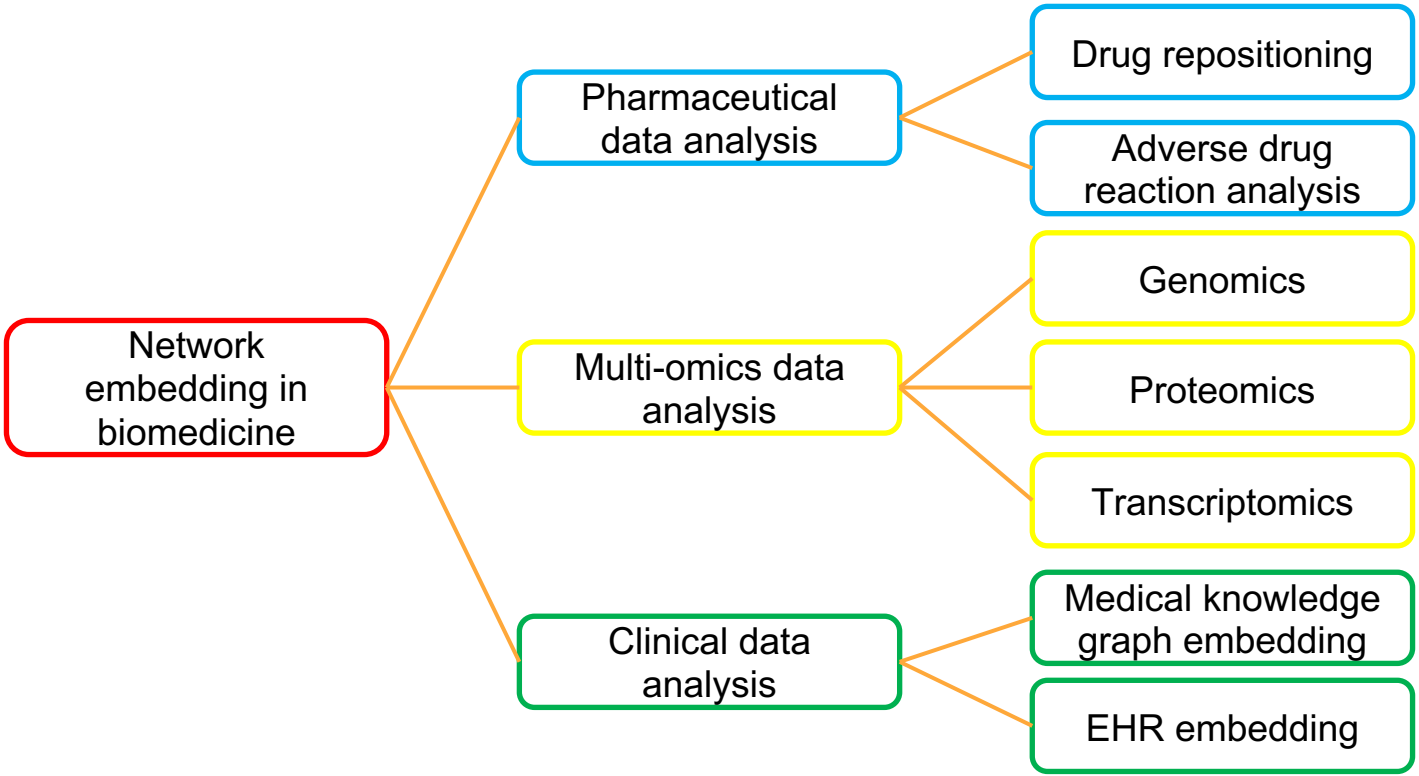
- **Scalability and parallelization**. In the big-data era, real graphs can easily have millions of nodes and edges, such as social networks or e-commerce networks [8]. As a result, how to design scalable models, preferably with a linear time complexity, becomes a key problem. In addition, since nodes and edges in the graph are interconnected and often need to be modeled as a whole, how to conduct parallel computing is another critical issue.

Ziwei Zhang, Peng Cui, Wenwu Zhu. Deep Learning on Graphs: A Survey. Arxiv, *2018*.

Outline

- Structure-preserved network embedding
- Property-preserved network embedding
- Dynamic network embedding
- Robustness, Explainability and Applicability
- Network embedding for biomedical applications

Network Embedding for Biomedical Applications



1. Chang Su, Jie Tong, Yongjun Zhu, Peng Cui, Fei Wang, Network embedding in biomedical data science, Briefings in Bioinformatics, , bby117, https://doi.org/10.1093/bib/bby117

Pharmaceutical data analysis

- **Drug repositioning**

  Exploring new usage for existing drugs.

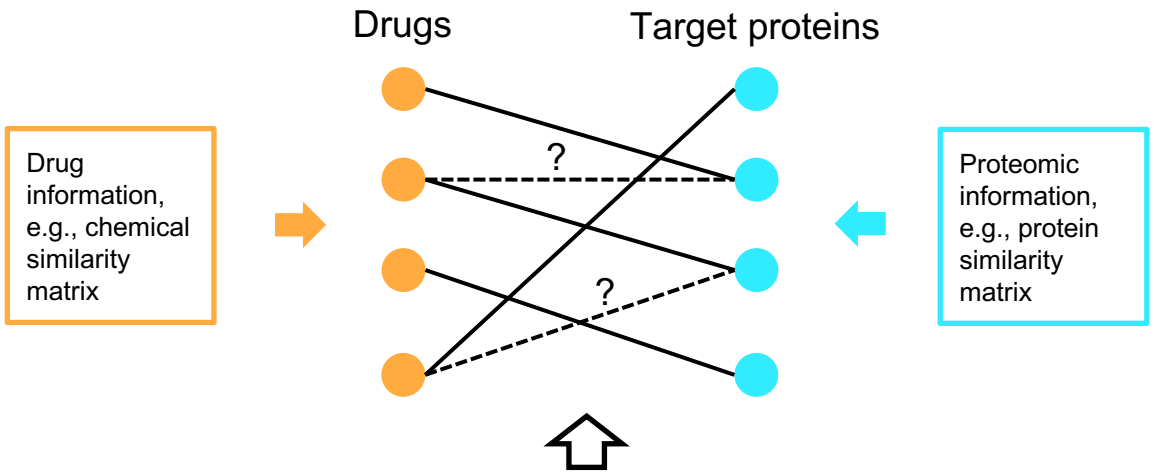
  - save drug development cost
  - increase productivity

  Aiming at predicting:

  - unknown drug-target interactions
  - unknown drug-disease interactions

Pharmaceutical data analysis

- Drug-target interaction prediction

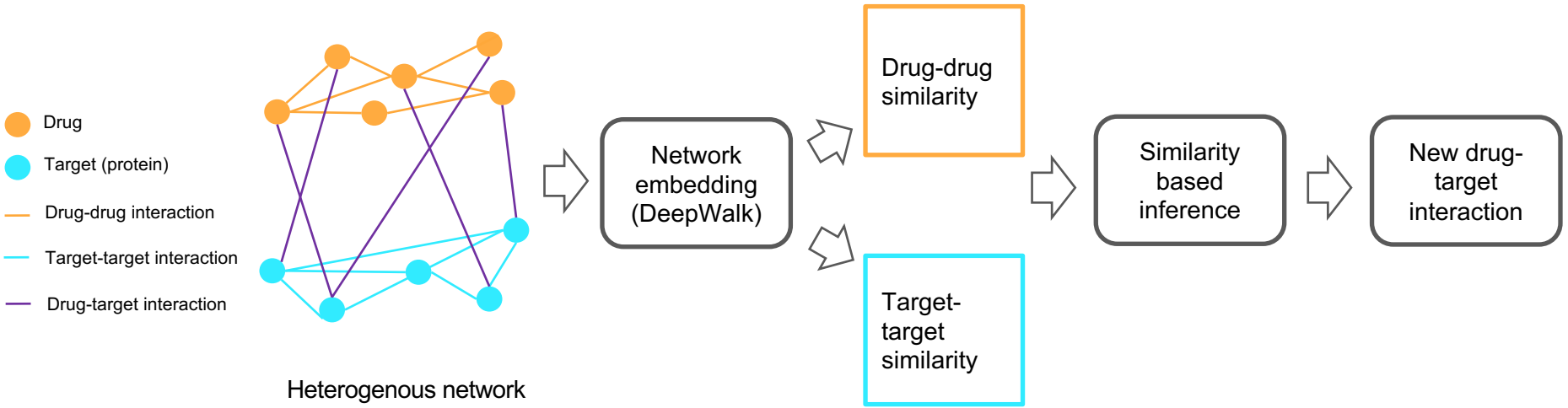


Matrix factorization based embedding methods, e.g., LE

1. Yamanishi Y, Araki M, Gutteridge A, et al. Prediction of drug–target interaction networks from the integration of chemical and genomic spaces. Bioinformatics 2008;24: i232–40.
2. Cobanoglu MC, Liu C, Hu F, et al. Predicting drug–target interactions using probabilistic matrix factorization. J Chem Inf Model 2013;53:3399–409.
3. Zheng X, Ding H, Mamitsuka H, Zhu S. Collaborative matrix factorization with multiple similarities for pre- dicting drug–target interactions. KDD ’13, 2013, pp. 1025–1033. Chicago, Illinois, USA.

Pharmaceutical data analysis

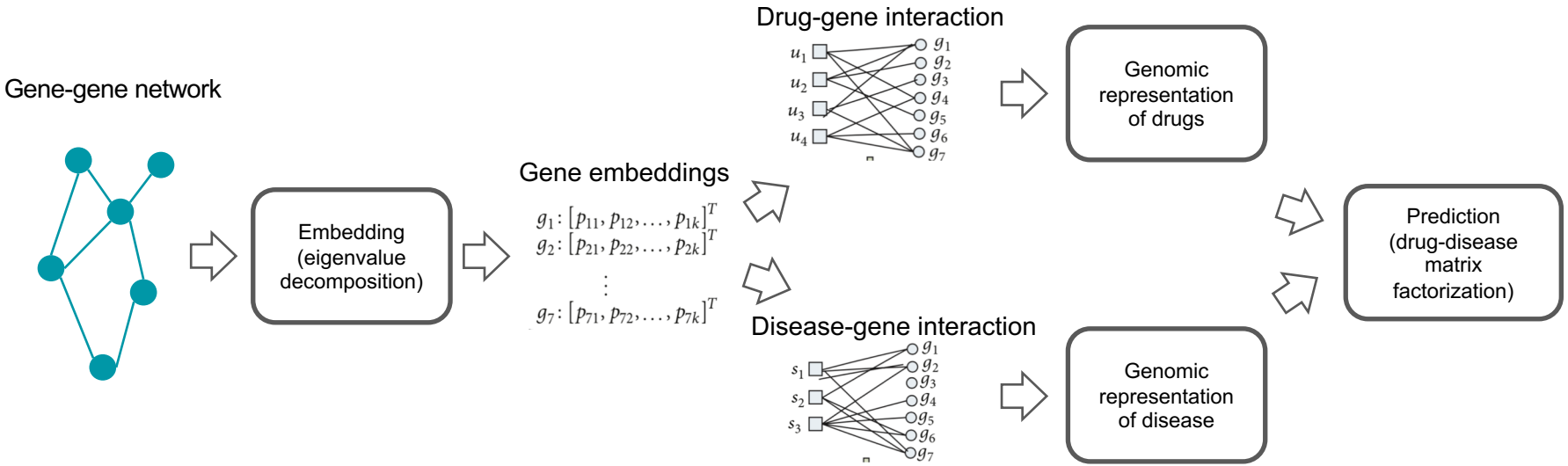
- Drug-target interaction prediction (Zong et al. 2017)



1. Zong N, Kim H, Ngo V, et al. Deep mining heterogeneous networks of biomedical linked data to predict novel drug– target associations. Bioinformatics 2017;33:2337–44.

Pharmaceutical data analysis

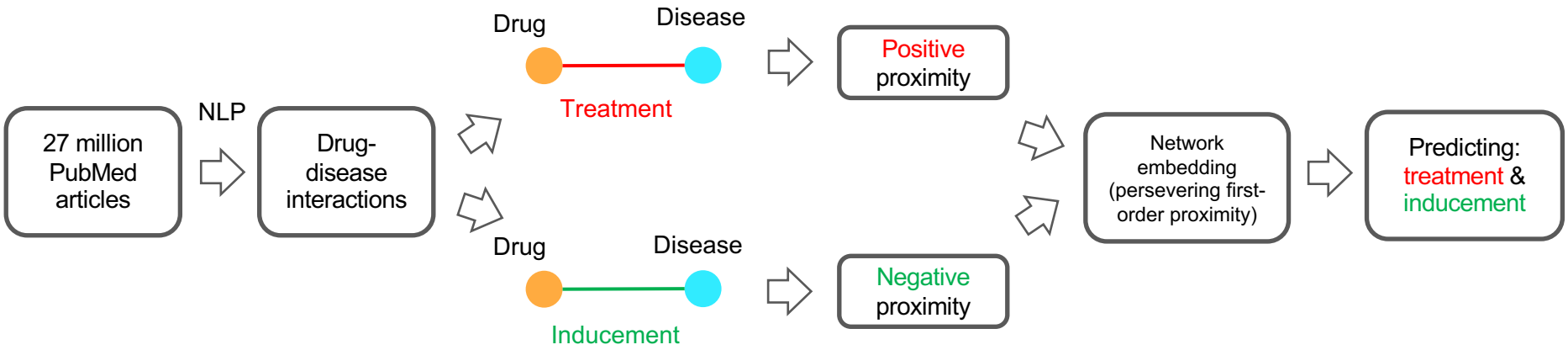
- Drug-disease interaction prediction (Dai et al. 2015)



1. Dai W, Liu X, Gao Y, et al. Matrix factorization-based predic- tion of novel drug indications by integrating genomic space. Comput Math Methods Med 2015;2015:275045.

Pharmaceutical data analysis

- Drug-disease interaction prediction (Wang et al. 2017)



1. Wang P, Hao T, Yan J, et al. Large-scale extraction of drug– disease pairs from the medical literature. J Assoc Inf Sci Technol 2017;68:2649–61.

Pharmaceutical data analysis

- Adverse drug reaction analysis

  An adverse drug reaction (ADR) is defined as any undesirable effect from the medical use of drugs beyond its anticipated therapeutic effects that occurs at a usual dosage.

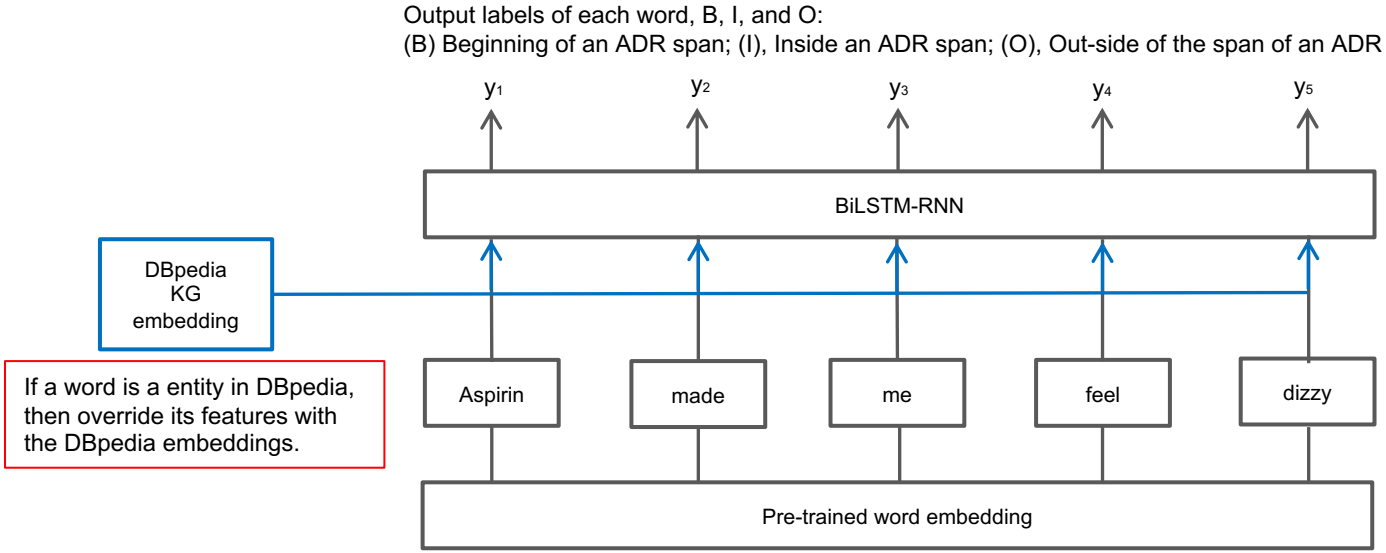
  The ADR study is implemented before a drug is launched on clinical application.

  - Adverse drug reaction (ADR) prediction
  - Drug-drug interaction (DDI) prediction

Pharmaceutical data analysis

- Adverse drug reaction (ADR) prediction (Stanovsky et al. 2017)

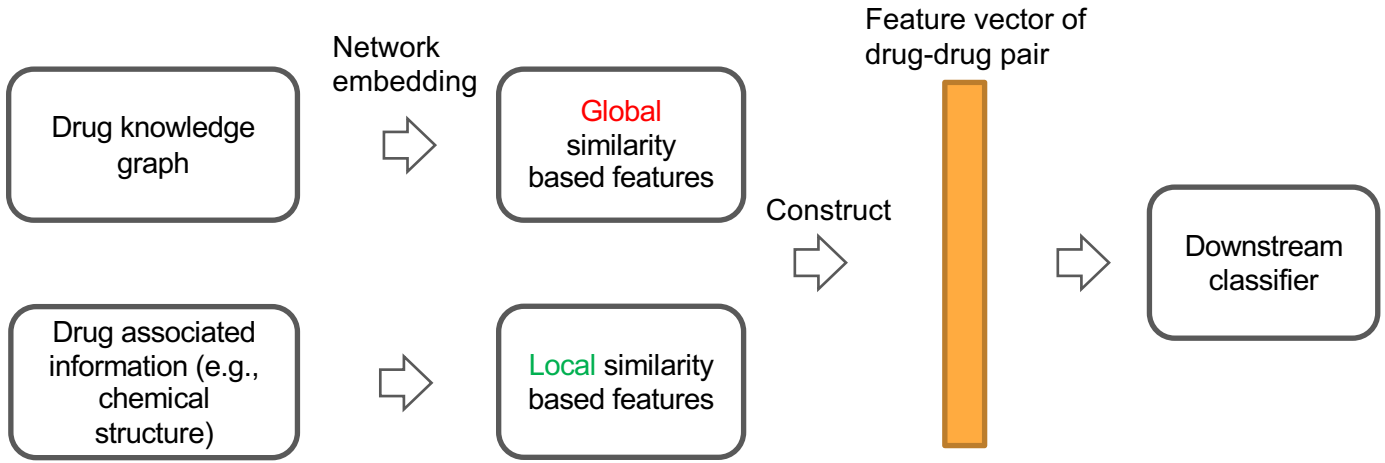
  Identify ADR from social media posts:



1. Stanovsky G, Gruhl D, Mendes P. Recognizing mentions of adverse drug reaction in social media using knowledge- infused recurrent models. In: Proceedings of the 15th Confer- ence of the European Chapter of the Association for Computa- tional Linguistics, 2017, pp. 142–51. Valencia, Spain.

Pharmaceutical data analysis

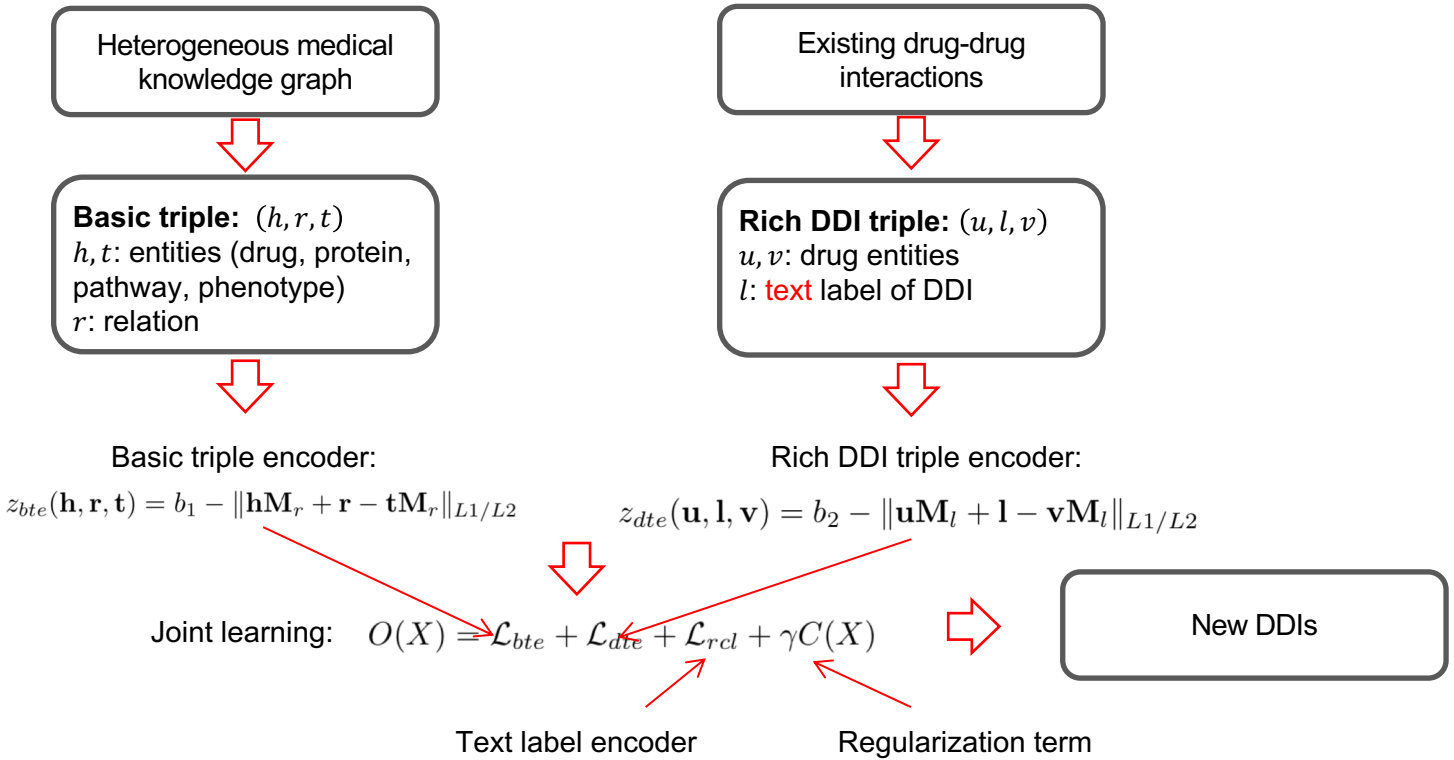
- Drug-drug interaction (DDI) prediction (Abdelaziz et al. 2017)



1. Abdelaziz I, Fokoue A, Hassanzadeh O, et al. Large-scale structural and textual similarity-based mining of knowl- edge graph to predict drug–drug interactions. Web Semant 2017;44:104–17.

Pharmaceutical data analysis

- Drug-drug interaction (DDI) prediction (Wang et al. 2017)

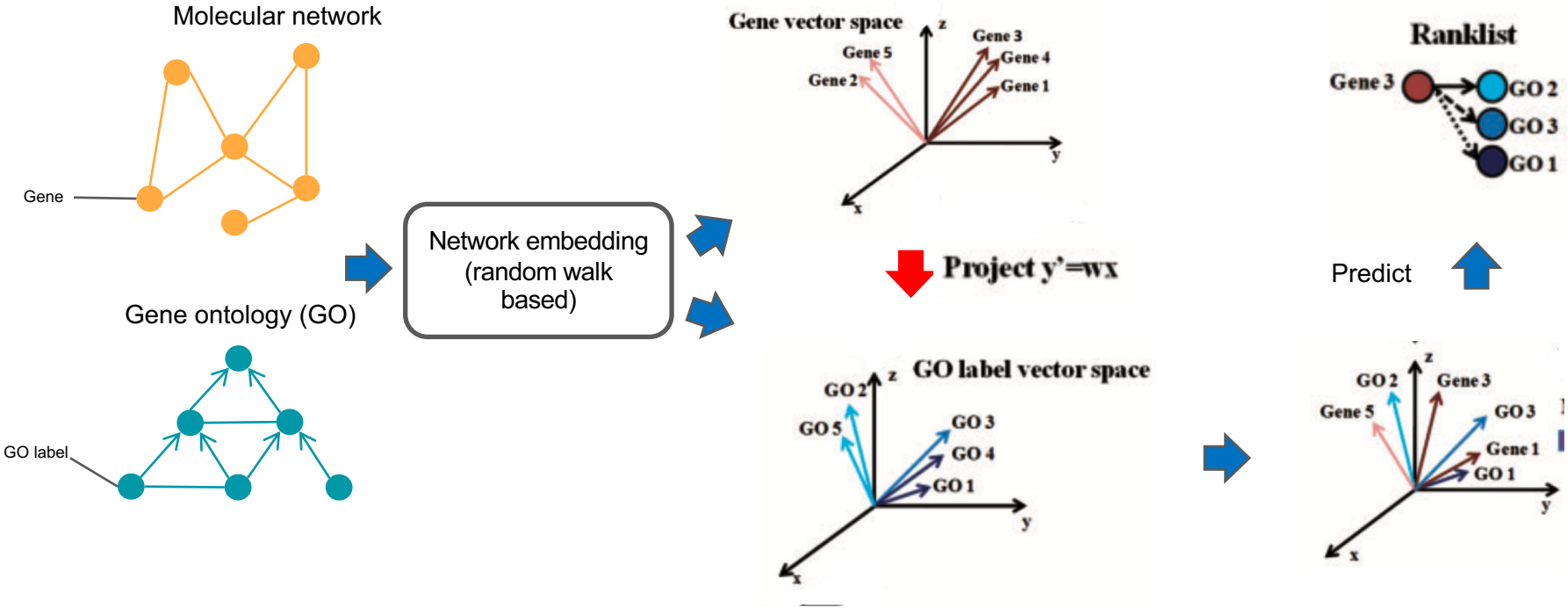


1. Wang M, Chen Y, Qian B, et al. Predicting rich drug–drug interactions via biomedical knowledge graphs and text jointly embedding, 2017, arXiv preprint arXiv:171208875.

Multi-omics data analysis

❑ Genomics data analysis

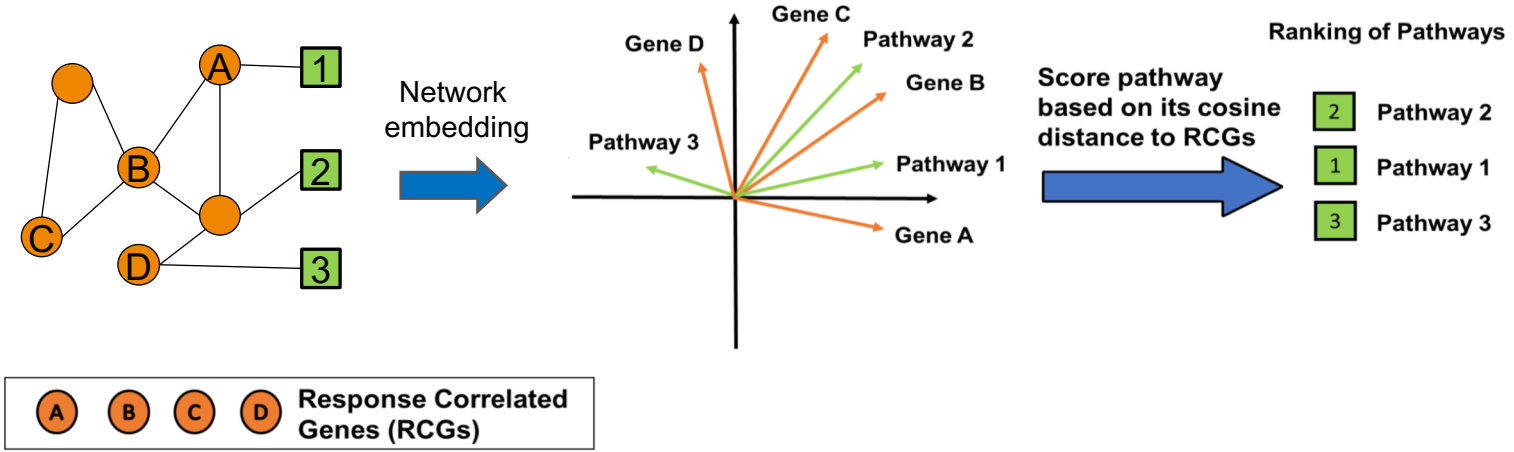
○ Gene function prediction (Wang et al. 2015)



1. Wang S, Cho H, Zhai C, et al. Exploiting ontology graph for predicting sparsely annotated gene function. Bioinformatics 2015;31:i357–64

Multi-omics data analysis

- Identification of Pathways Associated with Chemosensitivity (Wang et al. 2017)



1. Wang S, Huang E, Cairns J, et al. Identification of pathways associated with chemosensitivity through net- work embedding. 2017, bioRxiv preprint bioRxiv: 168450 doi:10.1101/168450

Multi-omics data analysis

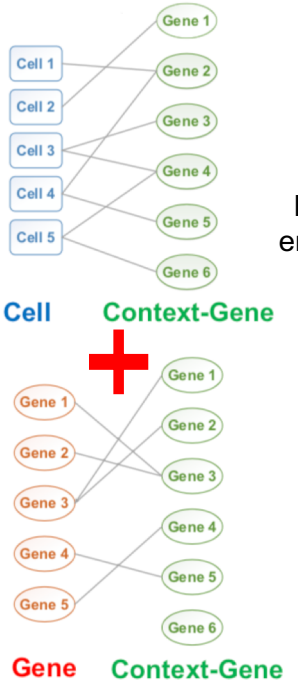
- Cell and gene representation (Li et al. 2017)

scRNA-seq data

	Gene1	Gene2	Gene3	Gene4	Gene5	Gene6
Cell1	0	5	0	0	0	0
Cell2	3	0	0	0	0	0
Cell3	0	0	100	50	0	0
Cell4	0	56	0	0	10	0
Cell5	0	0	0	99	0	10

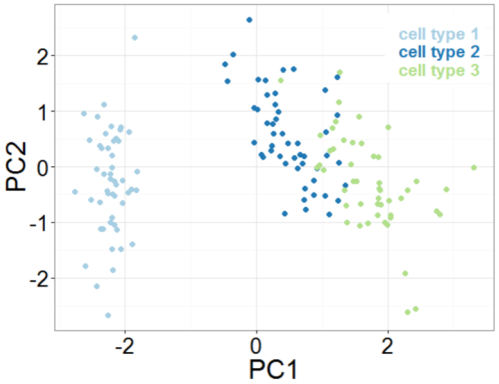
Pathway data

	Gene1	Gene2	Gene3	Gene4	Gene5
Gene1	0	0	1	0	0
Gene2	0	0	1	0	0
Gene3	1	1	0	0	0
Gene4	0	0	0	0	1
Gene5	0	0	0	1	0



Network embedding

Capturing heterogeneity of cells



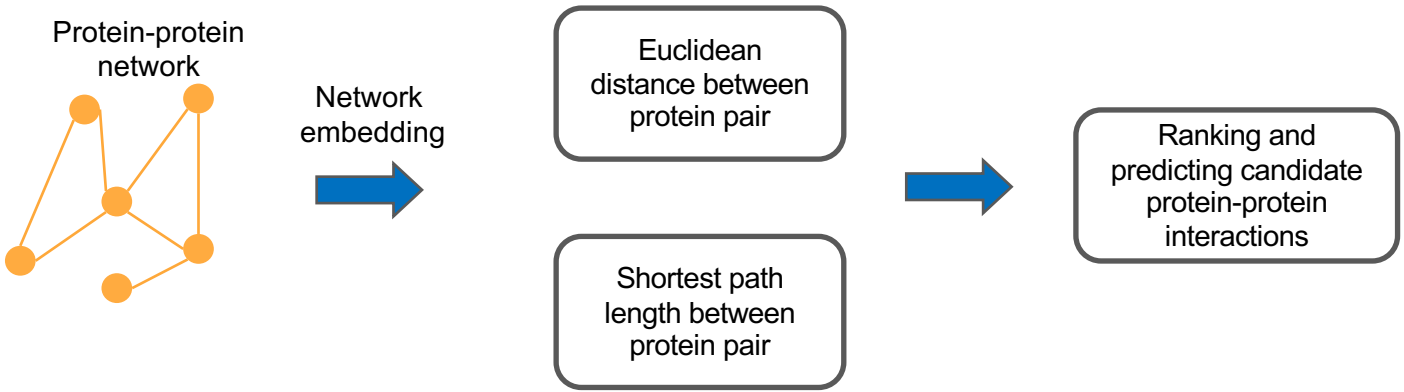
1. Wang S, Huang E, Cairns J, et al. Identification of pathways associated with chemosensitivity through net- work embedding. 2017, bioRxiv preprint bioRxiv: 168450 doi:10.1101/168450

Multi-omics data analysis

❑ Proteomics data analysis

- Protein-protein interaction (PPI) prediction (Cannistraci et al. 2013)

Experimental identification of PPIs is time-consuming.
Computational method was proposed to predict candidate PPIs based on existing interactions.

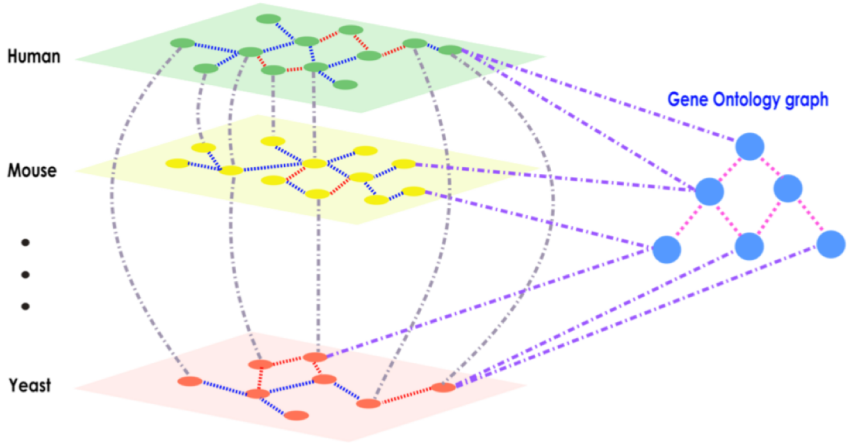


1. Cannistraci CV, Alanis-Lobato G, Ravasi T. Minimum curvi- linearity to enhance topological prediction of protein interactions by network embedding. Bioinformatics 2013;29: i199–209.

Multi-omics data analysis

- Protein function prediction (Wang et al. 2017)

Heterogenous protein-protein network



1. Construct protein-protein network;
2. Learn low-dimensional representations for proteins;
3. Calculate an intra-species affinity score and an inter-species affinity score by transferring Gene annotations
4. Rank the score and pick the function(s) with the highest score(s) for queried protein (s)

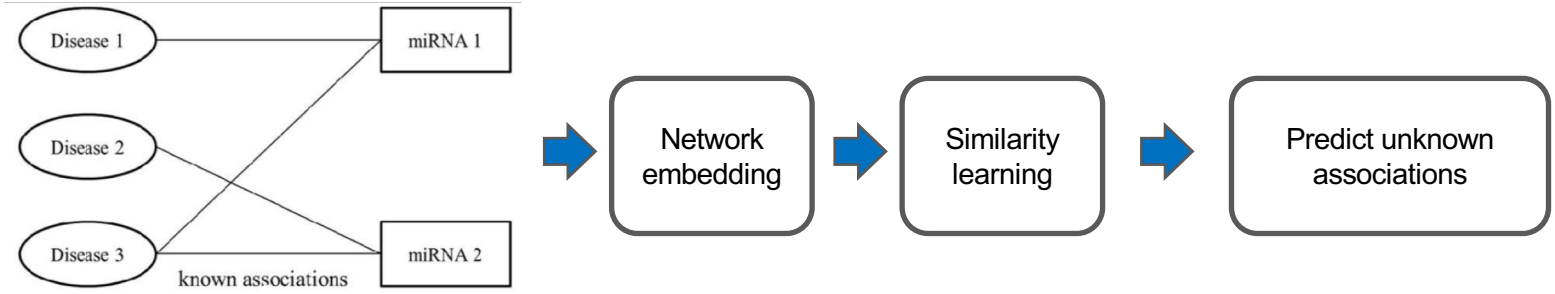
1. Wang S, Qu M, Peng J. Prosnet: integrating homology with molecular networks for protein function prediction. Pac Symp Biocomput 2017;22:27–38.

Multi-omics data analysis

❑ Transcriptomics data analysis

- MicroRNA-disease association prediction (Li et al. 2017)

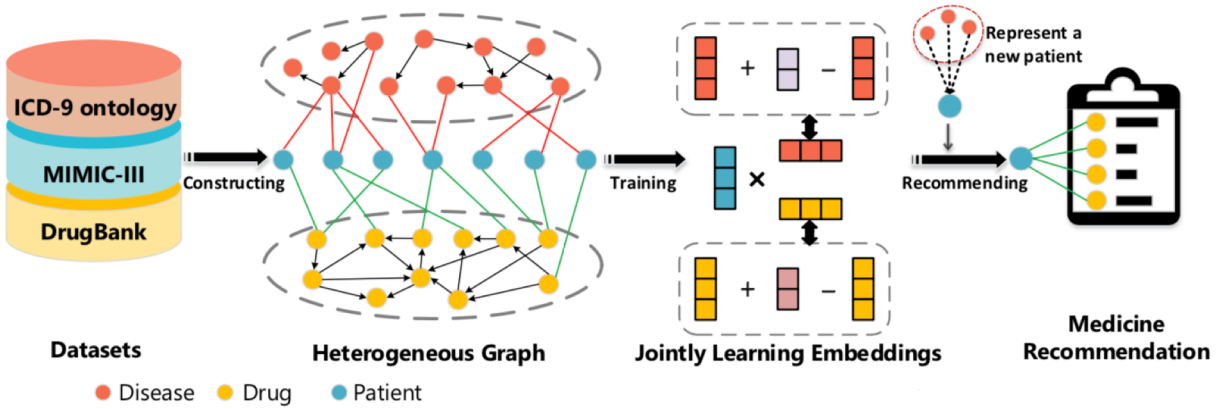
  - microRNAs (miRNAs) are connected with several complex human diseases;
  - Identifying human disease-related miRNAs will be useful in uncovering novel prognostic markers for cancer.



1. Li G, Luo J, Xiao Q, et al. Predicting microRNA-disease associations using network topological similarity based on DeepWalk. IEEE Access 2017;5:24032–9.

Medical data analysis

- **Medical knowledge graph embedding**
  - Safe medication recommendation (Wang et al. 2017) – predicting patient-medicine association



$$\mathcal{L}(X) = \mathcal{L}_{G_m} + \mathcal{L}_{G_d} + \mathcal{L}_{G_{pm}} + \mathcal{L}_{G_{pd}} + \gamma C(X)$$

Regularization term

Medicine network embedding

Disease network embedding

Patient-medicine network embedding

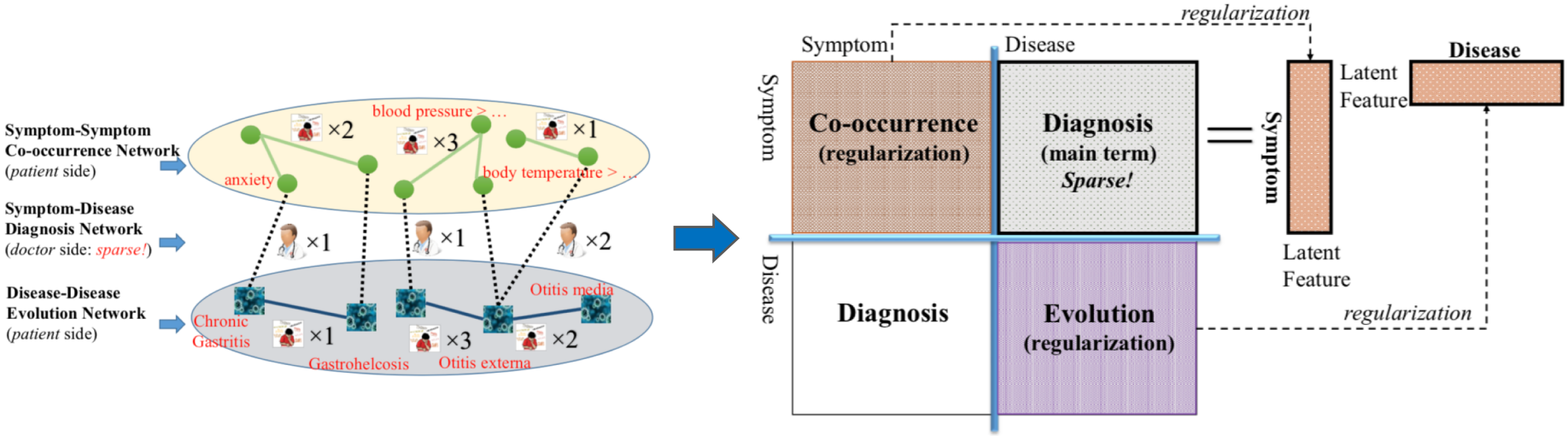
Patient-disease network embedding

1. Wang M, Liu M, Liu J, et al. Safe medicine recommenda- tion via medical knowledge graph embedding, 2017, arXiv preprint arXiv:171005980

Medical data analysis

- Representation learning on medical forum data (Zhao et al. 2017)

  Learning low-dimensional representations of diseases and symptoms.



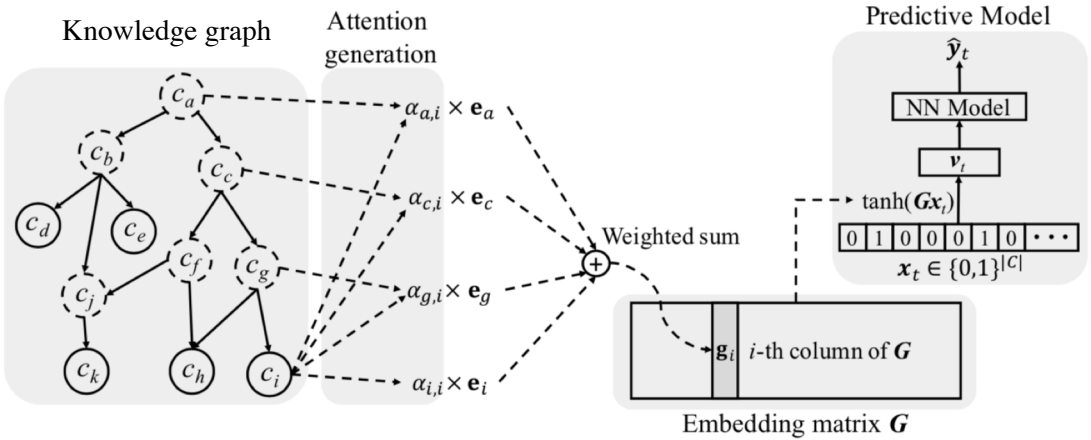
1. Zhao S, Jiang M, Yuan Q, et al. ContextCare: incorporating contextual information networks to representation learn- ing on medical forum data. In: Proceedings of the 26th Inter- national Joint Conference on Artificial Intelligence, 2017, pp. 3497–503. AAAI Press, Melbourne, Australia.

Medical data analysis

❑ EHR embedding

○ Healthcare representation learning (Choi et al. 2017)

- EHR representation with hierarchical information inherent to medical ontologies;
- The model represents a medical concept as a combination of its ancestors in the ontology via an attention mechanism.



Attention is assigned to the node (medical concept) embedding:

$$\mathbf{g}_i = \sum_{j \in \mathcal{A}(i)} \alpha_{ij} \mathbf{e}_j, \qquad \sum_{j \in \mathcal{A}(i)} \alpha_{ij} = 1, \ \alpha_{ij} \geq 0$$

$$\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t = \tanh(\mathbf{G}[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t]),$$
$$\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_t = \mathrm{RNN}(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_t, \theta_r),$$
$$\widehat{\mathbf{y}}_t = \widehat{\mathbf{x}}_{t+1} = \mathrm{Softmax}(\mathbf{W}\mathbf{h}_t + \mathbf{b}),$$

Node: medical concept

1. Choi E, Bahadori MT, Song L, et al. GRAM: graph-based attention model for healthcare representation learning. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 787–95. ACM, Halifax, Nova Scotia, Canada.

Network Embedding for Biomedical Applications

❑ Challenges

- Data quality. Networks constructed from the biomed- ical data are usually noisy and incomplete. For example, the PPI data produced by high-throughput techniques, suffer from high false negative rates up to 70% and high false positive rates up to 64%.
- Local and global. Network embedding and its downstream tasks rely on the type of structural property to preserve. Designing embedding method by properly considering local and global structure properties require the development of novel solutions.
- Network evolution. Networks are always not static, especially in the biomedical domain. Yet most existing embedding models focus on static network.
- Domain complexity. Network structure is highly associated with domain knowledge.

Network Embedding for Biomedical Applications

- **Opportunities**
  - Local and global trade-off embedding.
  - Dynamic network embedding.
  - Text associated embedding. Medical knowledge bases always contain rich text information such as descriptions of entities and relations, which would have high potential to address network incompleteness and improve under- standing of topological properties.
  - Domain-knowledge-associated embedding. Incorporating external domain knowledge into network embedding.

Summary and Conclusion

- **Structure-preserved network embedding**
  - Guarantee information equivalence
- **Property-preserved network embedding**
  - Enabling network inference in embedding space
- **Dynamic network embedding**
  - Incorporating dynamic changes
- **Robustness, Explainability and Applicability**
  - Promote network embedding in real applications
- **Network embedding for biomedical applications**
  - Prove the effectiveness of NE in real applications