Machine Learning with Graphs (MLG)

Graph Representation Learning

Learning node embeddings in a graph

Cheng-Te Li (李政德)

Institute of Data Science

National Cheng Kung University

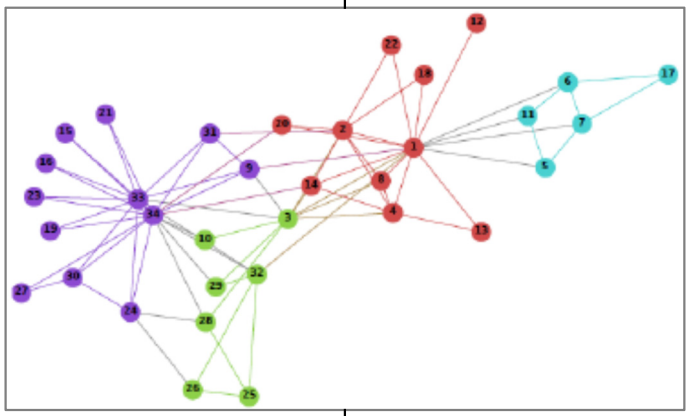
chengte@mail.ncku.edu.tw

Graph Representation:

from **Hand-crafted** Features to **Learned** Features

x_{ij}: node v_i's j^{th} feature,
e.g., v_i's degree, centrality, and pagerank values

Feature Engineering



v_i

$\boldsymbol{X}$

Hand-craft Feature Matrix

Machine Learning Models

Network Analysis Tasks
- ✓ Node Classification
- ✓ Link Prediction
- ✓ Community Detection
- ✓ User Profiling
- ✓ etc.

x_{ij}: node v_i's j^{th} learned feature

v_i

$\boldsymbol{Z}$

Learned Latent Feature Matrix

Feature Learning

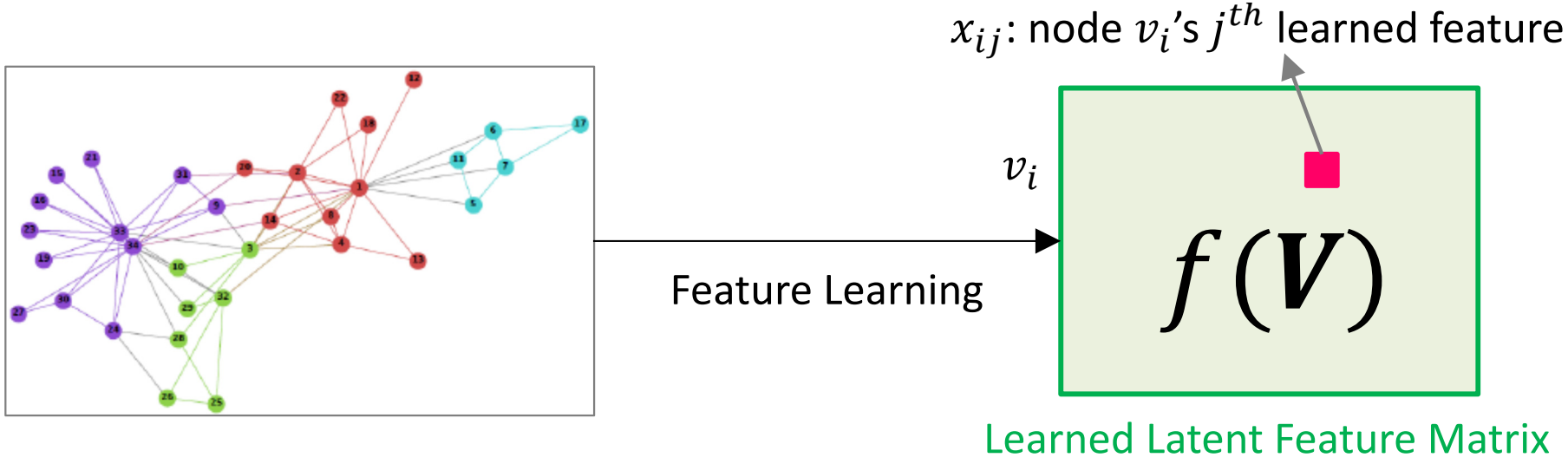
Machine Learning Models

Roadmap on GRL

- Introduction to NRL
- From Word Embedding to Network Embedding
- NRL Models
  - Factorization-based
    - Graph Factorization, GraRep, HOPE
  - Random Walk-based
    - DeepWalk, node2vec, LINE

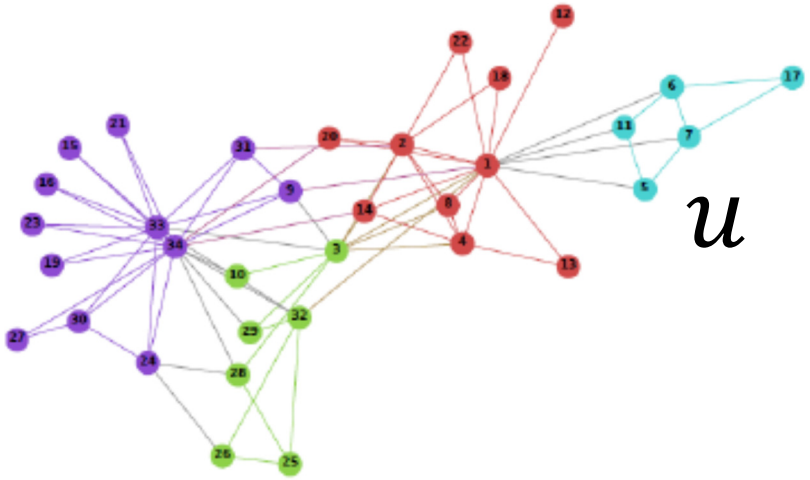
Graph Representation Learning

- Problem: Graph Representation Learning,
  (a.k.a. Network Embedding Learning)
  - Input: a network $G = (V, E)$
  - Output: $\boldsymbol{Z} \in \mathbb{R}^{|V| \times d}, d \ll |V|$, d-dim vector $\boldsymbol{Z}_v$ for each node v
- The goal is to map each node into a latent low-dimension space such that network structure information is encoded into distributed node representations



Graph Representation Learning

[Perozzi et al.'14, Tang et al.'15, Grover & Leskovec'16]



u

Network of Zachary's Karate Club

$f: u \rightarrow \mathbb{R}^d$

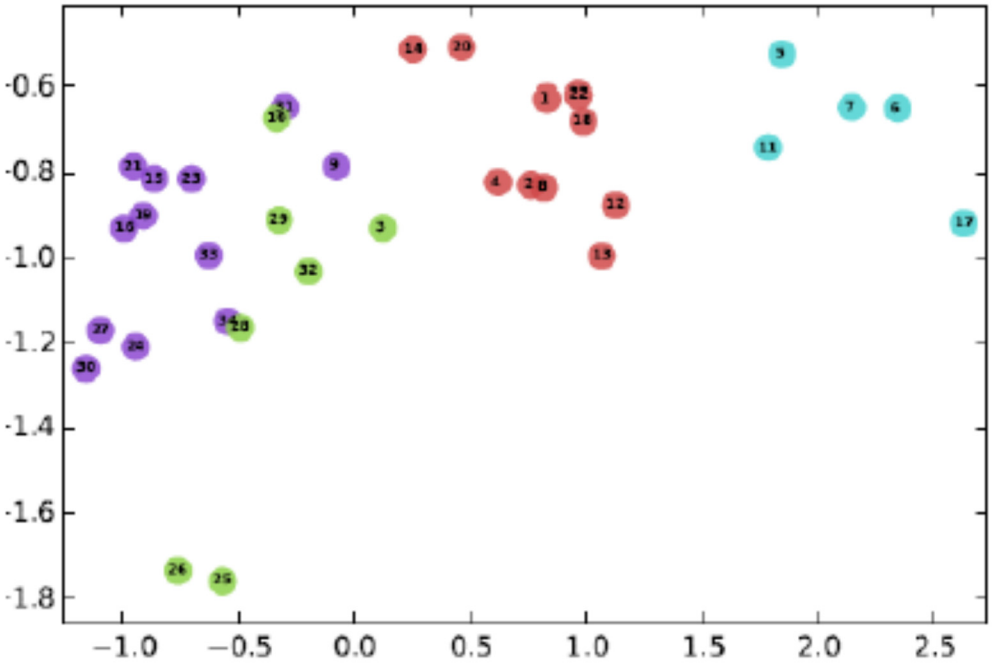
Embedding Learning

$\mathbb{R}^d$

Embedding Vector

- **Embedding learning**: map each node into a low-dimensional space
- **Preserving proximity**: similarity between node embeddings indicates proximity between nodes

e.g., $d = 2$

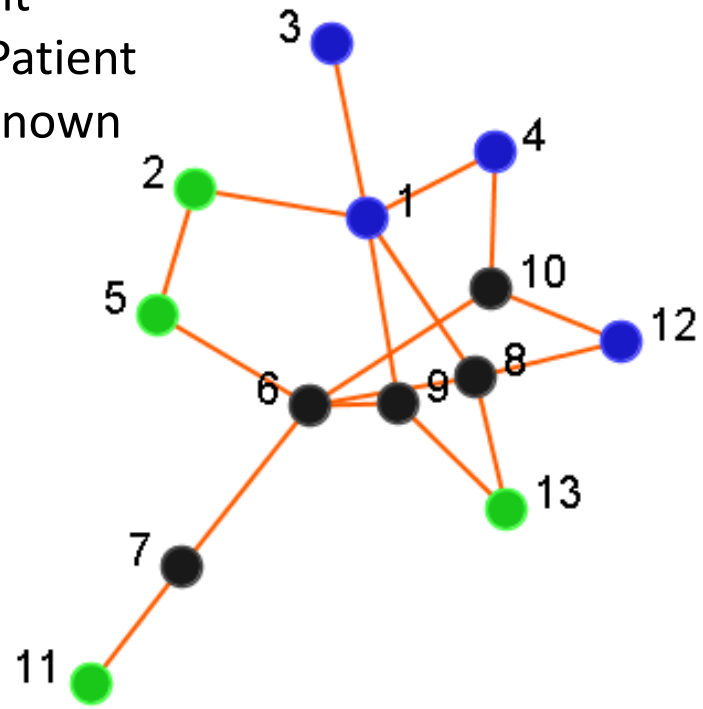


Typical Application 1: Node Classification

Potential Labels:
Disease Spread
Fake News Spread
Political opinion
User Profiles

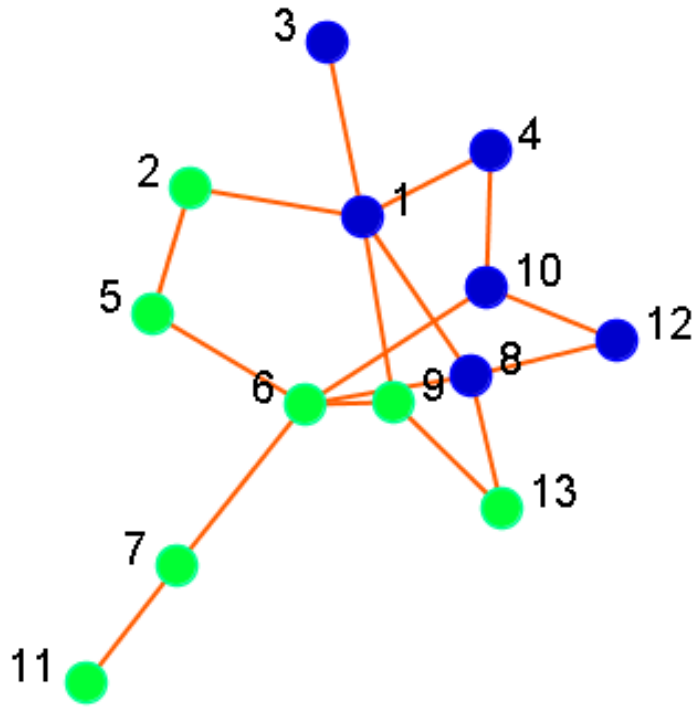
- Given labels of some nodes, predicting labels of the remaining nodes

●: Patient
●: Non-Patient
●: ? Unknown



predict

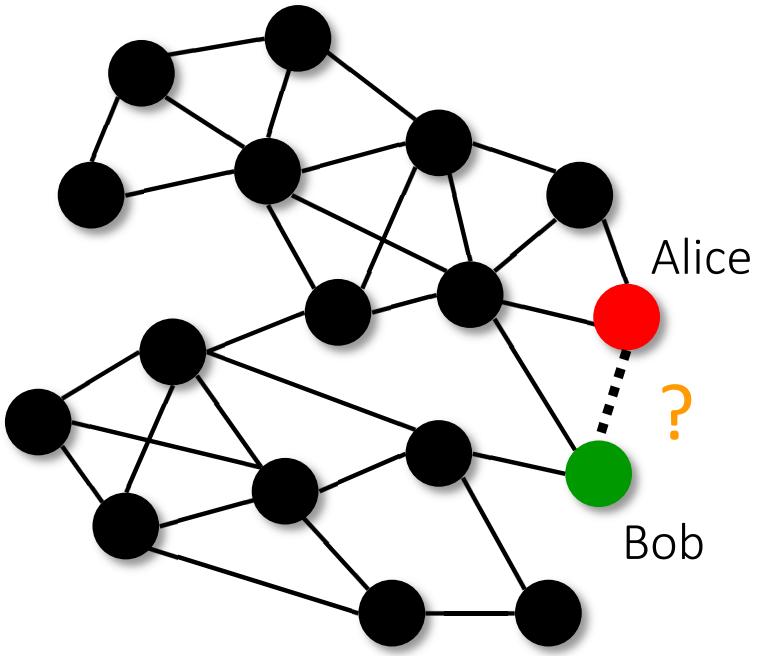
6: Non-Patient
7: Non-Patient
8: Patient
9: Non-Patient
10: Patient



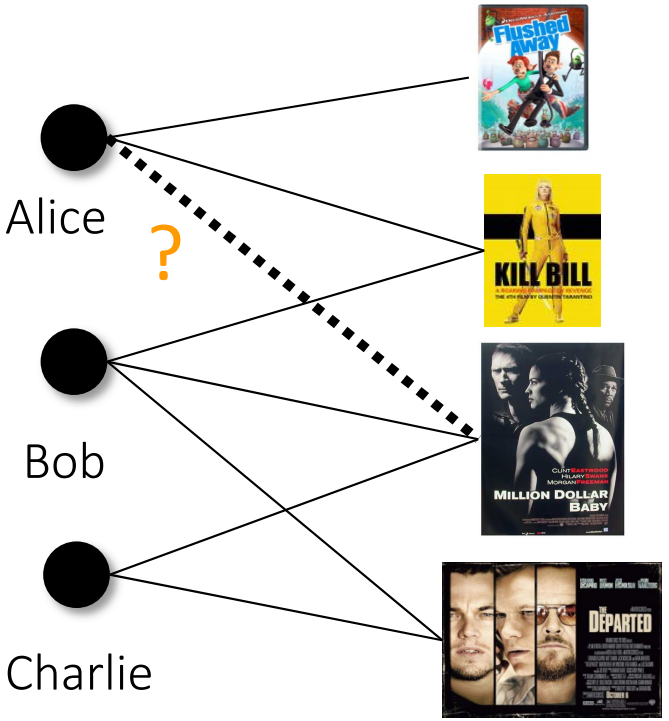
Typical Application 2: Link Prediction

- Given the current network structure, predicting which links will be created

Friend Recommendation



Item Recommendation



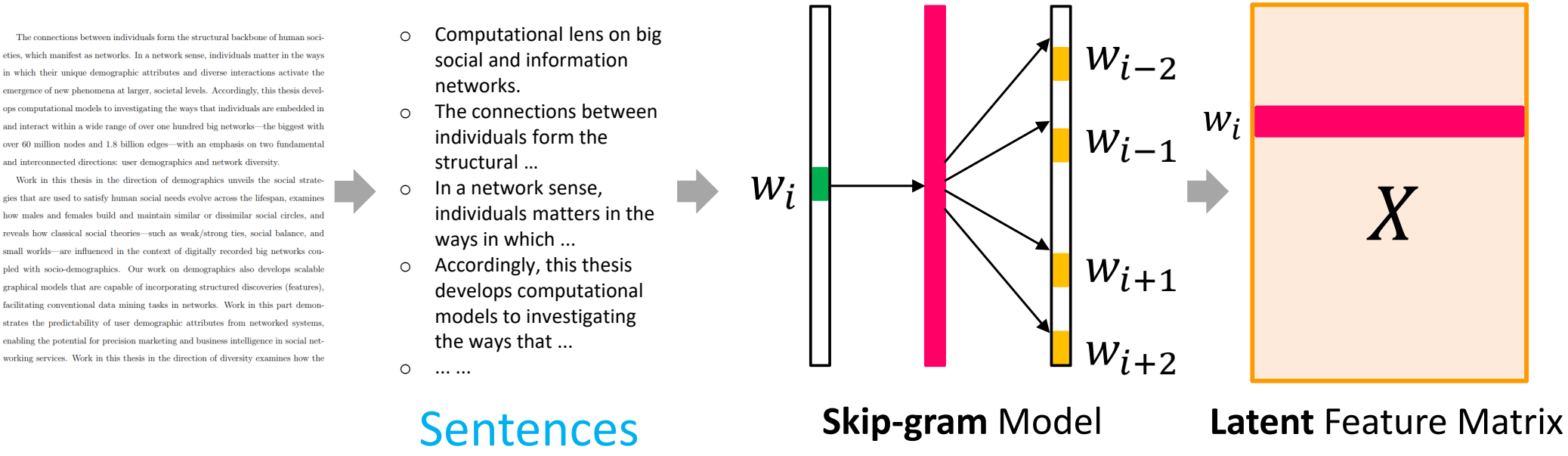
Literatures on GRL

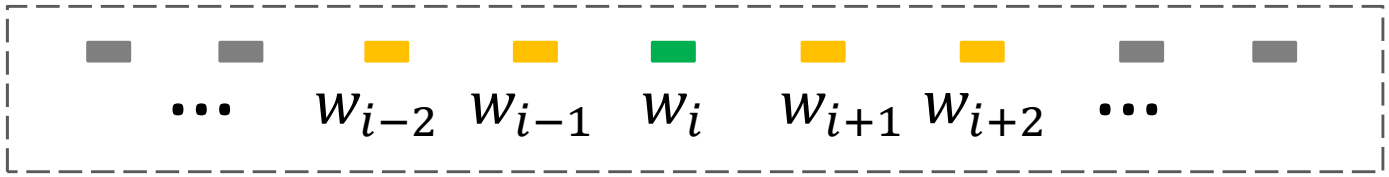
FastGCNs, Graph Attention Net	2018: Velickovic et al., ICLR'18, Chen et al., ICLR 2018
NetMF & NetSMF	2018: Qiu et al., WSDM'18 & WWW'19
Neural message passing, GraphSage	2017: Gilmer et al., ICML'17; Hamilton et al., NIPS'17
Gated graph neural network	2016: Li et al., ICLR'16
structure2vec	2016: Dai et al., ICML'16
Graph Convolutional Network	2015: Duvenaud et al., NIPS'15; Kipf & Welling ICLR'17
PTE, metapath2vec	2015: Tang et al., KDD'15; Dong et al., KDD'17
LINE, node2vec	2015: Tang et al., WWW'15; Grover & Leskovec, KDD'16
DeepWalk	2014: Perozzi et al., KDD'14
Spectral graph convolution	2014: Bruna et al., ICLR'14
word2vec (skip-gram)	2013: Mikolov et al., ICLR'13
Graph neural network	2005: Gori et al., IJCNN'05
Spectral clustering	2000: Ng et al. & Shi, Malik
Spectral partitioning	1973: Donath & Hoffman

Credit: Jie Tang

Word Embedding in NLP

- Input: a text corpus $D = \{W\}$
- Output: $\boldsymbol{X} \in \mathbb{R}^{|W| \times d}$, $d \ll |W|$, d-dim vector $\boldsymbol{X}_w$ for each word w





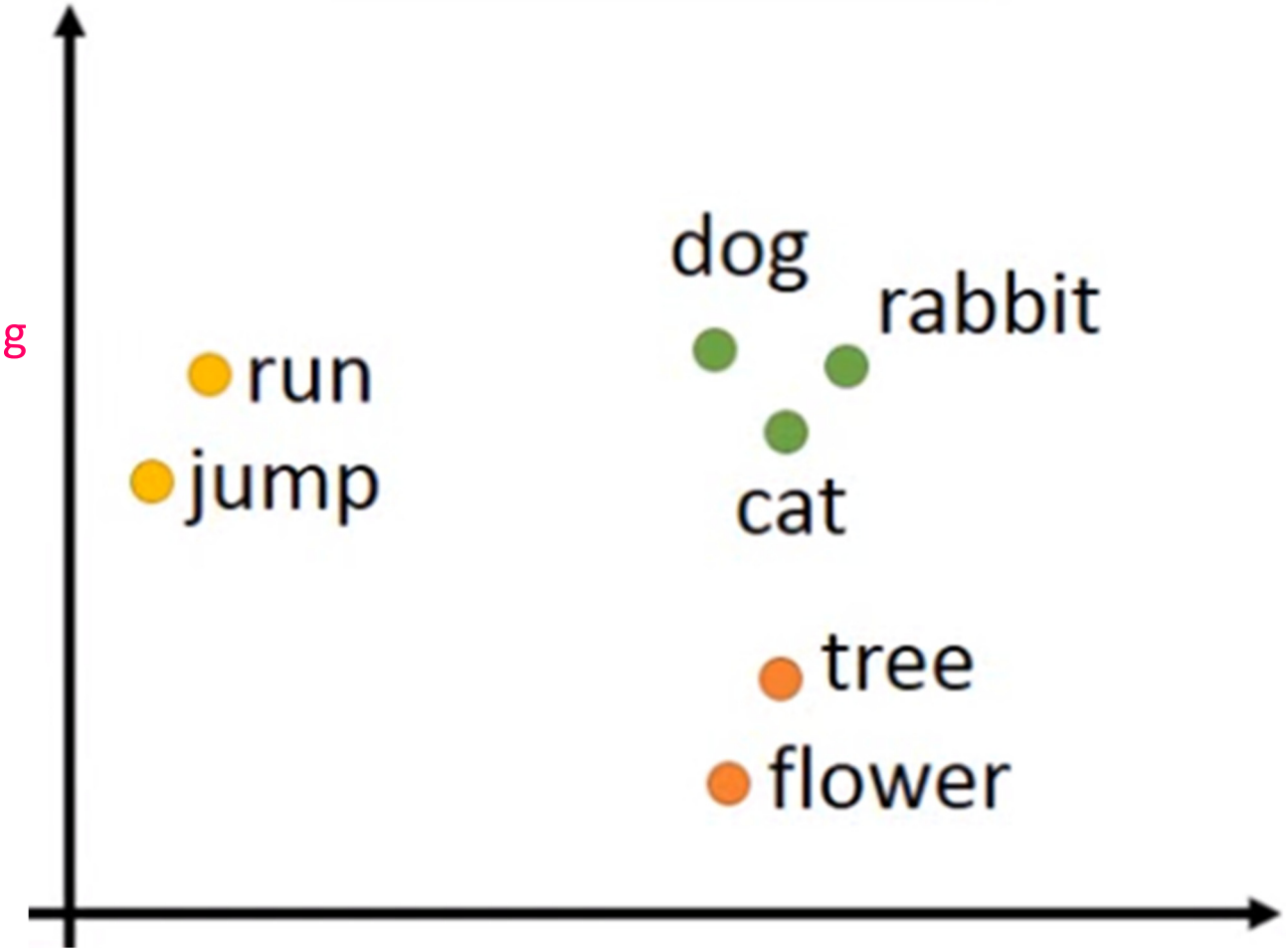
- Words in similar contexts have similar meanings
- Key idea: try to predict the words that surrounding each one

Illustration of Word Embedding

The connections between individuals form the structural backbone of human societies, which manifest as networks. In a network sense, individuals matter in the ways in which their unique demographic attributes and diverse interactions activate the emergence of new phenomena at larger, societal levels. Accordingly, this thesis develops computational models to investigating the ways that individuals are embedded in and interact within a wide range of over one hundred big networks—the biggest with over 60 million nodes and 1.8 billion edges—with an emphasis on two fundamental and interconnected directions: user demographics and network diversity.

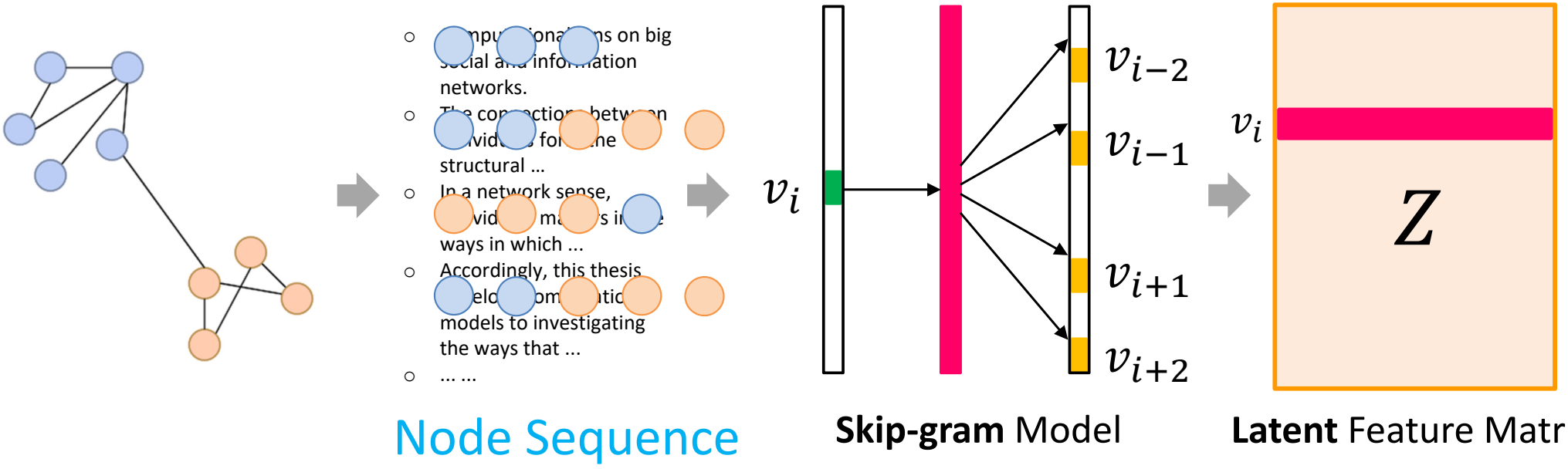
Work in this thesis in the direction of demographics unveils the social strategies that are used to satisfy human social needs evolve across the lifespan, examines how males and females build and maintain similar or dissimilar social circles, and reveals how classical social theories—such as weak/strong ties, social balance, and small worlds—are influenced in the context of digitally recorded big networks coupled with socio-demographics. Our work on demographics also develops scalable graphical models that are capable of incorporating structured discoveries (features), facilitating conventional data mining tasks in networks. Work in this part demonstrates the predictability of user demographic attributes from networked systems, enabling the potential for precision marketing and business intelligence in social networking services. Work in this thesis in the direction of diversity examines how the

Word Embedding



Network Embedding

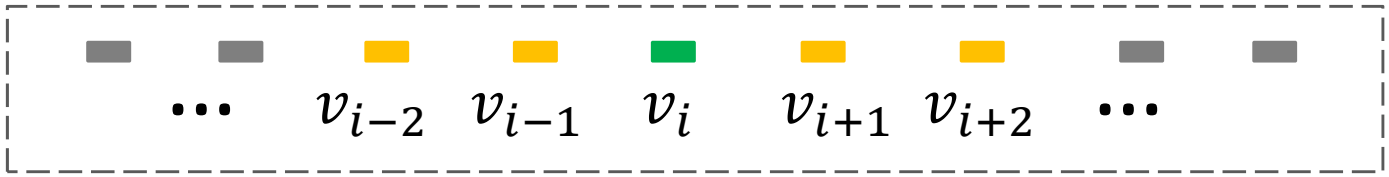
- Input: a text corpus $D = \{V\}$
- Output: $\boldsymbol{Z} \in \mathbb{R}^{|V| \times d}$, $d \ll |V|$, d-dim vector $\boldsymbol{z}_v$ for each node v



Node Sequence

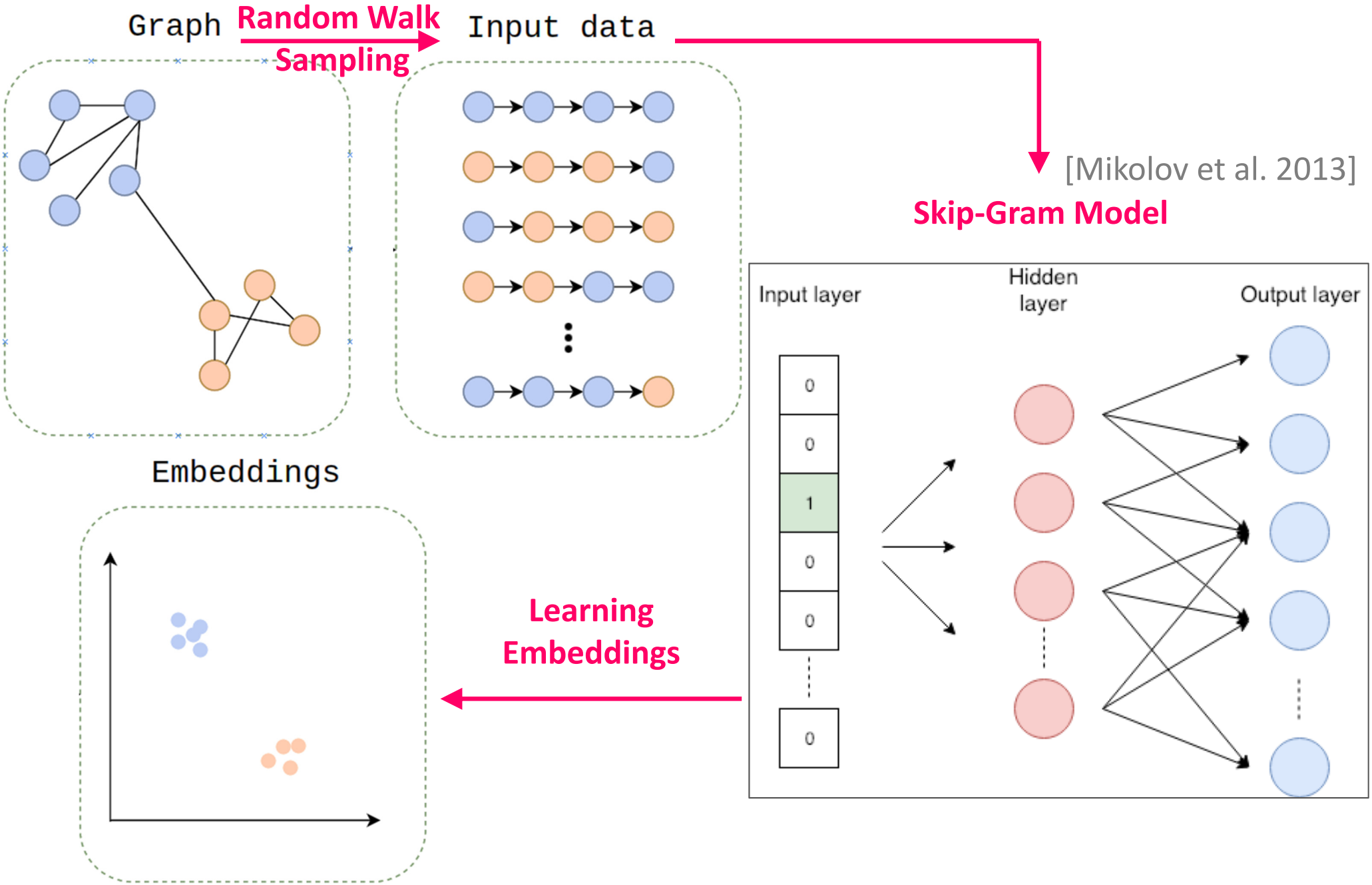
Skip-gram Model

Latent Feature Matrix



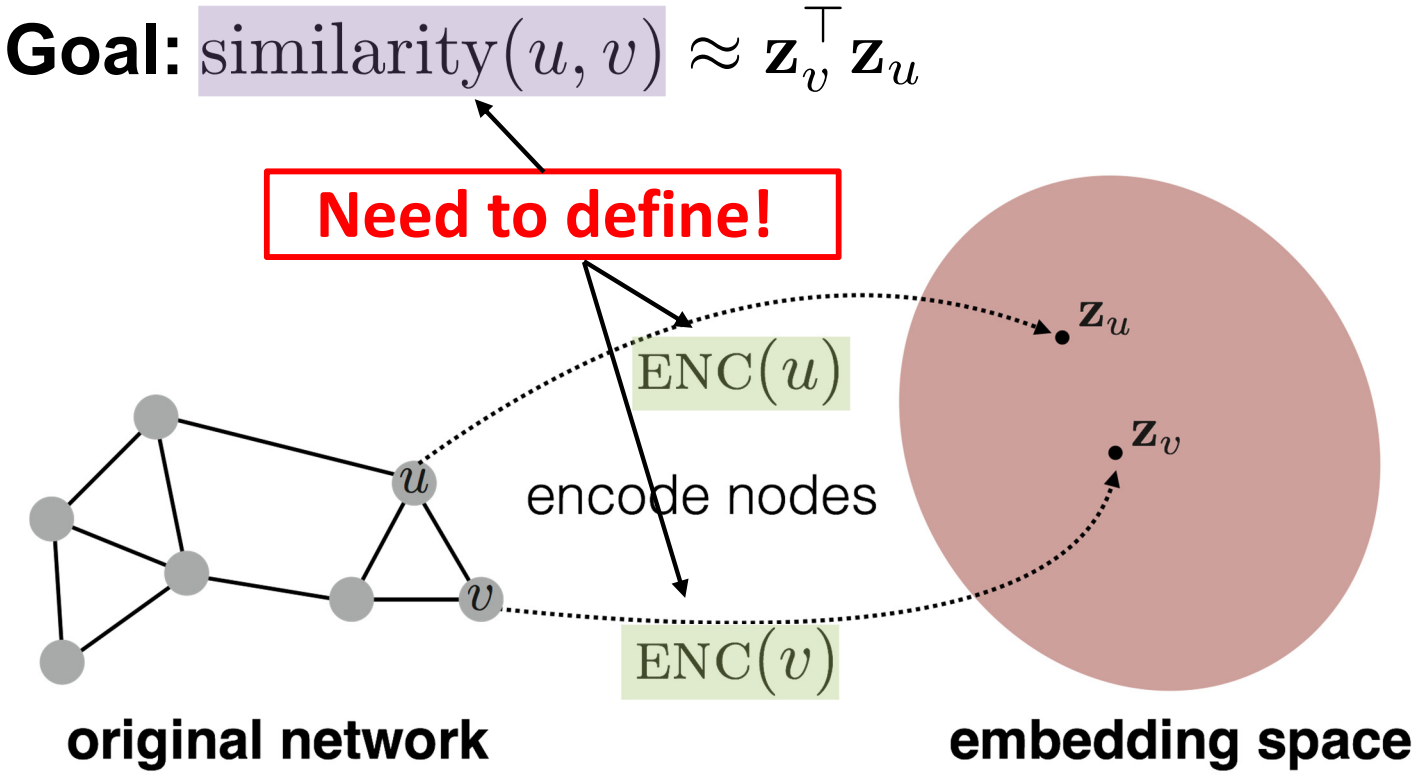
- Nodes in similar contexts have close to each other
- Key idea: try to predict the nodes that surrounding each one

Network Embedding: Basic Idea



Embedding Nodes

- Goal is to encode nodes so that similarity in the embedding space (e.g., dot product) approximates similarity in the original network



Learning Node Embeddings

1. **Define an encoder**
   i.e., a mapping from nodes to embeddings
2. **Define a node similarity function**
   i.e., a measure of similarity in original network
3. **Optimize the parameters of the encoder so that:**

$$\text{similarity}(u, v) \approx \mathbf{z}_v^\top \mathbf{z}_u$$

Two Key Components

- **Encoder** maps each node to a low-dimensional vector

$$\mathrm{ENC}(v) = \mathbf{z}_v$$

node in the input graph — d-dimensional embedding

- **Similarity function** specifies how relationships in vector space map to relationships in the original network

$$\mathrm{similarity}(u, v) \approx \mathbf{z}_v^\top \mathbf{z}_u$$

Similarity of u and v in the original network

dot product between node embeddings

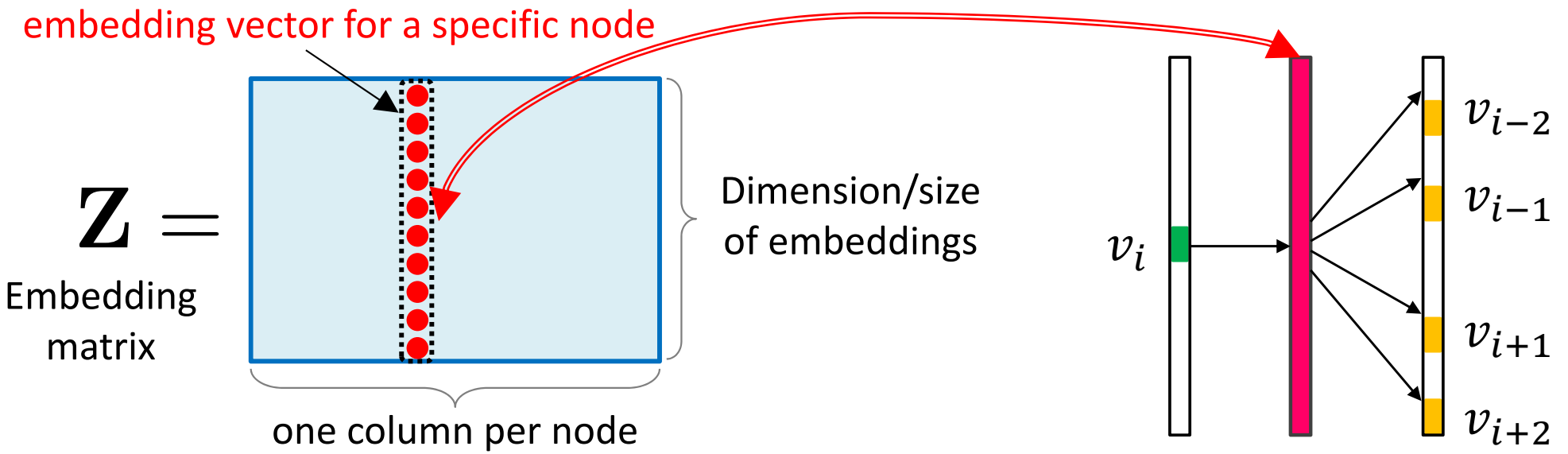
“Shallow” Encoding

- Simplest encoding approach: E.g., node2vec, DeepWalk, LINE
  encoder is just an embedding-lookup

$$\mathrm{ENC}(v) = \mathbf{Z}\mathbf{v}$$

$\mathbf{Z} \in \mathbb{R}^{d \times |\mathcal{V}|}$ matrix, each column is node embedding **[what we learn!]**

$\mathbf{v} \in \mathbb{I}^{|\mathcal{V}|}$ indicator vector, all zeroes except a one in column indicating v



How to Define Node Similarity?

Two nodes have similar embeddings if they:

- are connected:
  - Graph Factorization
  - LINE
- share neighbors:
  - Matrix Factorization-based: GraRep, HOPE
  - Random Walk-based: DeepWalk, LINE, Node2vec
  - Heterogeneous Net: metapath2vec, PTE
- have similar structures:
  - node2vec
  - struct2vec

Graph Factorization [Ahmed et al., 2013]

- **Similarity** function
  - Adjacency-based similarity
  - Or the edge weight between u and v in the network
- Intuition: dot products between node embeddings approximate edge existence

$$\mathcal{L} = \sum_{(u,v)\in V\times V} \|\mathbf{z}_u^\top \mathbf{z}_v - \mathbf{A}_{u,v}\|^2$$

loss (what we want to minimize)

sum over all node pairs

embedding similarity

(weighted) adjacency matrix for the graph

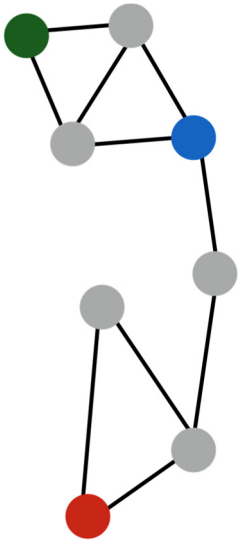
Ahmed et al. Distributed Natural Large Scale Graph Factorization. WWW 2013

Graph Factorization [Ahmed et al., 2013]

$$\mathcal{L} = \sum_{(u,v) \in V \times V} \left\| \mathbf{z}_u^\top \mathbf{z}_v - \mathbf{A}_{u,v} \right\|^2$$

- Goal: find embedding matrix $\mathbf{Z} \in \mathbb{R}^{d \times |V|}$ that minimizes the loss $\mathcal{L}$
  - Option 1: Use stochastic gradient descent (SGD) for optimization
    - Highly scalable, general approach
  - Option 2: Solve matrix decomposition solvers (e.g., SVD or QR)
    - Only works in limited cases
- Drawbacks
  - $O(|V|^2)$ runtime (must consider all node pairs)
  - Only consider direct local connections

    e.g., the blue node is obviously more similar to green compared to red node, despite none having direct connections

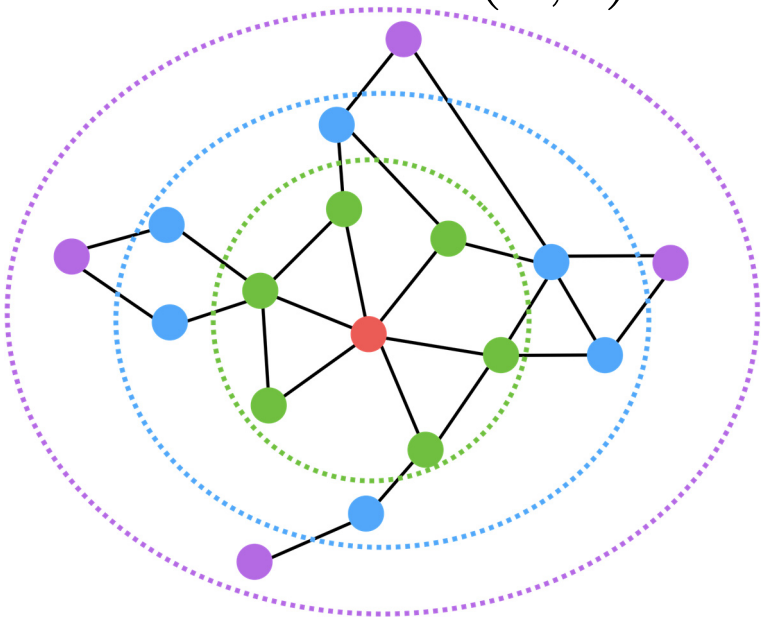


Multi-hop Similarity: GraRep

[Cao et al., 2015]

- **Idea:** Consider k-hop node neighbors
  - E.g., two or three-hop neighbors
- Train embeddings to predict k-hop neighbors

$$\mathcal{L} = \sum_{(u,v)\in V\times V} \|\mathbf{z}_u^\top \mathbf{z}_v - \mathbf{A}_{u,v}^k\|^2$$



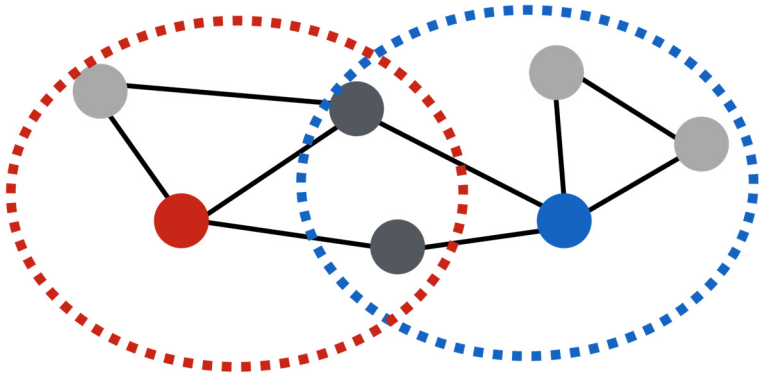
- **Red:** Target node
- **Green**: 1-hop neighbors
  - $\boldsymbol{A}$ (i.e., adjacency matrix)
- **Blue**: 2-hop neighbors
  - $\boldsymbol{A^2}$
- **Purple**: 3-hop neighbors
  - $\boldsymbol{A^3}$

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

Multi-hop Similarity: HOPE

[Ou et al., 2016]

- Another option: measure overlap between node neighborhoods (e.g., Jaccard and Adamic Adar score)



$$\mathcal{L} = \sum_{(u,v)\in V\times V} \| \mathbf{z}_u^\top \mathbf{z}_v - \mathbf{S}_{u,v} \|^2$$

embedding similarity

multi-hop network similarity
(i.e., any neighborhood overlap measure)

$\boldsymbol{S}_{uv}$ is the neighborhood overlap between u and v

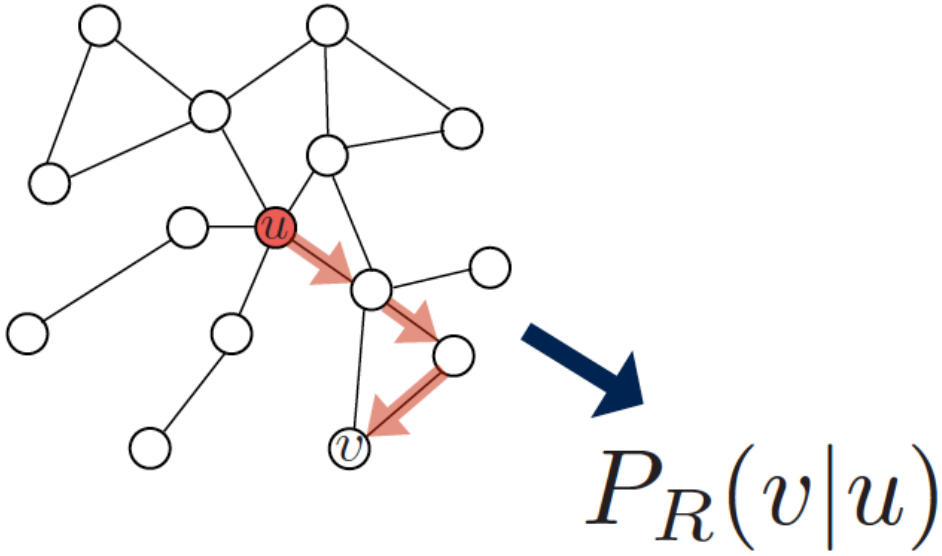
Ou et al. Asymmetric Transitivity Preserving Graph Embedding. KDD 2016

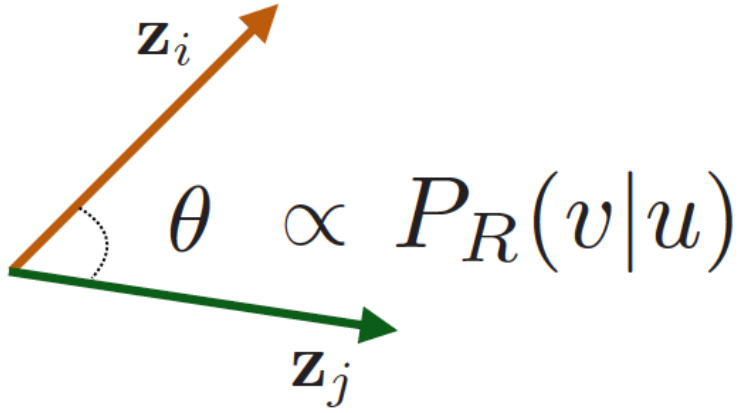
Random-Walk Embeddings

$$\mathbf{z}_u^\top \mathbf{z}_v \approx$$

probability that u and v co-occur on a random walk over the network

- Estimate probability of visiting node v on a random walk starting from node u using some random walk strategy R
- Optimize embeddings to encode these random walk statistics





Why Random Walks?

1. **Expressivity**
   - Flexible stochastic definition of node similarity that incorporates both local and higher-order neighborhood information
2. **Efficiency**
   - Do NOT need to consider all node pairs when training
   - Only need to consider pairs that co-occur on random walks

Random Walk-based Optimization

1) Run short random walks starting from each node on the graph using some RW strategy R
2) For each node u, collect $N_R(u)$, the multiset* of nodes visited on random walks starting from u
3) Optimize embeddings to according to:

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

* $N_R(u)$ can have repeat elements since nodes can be visited multiple times on random walks

Random Walk-based Optimization

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

- **Intuition:** Optimize embeddings to maximize likelihood of random walk co-occurrences
- Parameterize $P(v|\mathbf{z}_u)$ via softmax: $P(v|\mathbf{z}_u) = \frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)}$
- Putting things together:

Optimize random walk embeddings = Find embeddings z_u minimizing L

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log\left(\frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)}\right)$$

sum over all nodes u

sum over nodes v seen on random walks starting from u

predicted probability of u and v co-occuring on random walk

Negative Sampling

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log\left(\frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)}\right)$$

Nested sum over nodes gives $O(|V|^2)$ complexity!!

The normalization term from the softmax is the culprit! Can we approximate it?

$$\log\left(\frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)}\right)$$

$$\approx \log(\sigma(\mathbf{z}_u^\top \mathbf{z}_v)) - \sum_{i=1}^{k} \log(\sigma(\mathbf{z}_u^\top \mathbf{z}_{n_i})), n_i \sim P_V$$

sigmoid function

random distribution over all nodes

i.e., instead of normalizing w.r.t. all nodes,
just normalize against k random "**negative samples**"

Note 1: Sample negative nodes proportional to degree
Note 2: Higher k gives more robust estimates

Random Walk-based Optimization

1) Run short random walks starting from each node on the graph using some RW strategy R
2) For each node u, collect $N_R(u)$, the multiset* of nodes visited on random walks starting from u

How should we randomly walk to obtain $N_R(u)$?

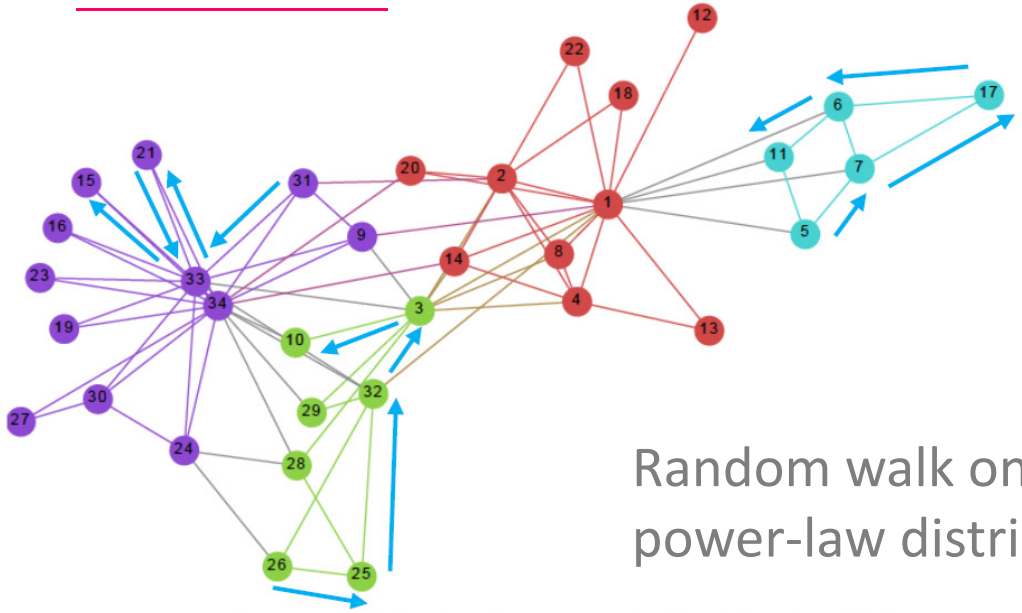
3) Optimize embeddings to according to:

$$\mathcal{L} = \sum_{u \in V} \sum_{v \in N_R(u)} -\log(P(v|\mathbf{z}_u))$$

We can efficiently approximate this using **negative sampling**!

DeepWalk: Truncated RW [Perozzi et al. 2014]

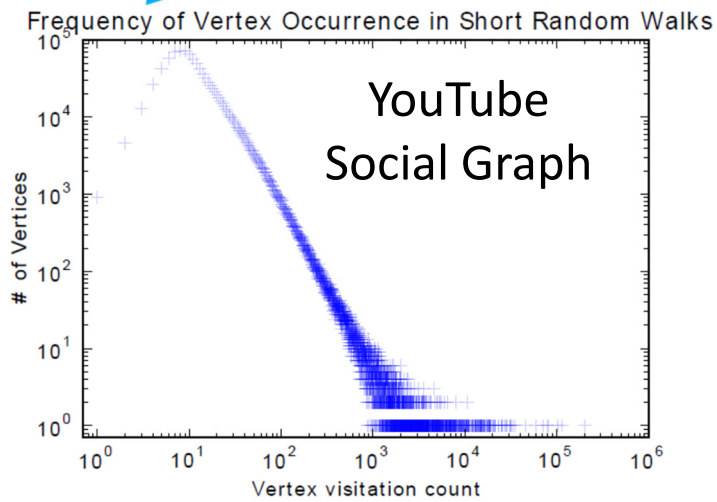
- Truncated random walk: just run fixed-length, **unbiased** random walks starting from each node

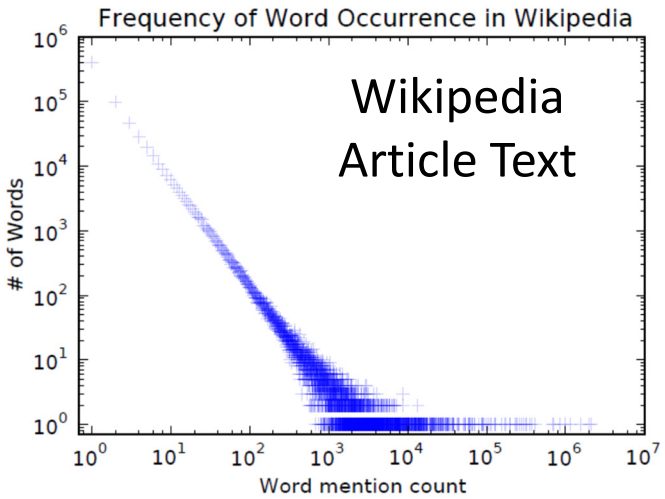


Random Walks on Graph

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

Random walk on networks can generate similar power-law distributions as word distributions in text!





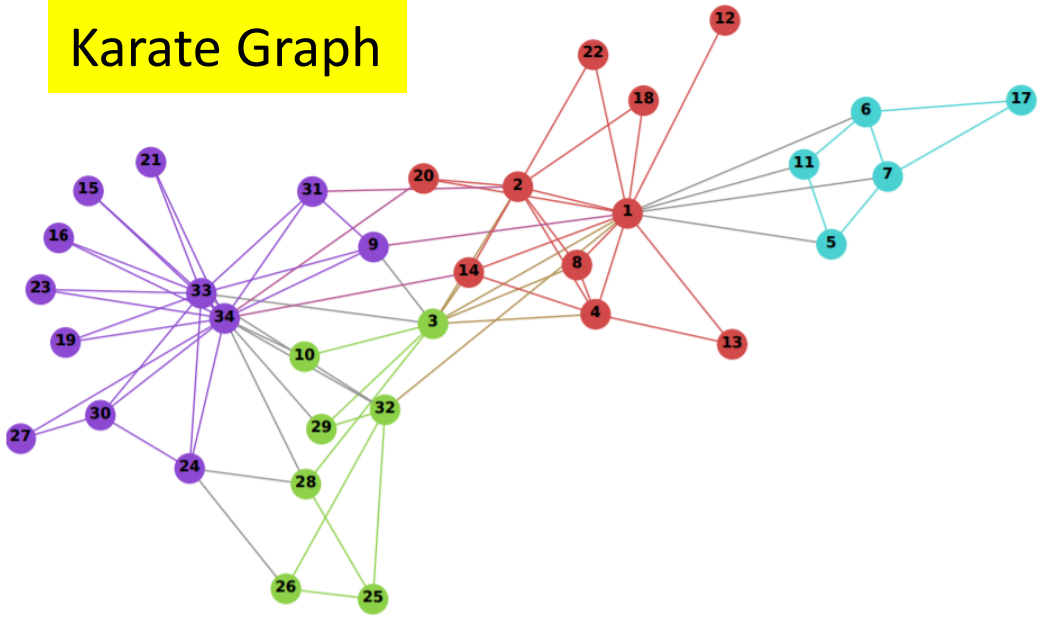
Perozzi et al. DeepWalk: Online Learning of Social Representations. KDD 2014

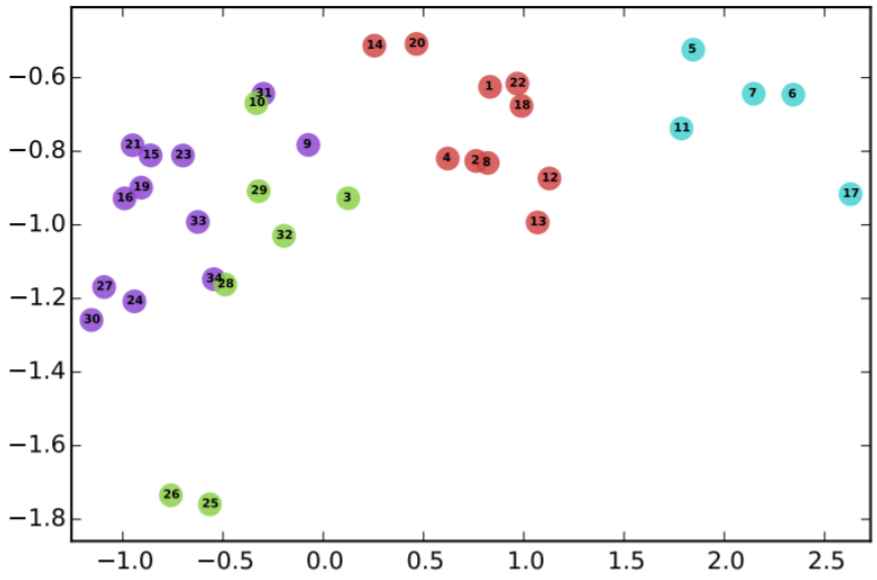
DeepWalk on Visualization & Classification

Node Classification

	% Labeled Nodes	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Micro-F1(%)	DeepWalk	**32.4**	**34.6**	**35.9**	**36.7**	**37.2**	**37.7**	**38.1**	**38.3**	**38.5**	**38.7**
	SpectralClustering	27.43	30.11	31.63	32.69	33.31	33.95	34.46	34.81	35.14	35.41
	EdgeCluster	25.75	28.53	29.14	30.31	30.85	31.53	31.75	31.76	32.19	32.84
	Modularity	22.75	25.29	27.3	27.6	28.05	29.33	29.43	28.89	29.17	29.2
	wvRN	17.7	14.43	15.72	20.97	19.83	19.42	19.22	21.25	22.51	22.73
	Majority	16.34	16.31	16.34	16.46	16.65	16.44	16.38	16.62	16.67	16.71
Macro-F1(%)	DeepWalk	**14.0**	17.3	**19.6**	**21.1**	**22.1**	**22.9**	**23.6**	**24.1**	**24.6**	**25.0**
	SpectralClustering	13.84	**17.49**	19.44	20.75	21.60	22.36	23.01	23.36	23.82	24.05
	EdgeCluster	10.52	14.10	15.91	16.72	18.01	18.54	19.54	20.18	20.78	20.85
	Modularity	10.21	13.37	15.24	15.11	16.14	16.64	17.02	17.1	17.14	17.12
	wvRN	1.53	2.46	2.91	3.47	4.95	5.56	5.82	6.59	8.00	7.26
	Majority	0.45	0.44	0.45	0.46	0.47	0.44	0.45	0.47	0.47	0.47

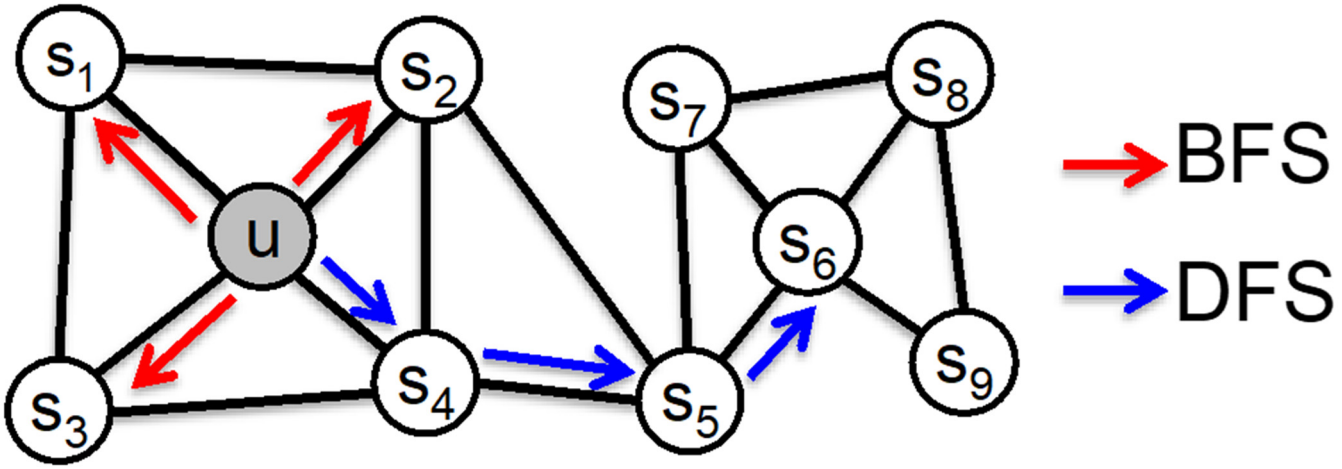
Karate Graph





node2vec: Biased RW [Grover et al., 2016]

- **Idea:** use flexible, biased random walks that can trade off between **local** and **global** views of the network
- Two classic strategies to define a neighborhood $N_R(u)$ of a given node u:



$N_{BFS}(u) = \{s_1, s_2, s_3\}$ Local microscopic view

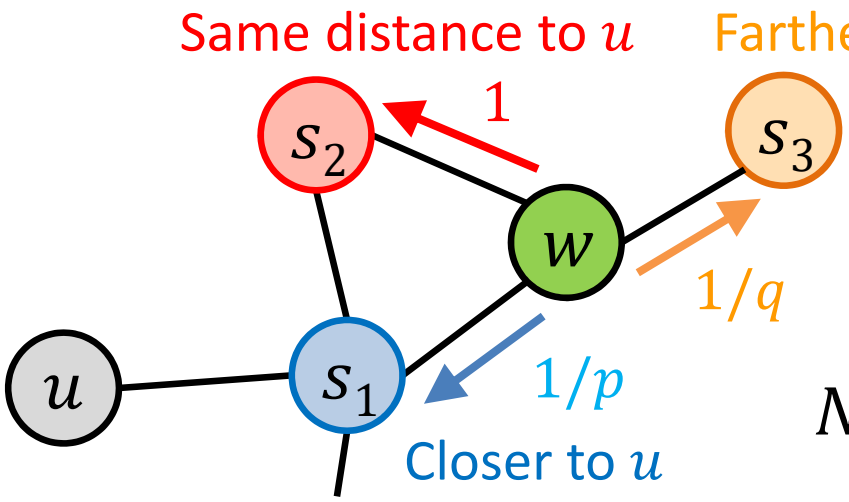
$N_{DFS}(u) = \{s_4, s_5, s_6\}$ Global macroscopic view

Grover et al. node2vec: Scalable Feature Learning for Networks. KDD 2016

node2vec: Biased RW

- Biased random walk R that given a node u generates neighborhood $N_R(u)$ with two parameters
  - Return parameter p: **Return** back to the previous node
  - In-out parameter q: Moving **outwards** (DFS) vs. inwards (BFS)
- **Idea:** Remember where that walk came from
  - RW started at u and is now at w
  - Insight: neighbors of w can only be

$1/p, 1/q, 1$ are unnormalized transition probabilities

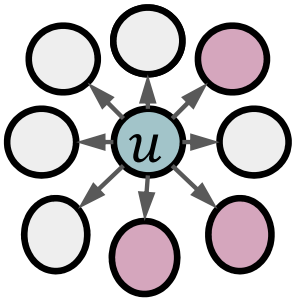


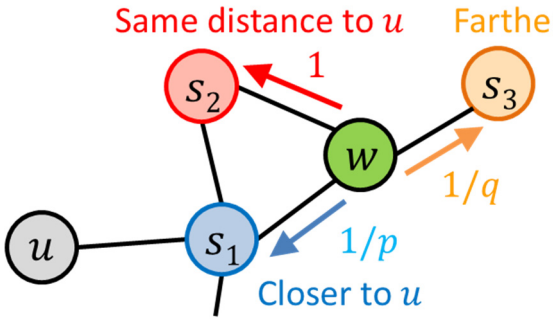
BFS-like walk: Low value of p

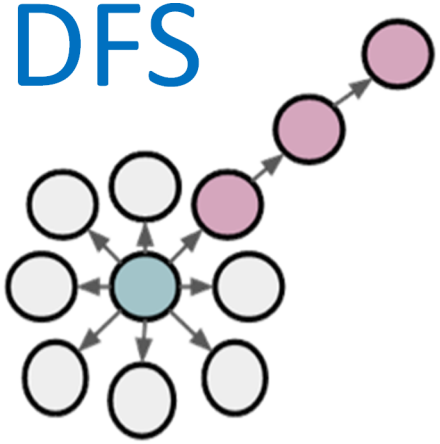
DFS-like walk: Low value of q

$N_S(u)$ are the nodes visited by the walker

node2vec: BFS vs. DFS





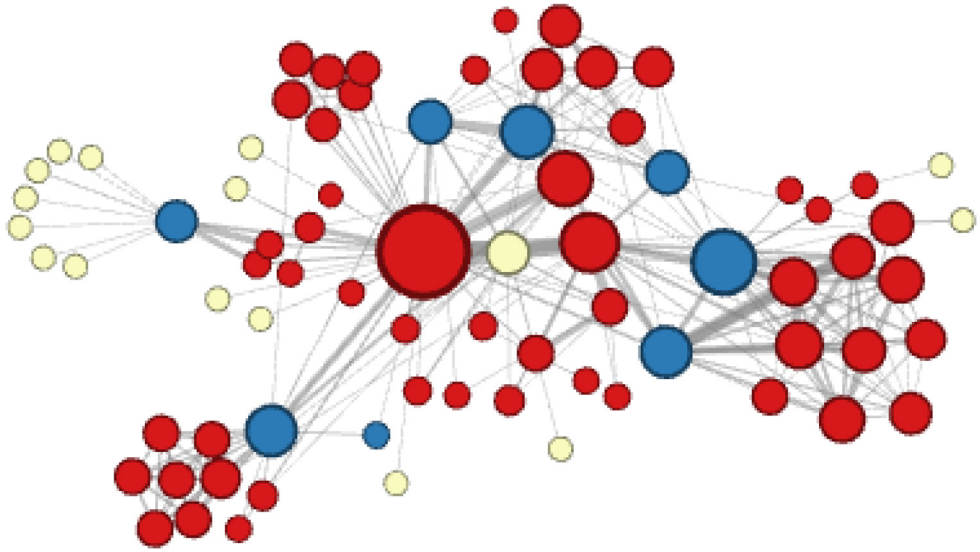


Breadth-First Search:

Micro-view of neighborhood

Depth-First Search:

Macro-view of neighborhood



$p = 1, q = 2$

Structural Equivalence

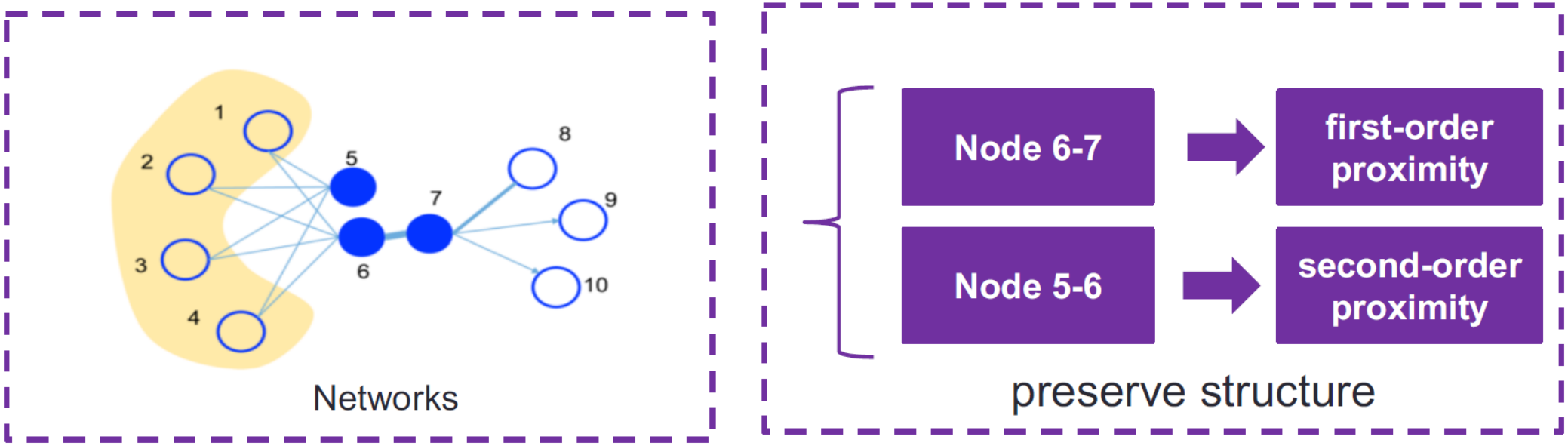
$p = 1, q = 0.5$

Homophily (Community)

LINE: Proximity-based Optimization

[Tang et al., 2015]

- First-order Proximity: directly-connected
  - However, many links between the nodes are not observed
- Second-order Proximity:
  Proximity between the neighborhood structures of nodes
  - *"The degree of overlap of two people's friendship networks correlates with the strength of ties between them"* – Mark Granovetter



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

LINE: Preserving the **First-order** Proximity

- Given an ***undirected*** edge (v_i, v_j), the joint probability of v_i, v_j

$$p_1(v_i, v_j) = \frac{1}{1 + \exp(-\vec{u}_i^T \cdot \vec{u}_j)}$$

$\vec{u}_i$: Embedding of vertex v_i

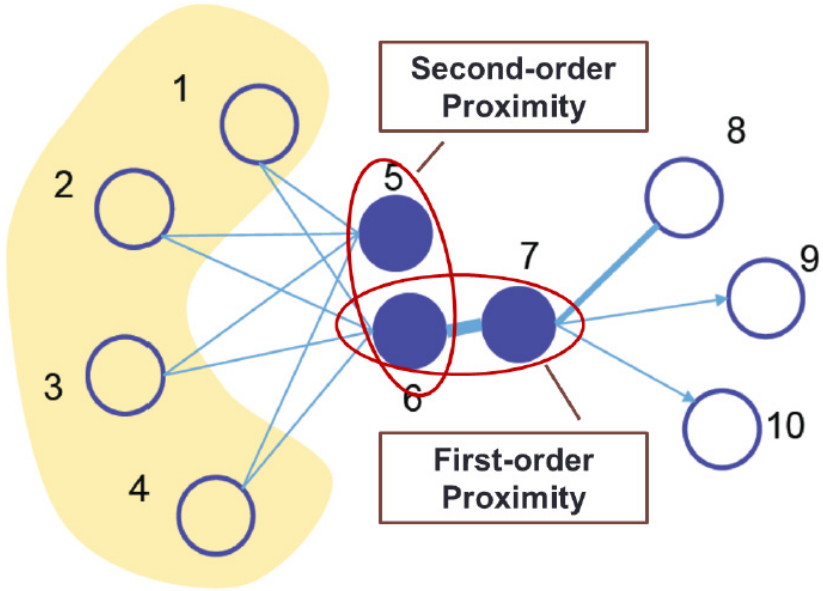
$$\hat{p}_1(v_i, v_j) = \frac{w_{ij}}{\sum_{(i',j')} w_{i'j'}}$$

- Objective:

$$O_1 = d(\hat{p}_1(\cdot,\cdot), p_1(\cdot,\cdot))$$

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

d: KL-divergence



LINE: Preserving **Second-order** Proximity

- Given a ***directed*** edge (v_i, v_j),
  the conditional probability of v_j given v_i is:

[undirected graph is viewed as bi-directed graph]

$$p_2(v_j|v_i) = \frac{\exp(\vec{u}_j'^T \cdot \vec{u}_i)}{\sum_{k=1}^{|V|} \exp(\vec{u}_k'^T \cdot \vec{u}_i)}$$

$$\hat{p}_2(v_j|v_i) = \frac{w_{ij}}{\sum_{k \in V} w_{ik}}$$

$\vec{u}_i$: Embedding of vertex *i* when *i* is source node;
$\vec{u}_i'$: Embedding of vertex *i* when *i* is target node.

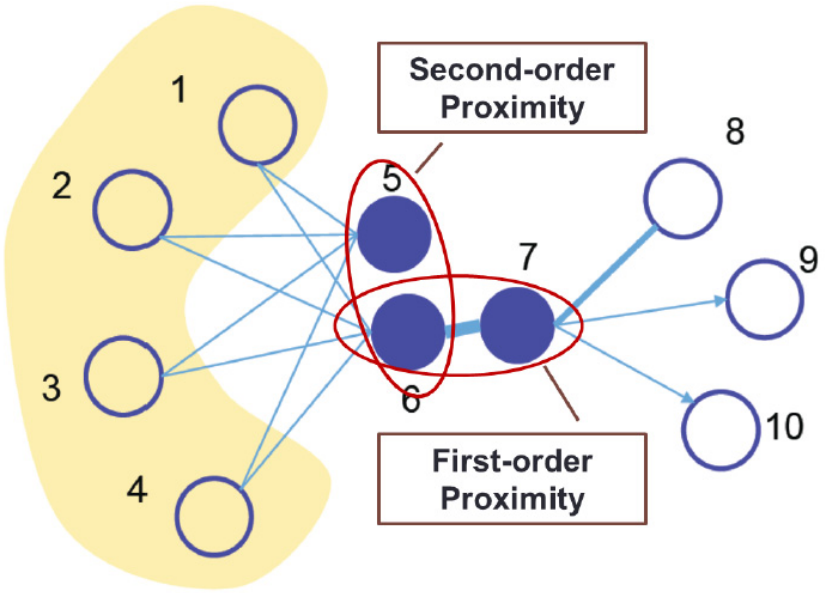
λ_i: Prestige of vertex in the network

$\lambda_i = \sum_j w_{ij}$

- Objective:

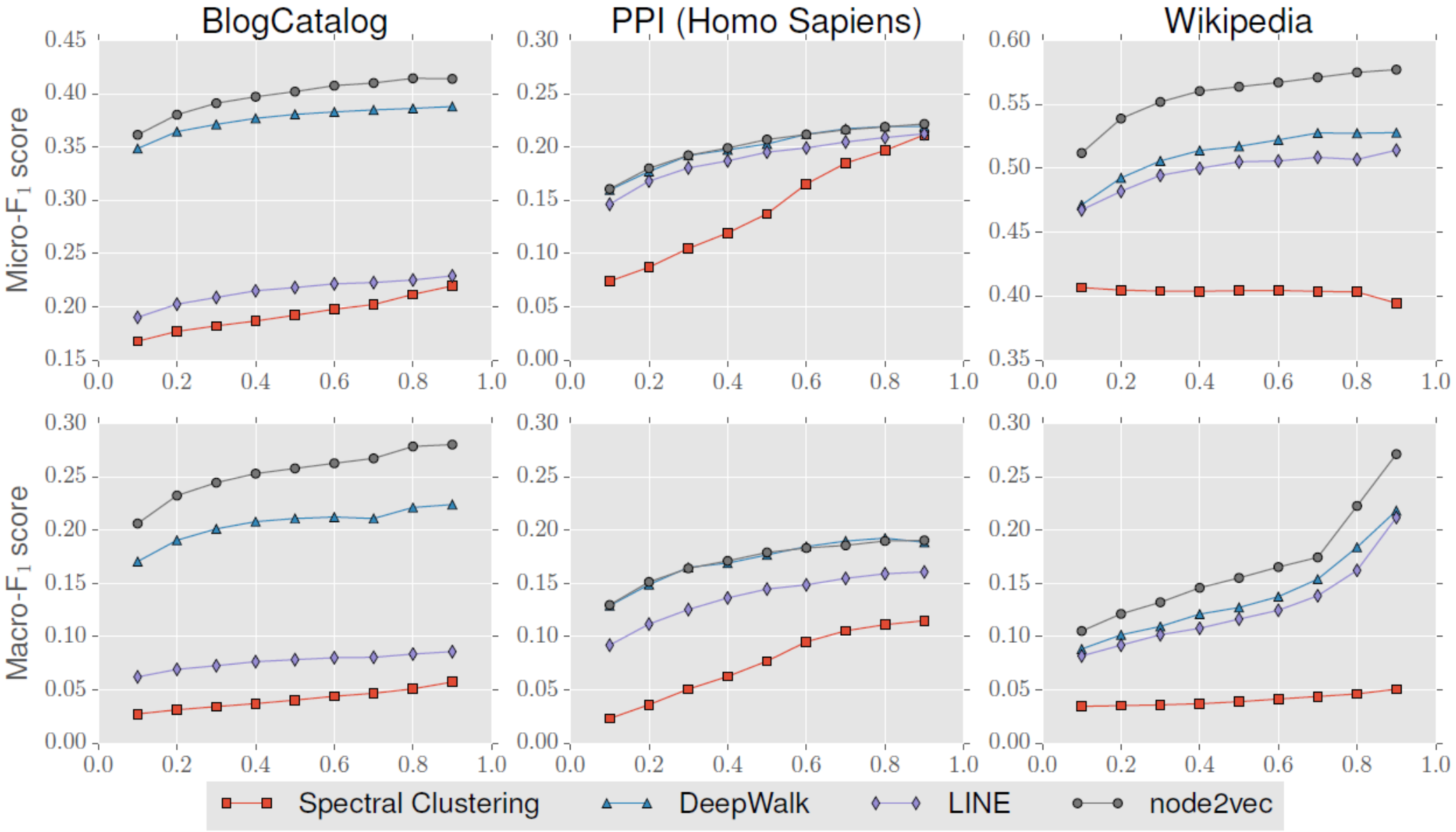
$$O_2 = \sum_{i \in V} \lambda_i d(\hat{p}_2(\cdot|v_i), p_2(\cdot|v_i))$$

$$\propto -\sum_{(i,j) \in E} w_{ij} \log p_2(v_j|v_i)$$



First-order ⊕ Second-order

Performance on Node Classification



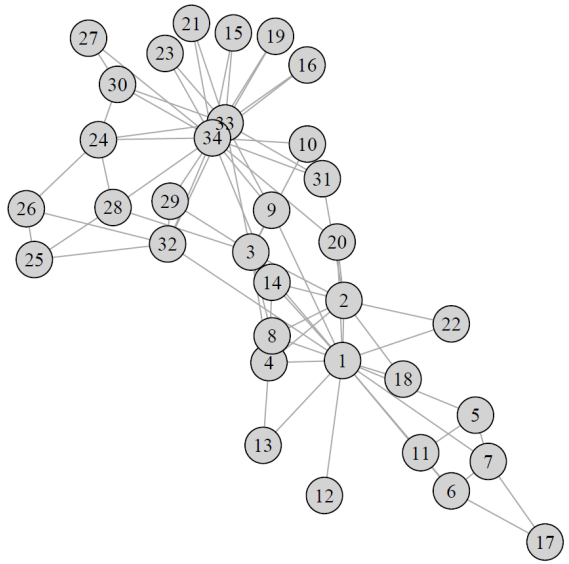
Incorporating Global Information

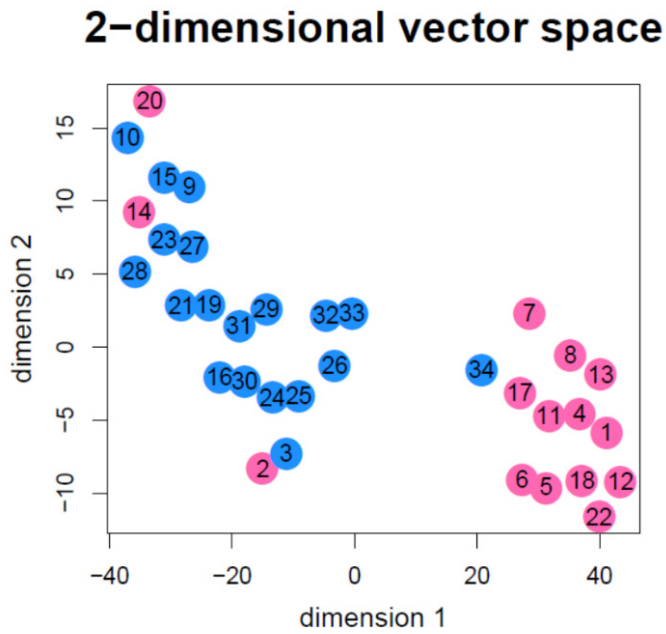
Existing NRL

DeepWalk [Perozzi et al. KDD'14]

LINE [Tang et al. WWW'15]

node2vec [Grover et al. KDD'16]

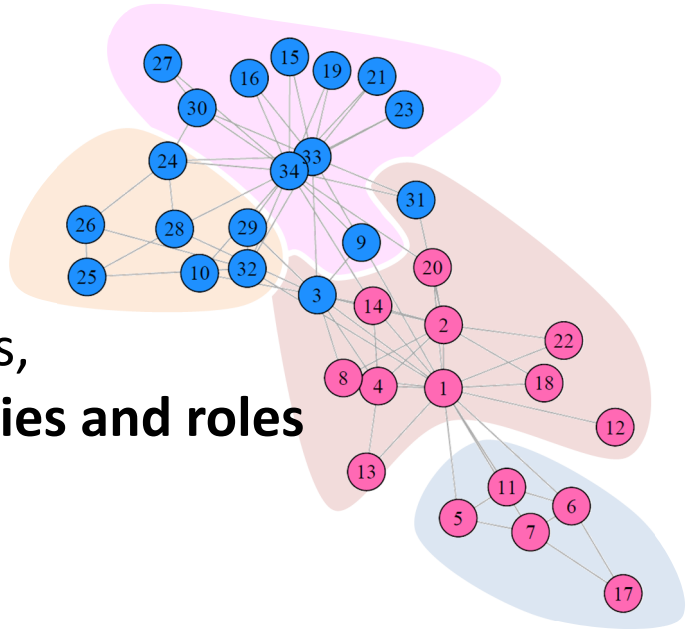


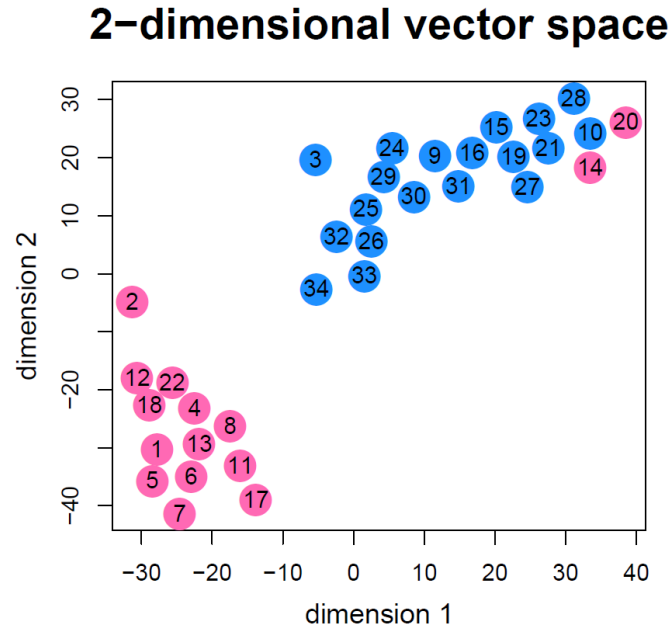


Our Proposal

NRL with global info

In addition to local neighbors,
let nodes see the communities and roles
in embedding learning

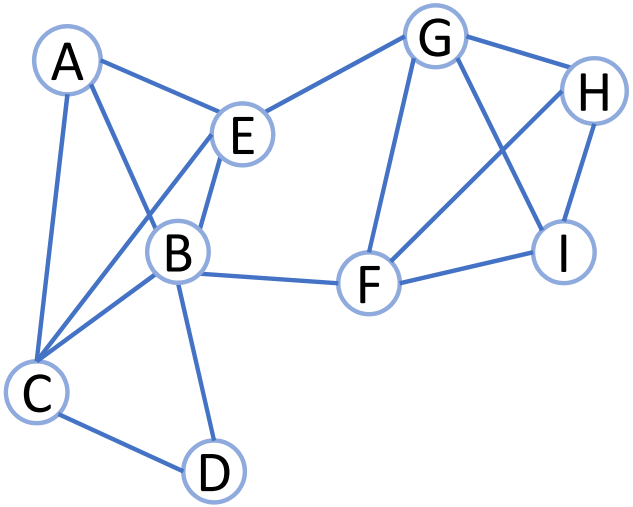




Coarsening-enhanced NRL

Original Graph

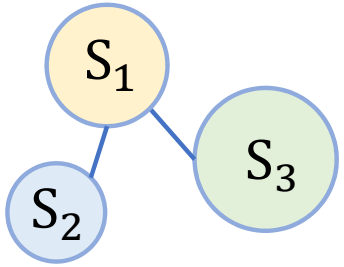
$G^0 = (V^0, E^0)$



Network Coarsening

Level-1 Super graph

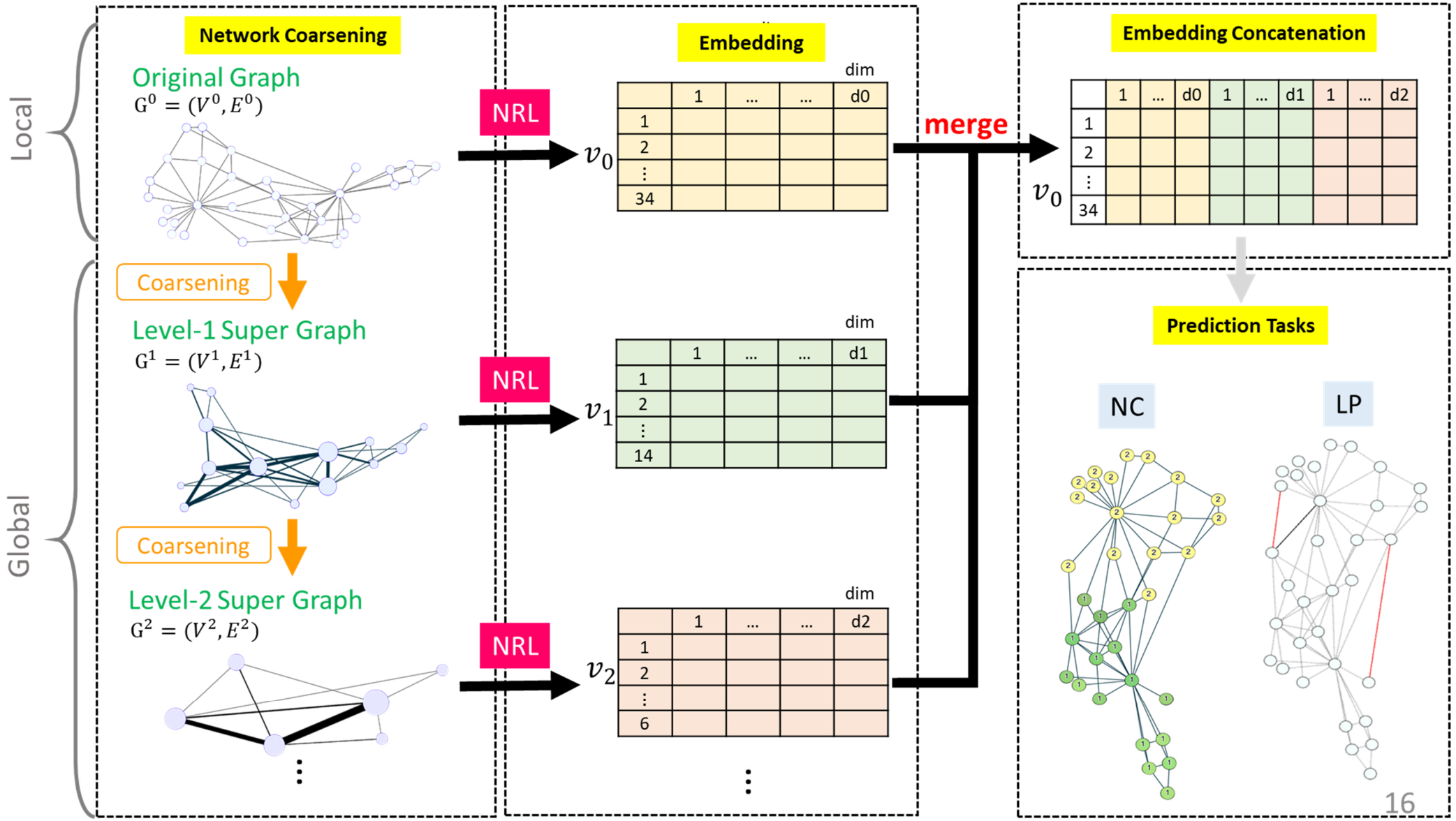
$G^1 = (V^1, E^1)$



How to incorporate global information ?

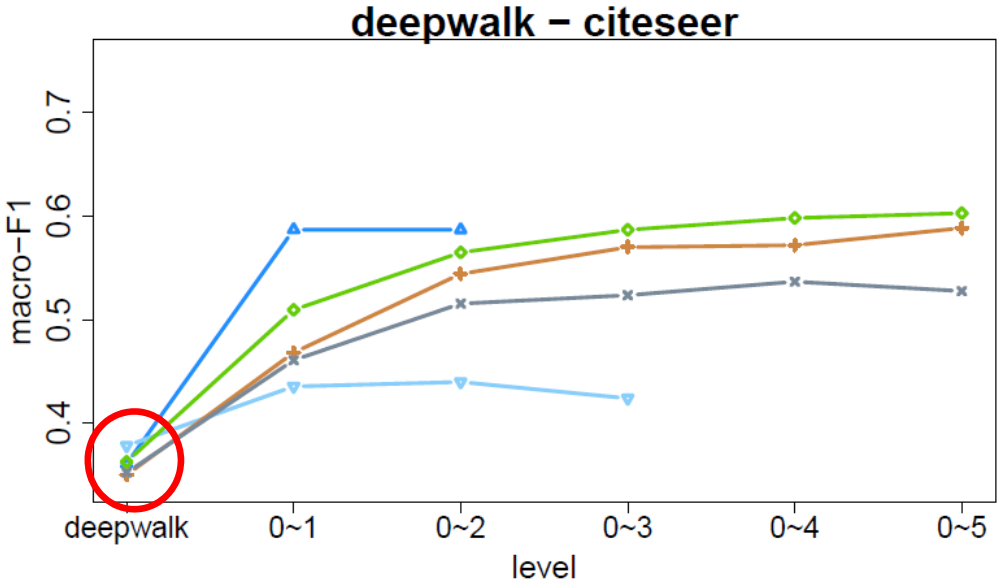
1) Community-based Coarsening
2) Social Circle-based Coarsening
3) Star-based Coarsening

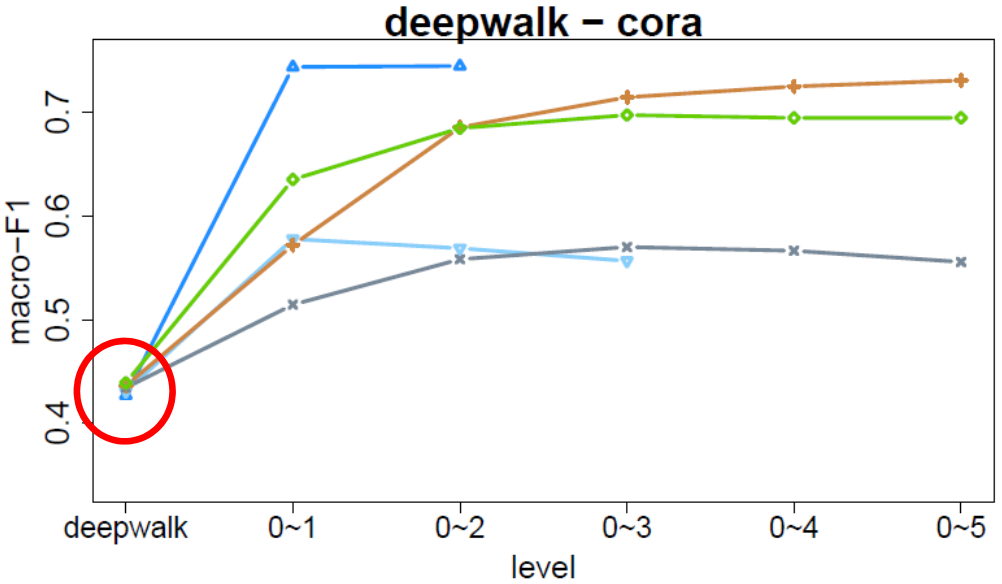
Coarsening-enhanced NRL

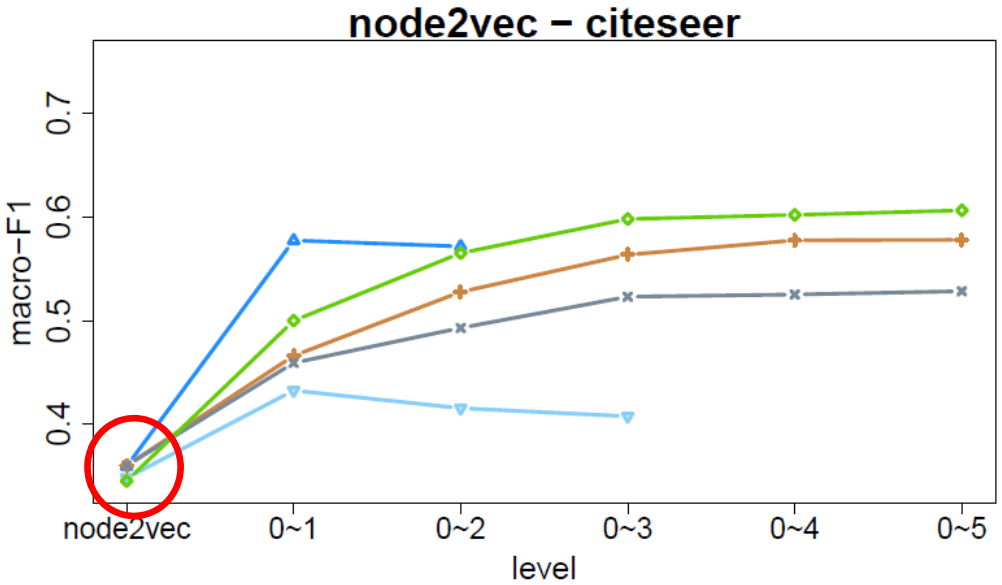


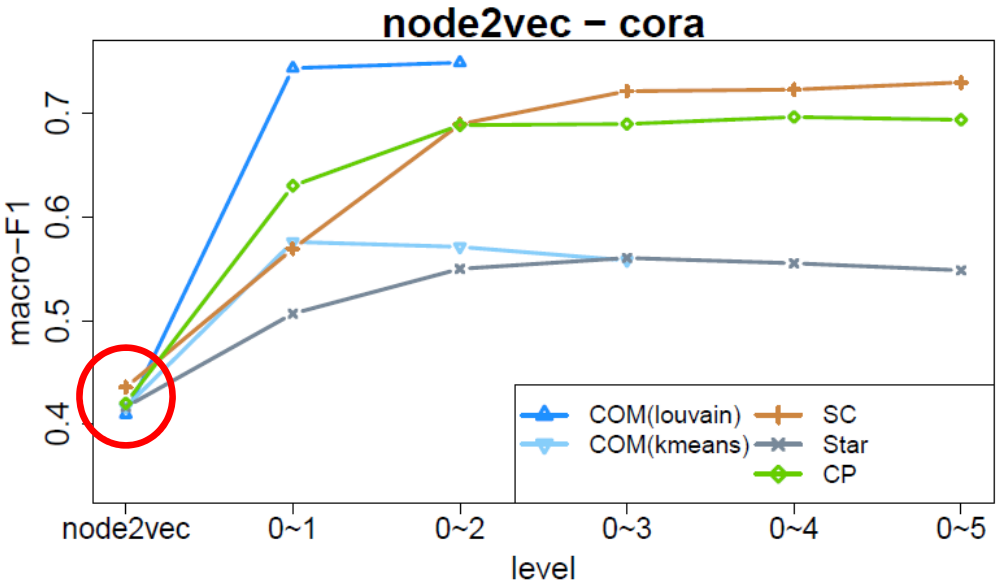
* Community-based Coarsening: by Louvain, Kmeans

Results on Node Classification



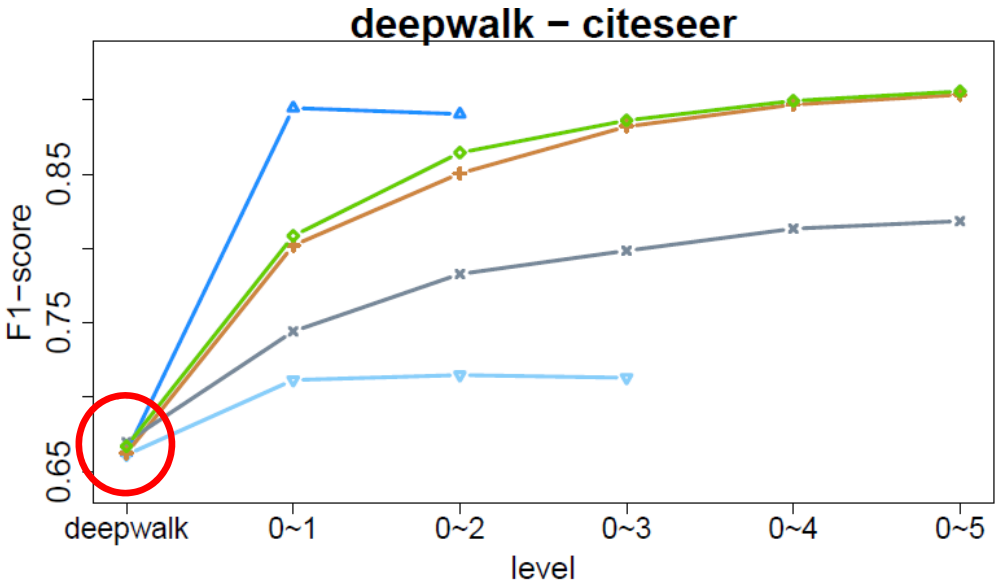


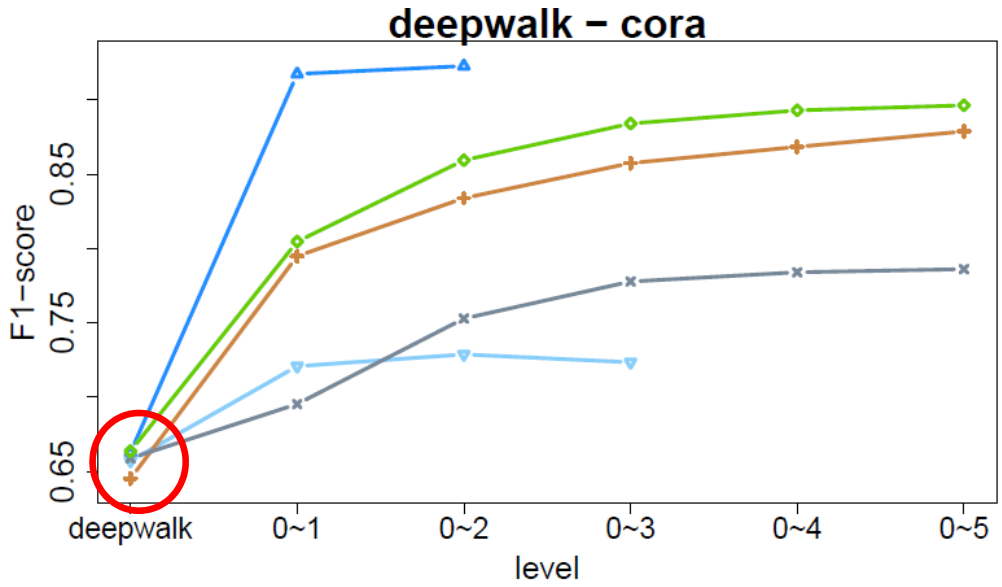


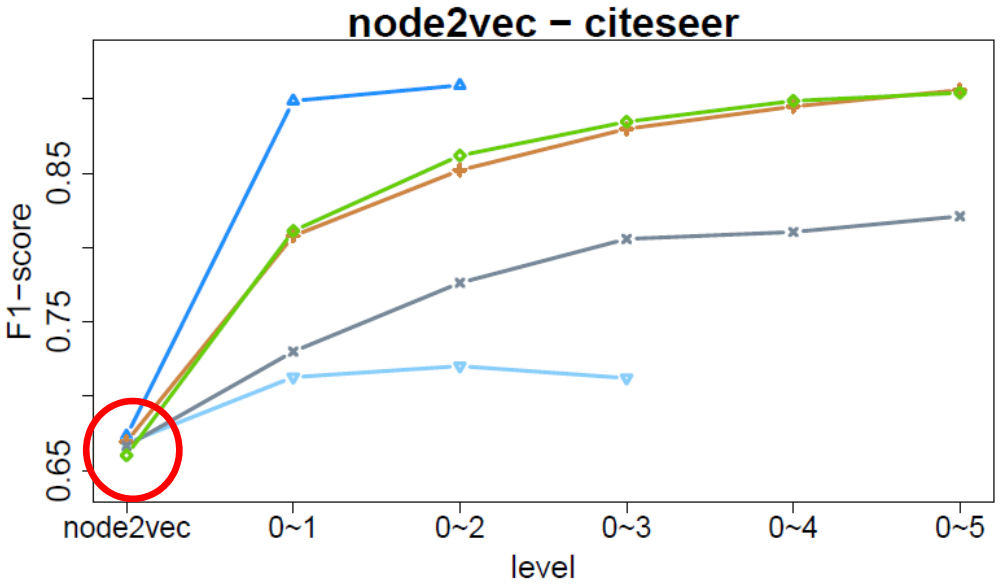


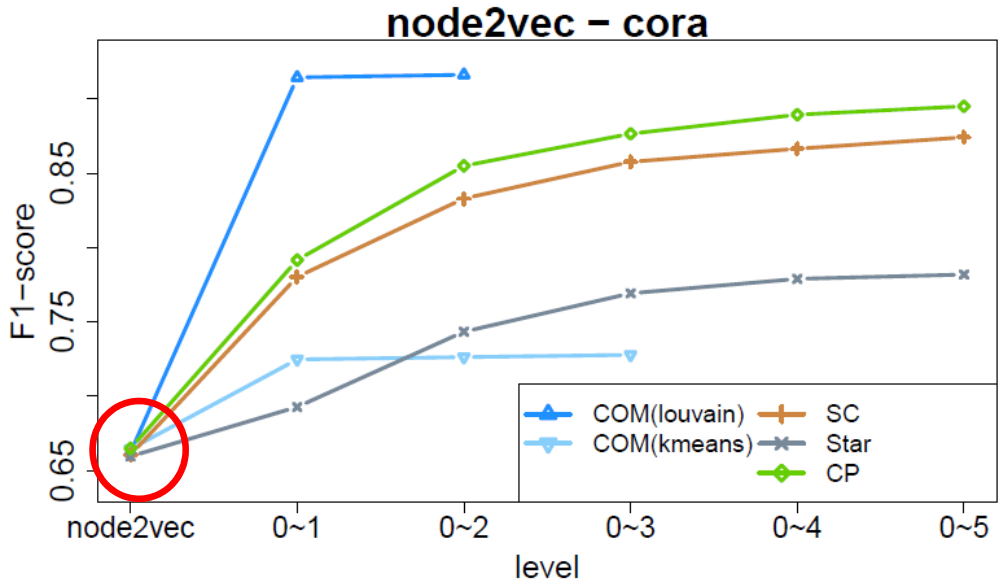
* logistic regression

Results on Link Prediction





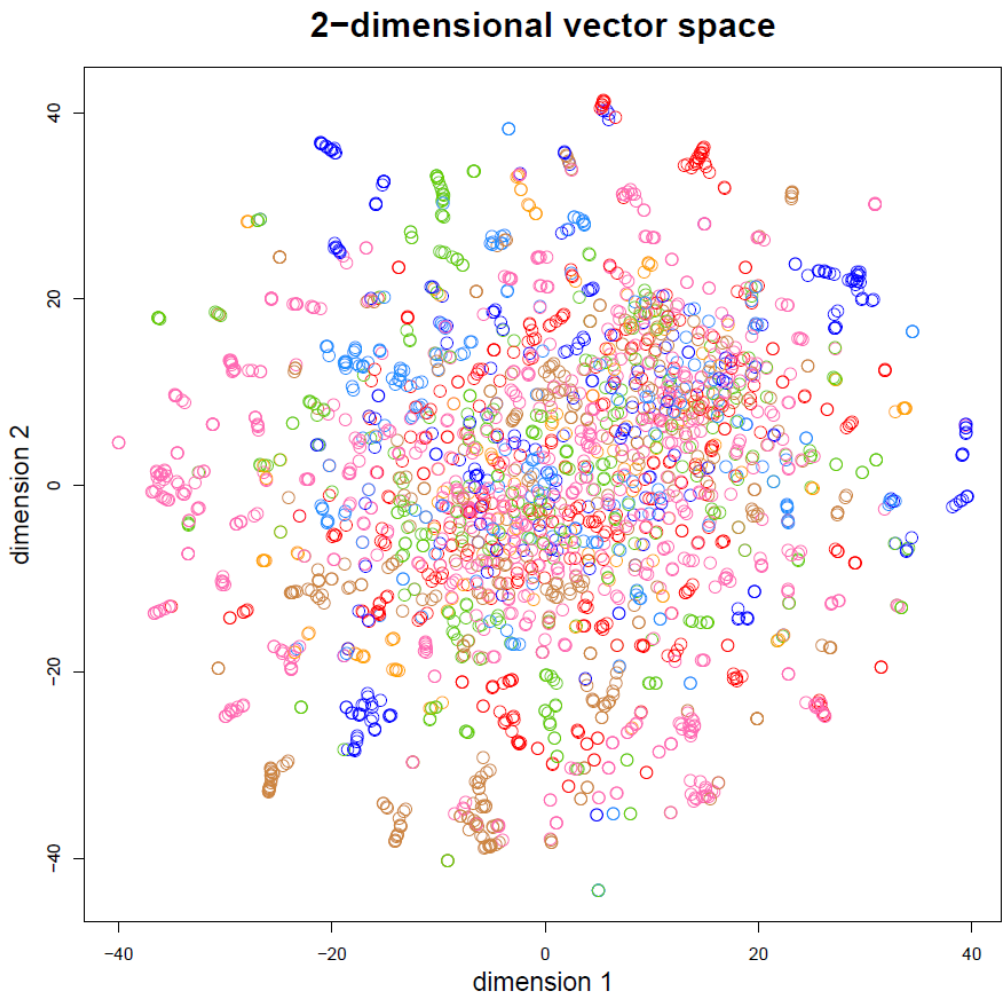




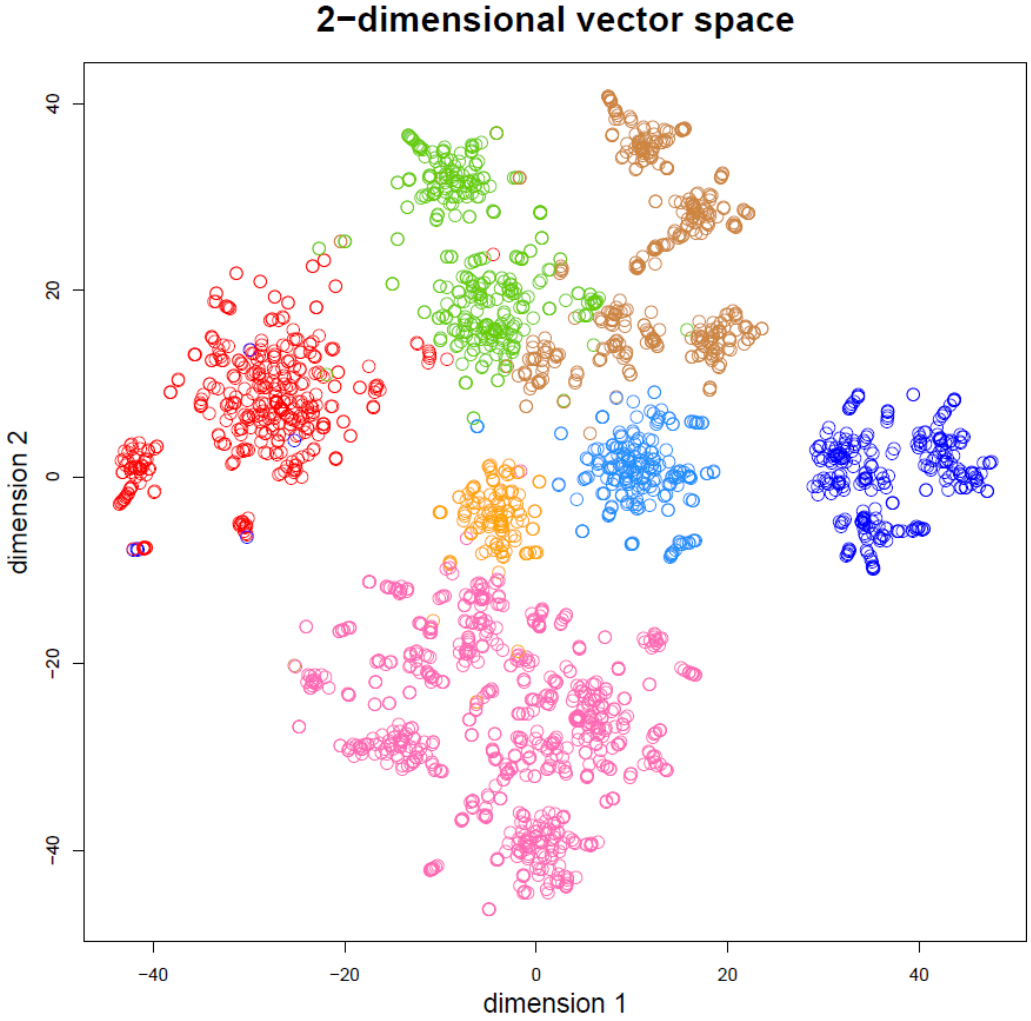
* logistic regression (50% training)

tSNE Visualization (data: cora graph)

node2vec [Grover et al. KDD'16]



Coarsening-node2vec



References

- A. Ahmed et al. “**Distributed Large-scale Natural Graph Factorization**” WWW 2013 **444 cites**
- S. Cao et al. “**GraRep: Learning Graph Representations with Global Structural Information**” ACM CIKM 2015 **1033 cites**
- M. Ou et al. “**Asymmetric Transitivity Preserving Graph Embedding**” ACM KDD 2016 **731 cites**
- B. Perozzi et al. “**DeepWalk: Online Learning of Social Representations**” ACM KDD 2014 **4964 cites**
- A. Grover and J. Leskovec. “**node2vec: Scalable Feature Learning for Networks**” ACM KDD 2016 **4914 cites**
- J. Tang et al. “**LINE: Large-scale Information Network Embedding**” WWW 2015 **3234 cites**
- H.-Y. Lin and C.-T. Li. “**Structural Hierarchy-Enhanced Network Representation Learning**” Applied Sciences 2020