Graph Embedding

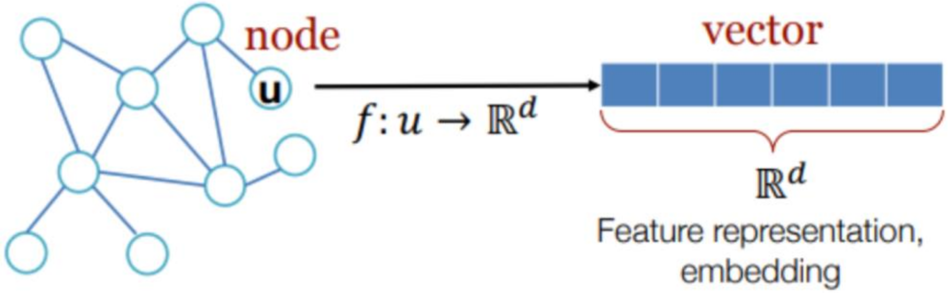
Prof. O-Joun Lee

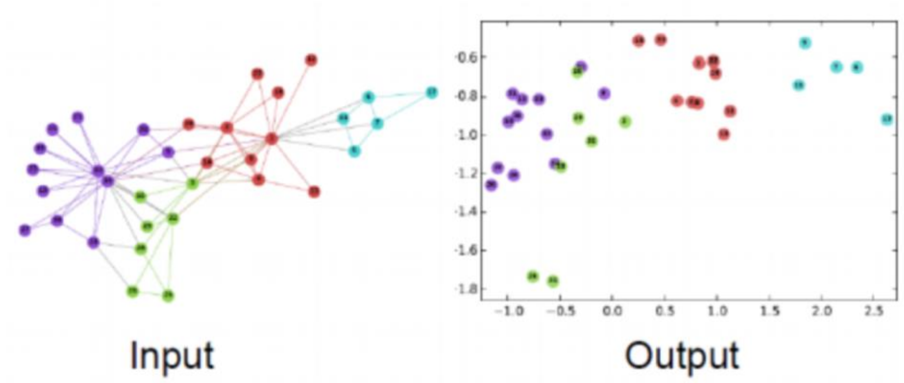
Dept. of Artificial Intelligence,
The Catholic University of Korea
ojlee@catholic.ac.kr

Contents

- Graph representation learning
- Representative models
  - Node2Vec
  - LINE
  - SDNE
  - Subgraph2Vec

- The objective of a graph embedding: nodes which are similar in the graph, should be mapped close in the vector space
- Schematic of graph (node) embedding





Graph representation learning: a survey

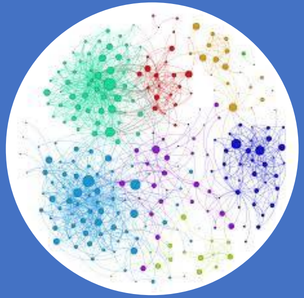
Applications of Graph Embedding

Network Compression

- A compression store networks more efficiently run graph algorithms faster

Visualization

- View it from an information visualization perspective



Community Mining

- Network partitioning (k-means on the embedding to cluster the nodes)

Link Prediction

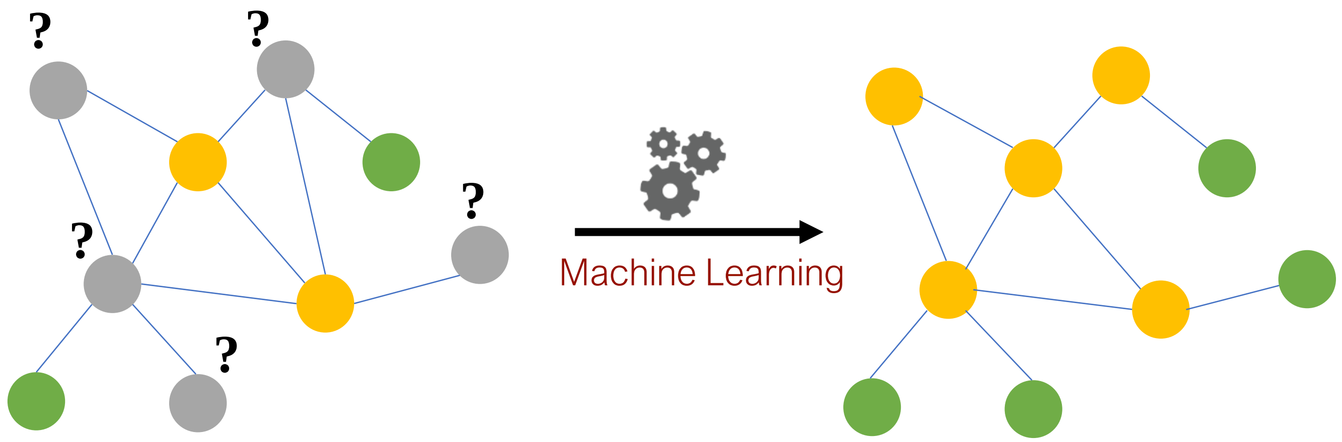
- Predict either missing interactions or links that may appear in the future

Node Classification

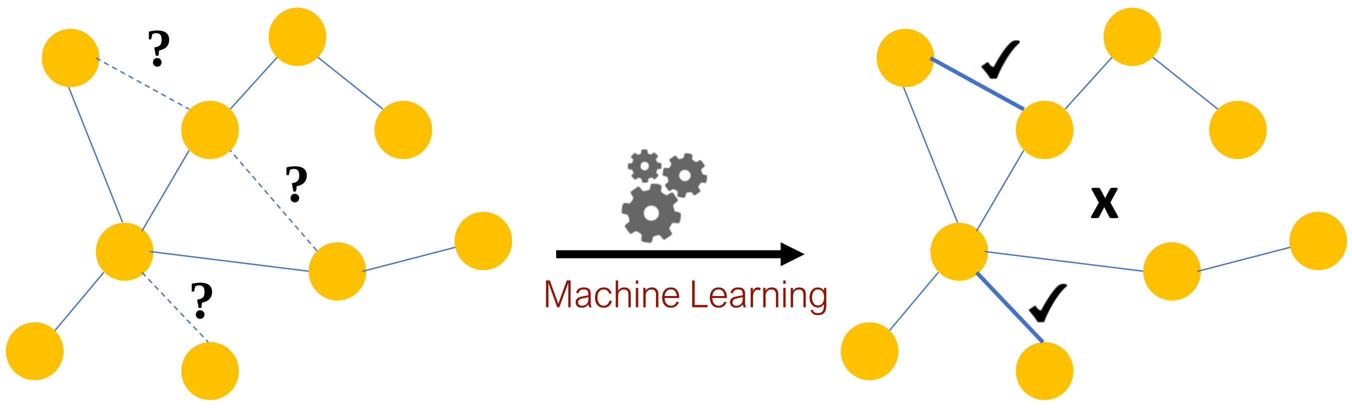
- Label the full graph based only on this small initial seed set

Example: Applications of Graph Embedding

➢ Node Classification is a machine learning task in graph-based data analysis, where the goal is to assign labels to nodes in a graph based on the properties



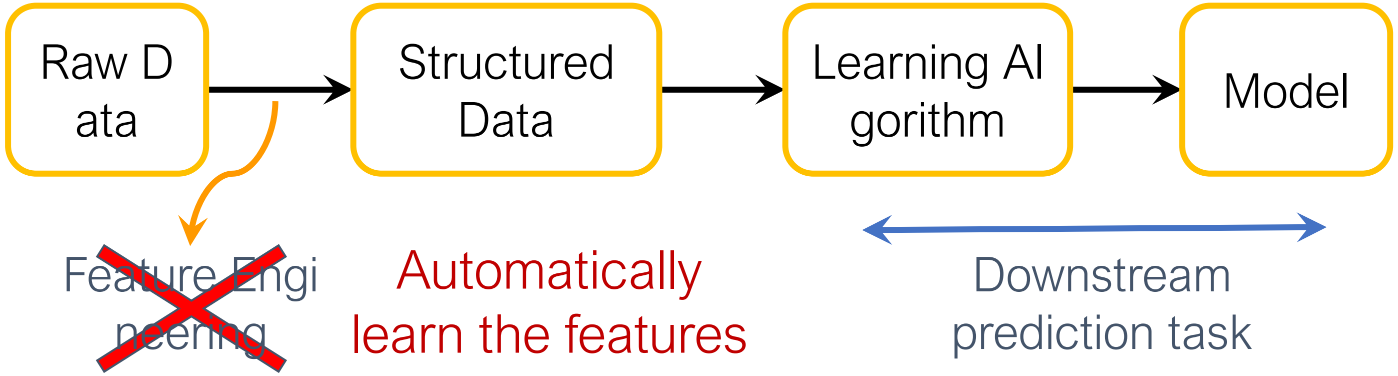
- **Link Prediction** is a task in graph and network analysis where the goal is to predict missing or future connections between nodes in a network.



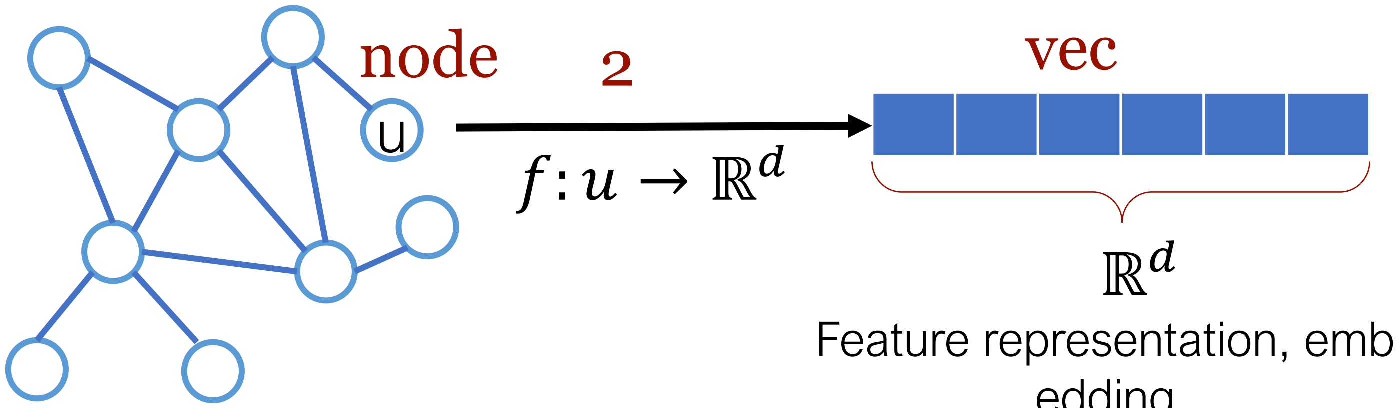
Representation Learning on Networks, snap.stanford.edu/proj/embeddings-www, WWW 2018

Machine Learning Lifecycle

➢ (Supervised) Machine Learning Lifecycle: This feature, that feature. Every single time!



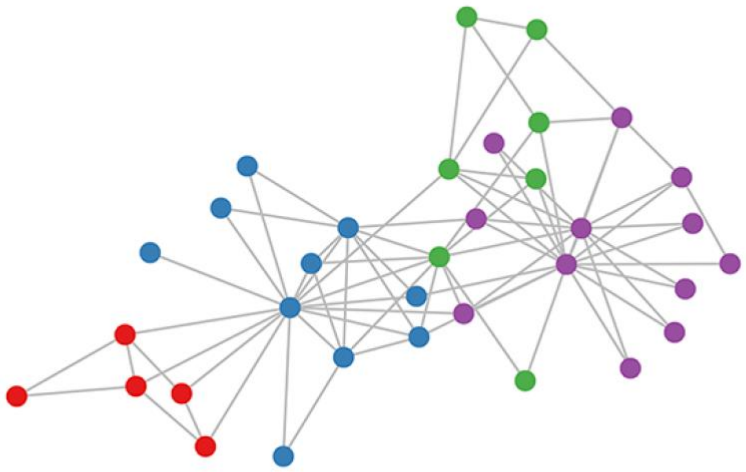
- **Goal:** Efficient task-independent feature learning for machine learning in networks!



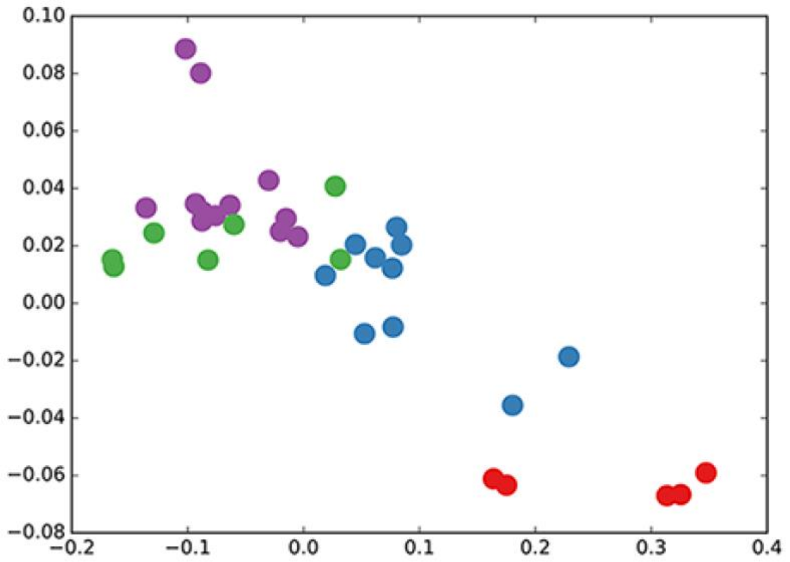
Feature representation, emb
edding

Example

- A good embedding should capture the graph topology, vertex-to-vertex relationship, and other relevant information about the graph, its subgraphs, and vertices.



Input

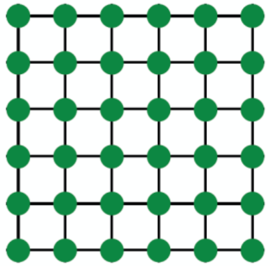


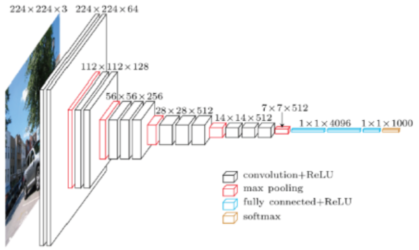
Output

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

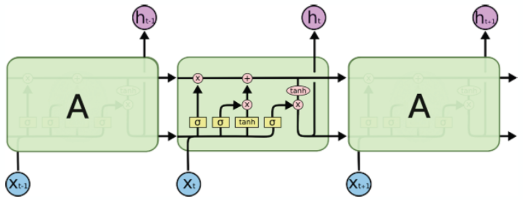
Why is it hard?

- Modern deep learning toolbox is designed for simple sequences or grids.
- CNNs for fixed-size images/grids….



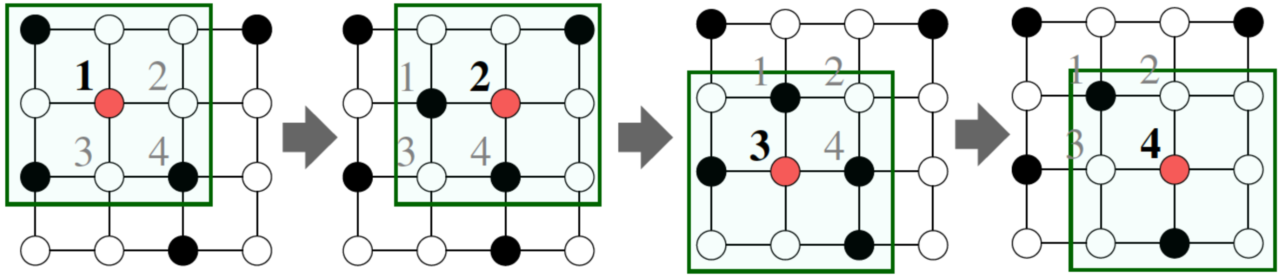


- RNNs or word2vec for text/sequences…



Why is it hard?

- ➢ But networks are far more complex
  - ➢ Complex topographical structure (i.e., no spatial locality like grids)



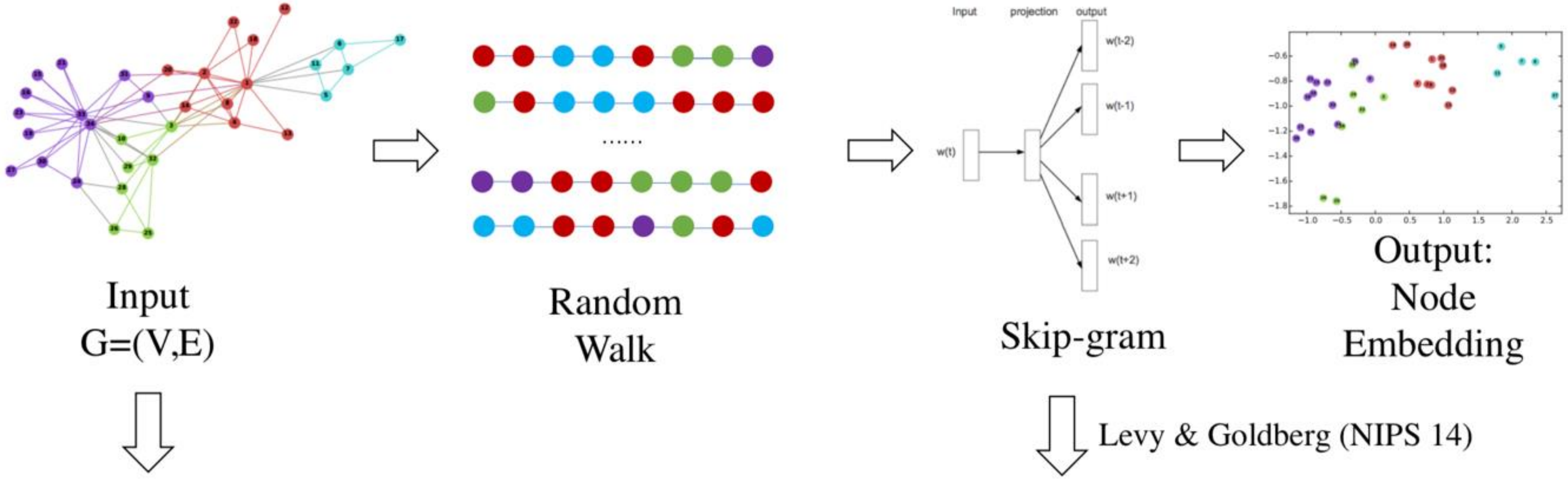
  - ➢ No fixed node ordering or reference point (i.e., the isomorphism problem)
  - ➢ Often dynamic and have multimodal features.

Typical Network Embedding Approaches

- Random walk approaches
  - Node2Vec
- Multi-hop Similarity
  - LINE
- Deep model (Autoencoder)
  - SDNE
- Subgraph learning
  - Subgraph2Vec

More details: graph representation learning and its applications: a survey

Overview of random walk–based methods

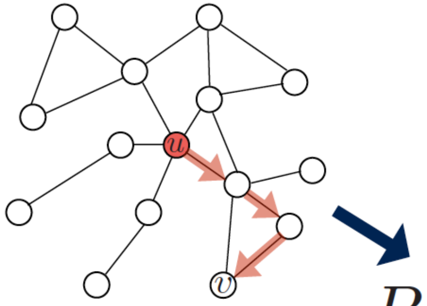


?

$$\log\left(\frac{\#(w,c)\,|\mathcal{D}|}{b\#(w)\#(c)}\right)$$

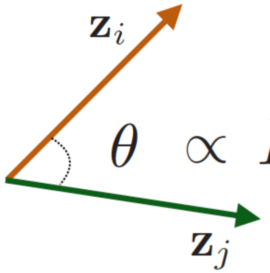
		b	Number of negative samples
$\#(w,c)$	Co-occurrence of w and c	$\#(w)$	Occurrence of word w
$\lvert\mathcal{D}\rvert$	Total number of word-context pairs	$\#(c)$	Occurrence of context c

- Estimate probability of visiting node *v* on a random walk starting from node *u* using some random walk strategy *R*.



$$P_R(v|u)$$

- Optimize embeddings to encode these random walk statistics.



$$\theta \propto P_R(v|u)$$

Why Random Walks?

- Expressivity:
  - Flexible stochastic definition of node similarity that incorporates both local and higher-order neighbourhood information.
- Efficiency:
  - Do not need to consider all node pairs when training; only need to consider pairs that co-occur on random walks.

- Run short random walks starting from each node on the graph using some strategy R.
- For each node u collect N(u), the multiset* of nodes visited on random walks starting from u.
- Optimize embeddings to according to:

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

- * N(u) can have repeat elements since nodes can be visited multiple times on random walks.

- Intuition: Optimize embeddings to maximize likelihood of random walk co-occurrences.

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

- Parameterize P(v | z_u) using softmax:

$$P(v|\mathbf{z}_u) = \frac{\exp(\mathbf{z}_u^\top \mathbf{z}_v)}{\sum_{n \in V} \exp(\mathbf{z}_u^\top \mathbf{z}_n)}$$

Random Walk Optimization

- Putting things together:
- Optimizing random walk embeddings =

  Finding embeddings z_u that minimize L

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

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

- But doing this naively is too expensive.

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

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

The normalization term from the softmax is the problem

Can we approximate it?

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

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

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

sigmoid function

random distribution over all nodes

Random Walks: Stepping Back

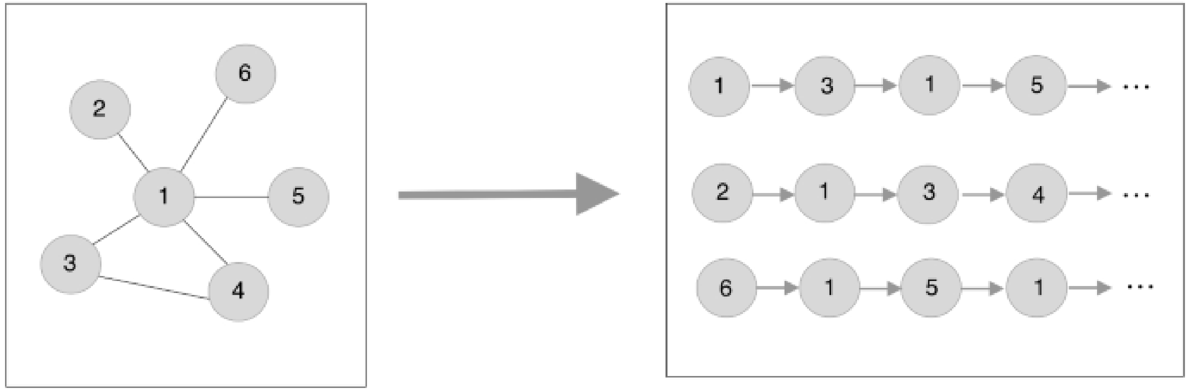
- Run short random walks starting from each node on the graph using some strategy R.
- For each node u collect NR(u), the multiset of nodes visited on random walks starting from u.
- Optimize embeddings to according to:

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

We can efficiently approximate this using negative sampling!

- What strategies should we use to run these random walks?

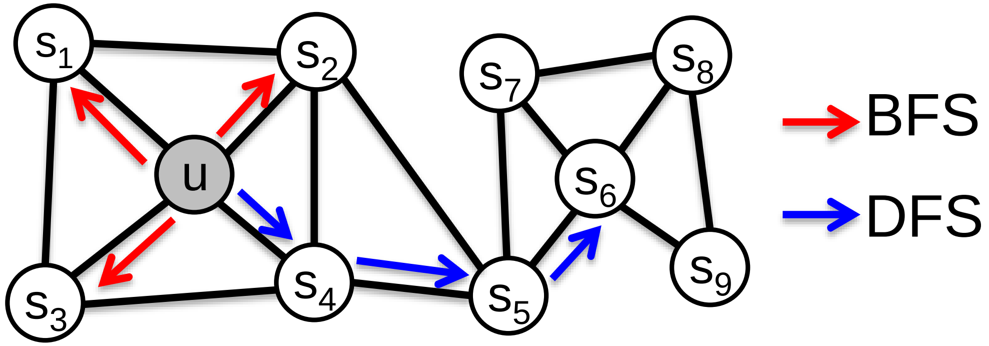
  - Simplest idea: Just **run fixed-length, unbiased random walks** starting from each node (i.e., DeepWalk from Perozzi et al., 2013).



- But can we do better?

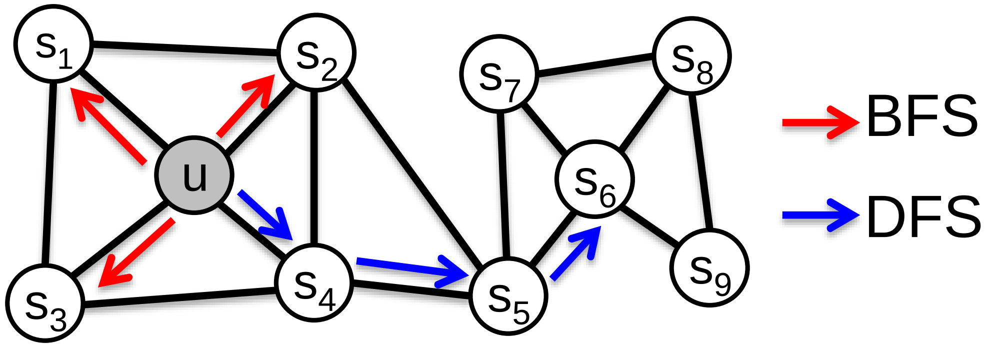
Node2Vec: Biased Walks

- Idea:
  - use flexible, biased random walks that can **trade-off between local and global views** of the network (Grover and Leskovec, 2016).



node2vec: Biased Walks

- Two classic strategies to define a neighborhood $N(u)$ of a given node u:



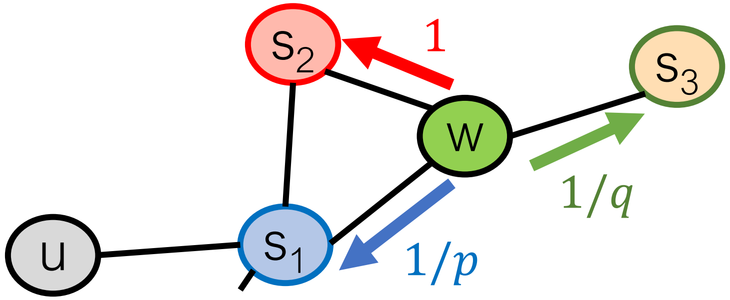
$N_{BFS}(u) = \{ s_1, s_2, s_3\}$ Local microscopic view

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

Interpolating BFS and DFS

- Biased random walk R that given a node u generates neighborhood $N(u)$
- Two parameters:
  - Return parameter p:
    - Return back to the previous node
  - In-out parameter q:
    - Moving outwards (DFS) vs. inwards (BFS)

- Walker is at w. Where to go next?



$$\mathrm{w} \rightarrow \boxed{\begin{matrix} s_1 \\ s_2 \\ s_3 \end{matrix}} \quad \boxed{\begin{matrix} 1/p \\ 1 \\ 1/q \end{matrix}}$$

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

where:

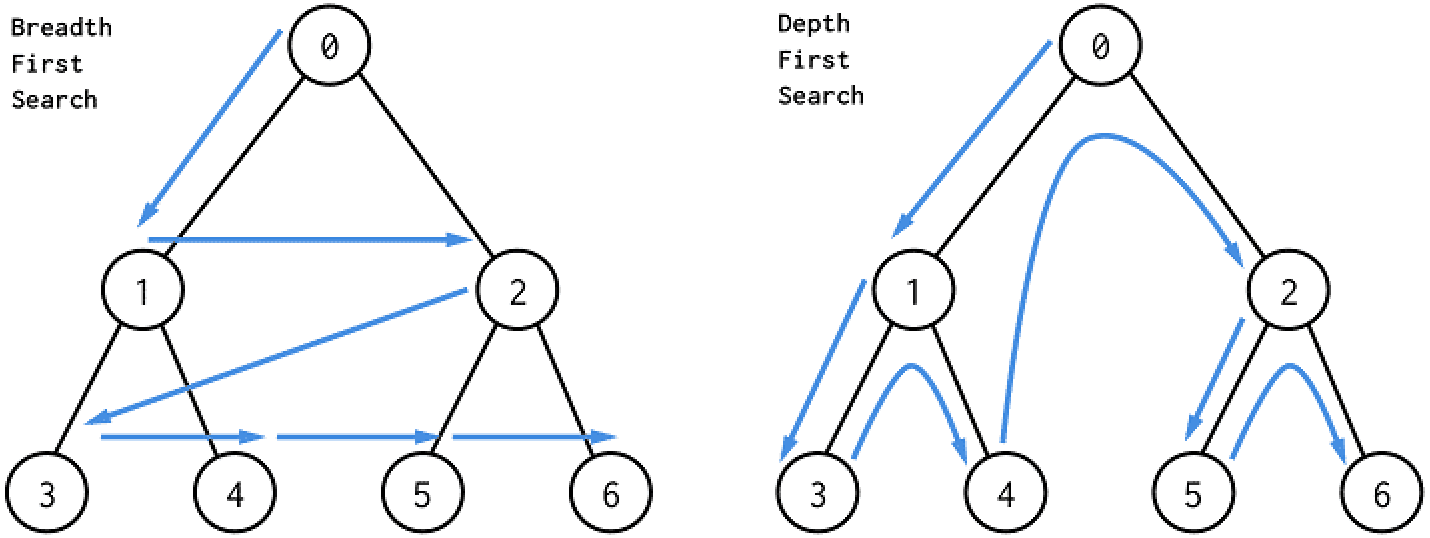
p,q: model transition probabilities

p: return parameter

q: "walk away" parameter

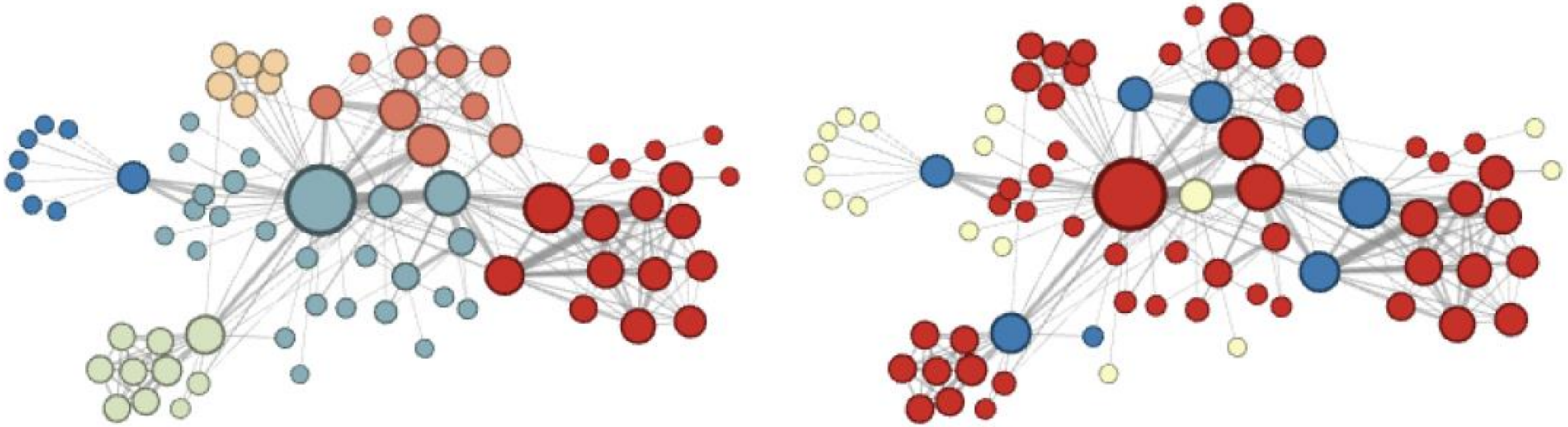
BFS vs. DFS

- BFS: Micro-view of neighbourhood
- DFS: Macro-view of neighbourhood



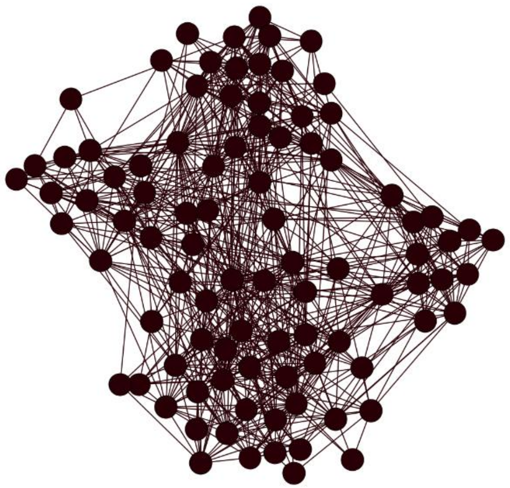
BFS vs. DFS

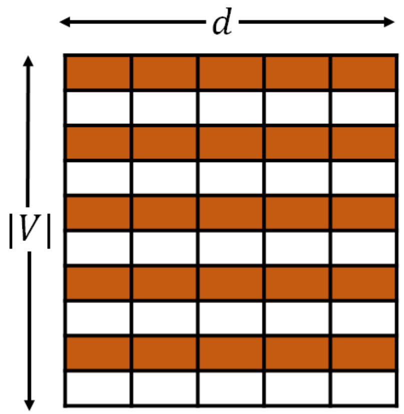
- BFS: Micro-view of neighbourhood
- DFS: Macro-view of neighbourhood



- Graph embedding:

Input Network **Node Embeddings** **Data Mining Tasks**





- Classification
- Community Detection
- Link Prediction
- Anomaly Detection
- Sense Making
- …

Many Possible Applications!

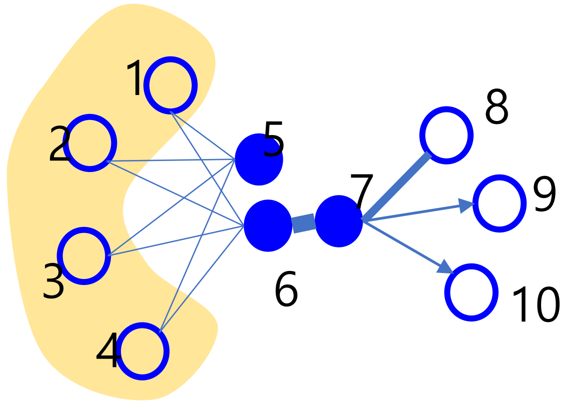
LINE model

- Has a clear objective function
  - Preserve the first-order and second-order proximity between the vertices
- Very scalable
  - Effective and efficient optimization algorithm through asynchronous stochastic gradient descent
  - Only take a couple of hours to embed network with millions of nodes, billions of edges on a single machine

Tang et al. LINE: Large-scale Information Network Embedding. WWW’15

First-order Proximity

- The local pairwise proximity between the vertices
  - Determined by the observed links
- However, many links between the vertices are missing
  - Not sufficient for preserving the entire network structure

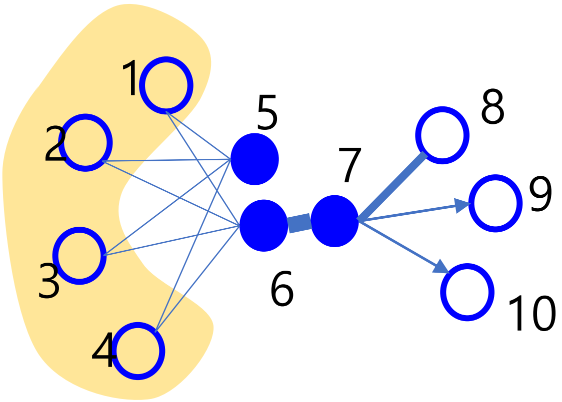


Vertex **6** and **7** have a large first-order proximity

Second-order Proximity

- The proximity between the neighbourhood structures of the vertices
- Mathematically, the second-order proximity between each pair of vertices (u,v) is determined by:

$$\hat{p}_u = (w_{u1}, w_{u2}, \dots, w_{u|V|})$$
$$\hat{p}_v = (w_{v1}, w_{v2}, \dots, w_{v|V|})$$



Vertex **5** and **6** have a large second-order proximity

$$\hat{p}_5 = (1,1,1,1,0,0,0,0,0,0)$$

$$\hat{p}_6 = (1,1,1,1,0,0,1,0,0,0)$$

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:

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

KL-divergence

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

Preserving the Second-order Proximity

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

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

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

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

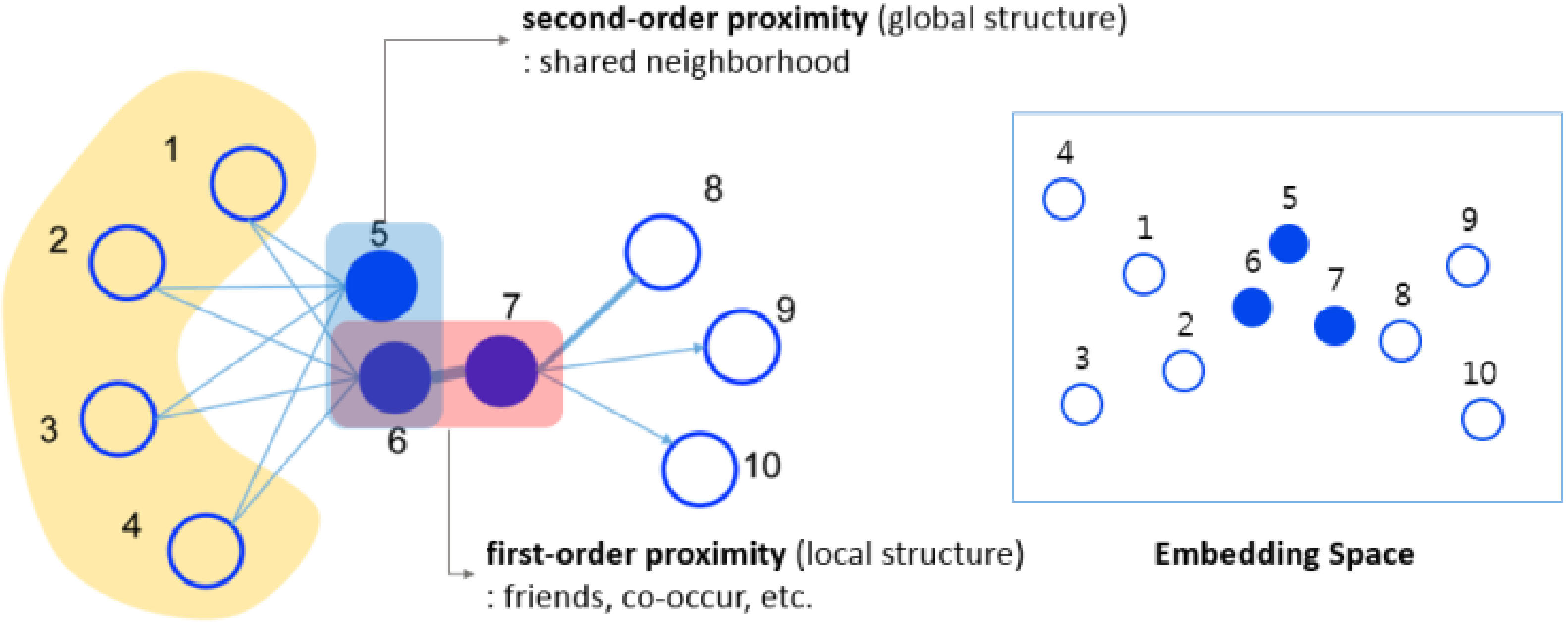
- Objective:

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

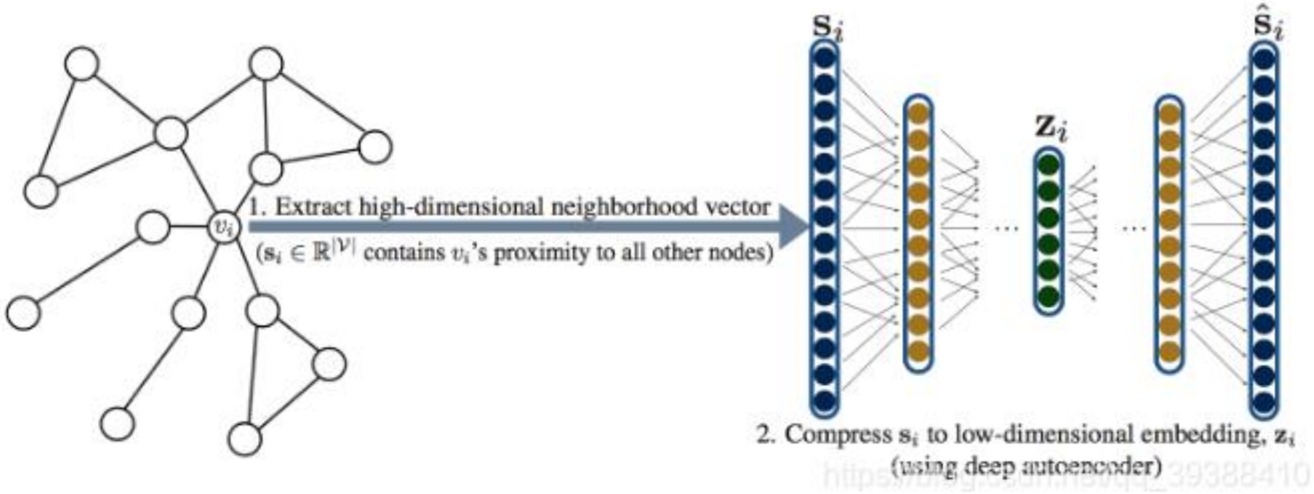
$$\propto -\sum_{(i,j) \in E} w_{ij} \log p_2(v_j|v_i)$$

λ_i: Prestige of vertex in the network $\lambda_i = \sum_j w_{ij}$

Preserve both local & global network structure

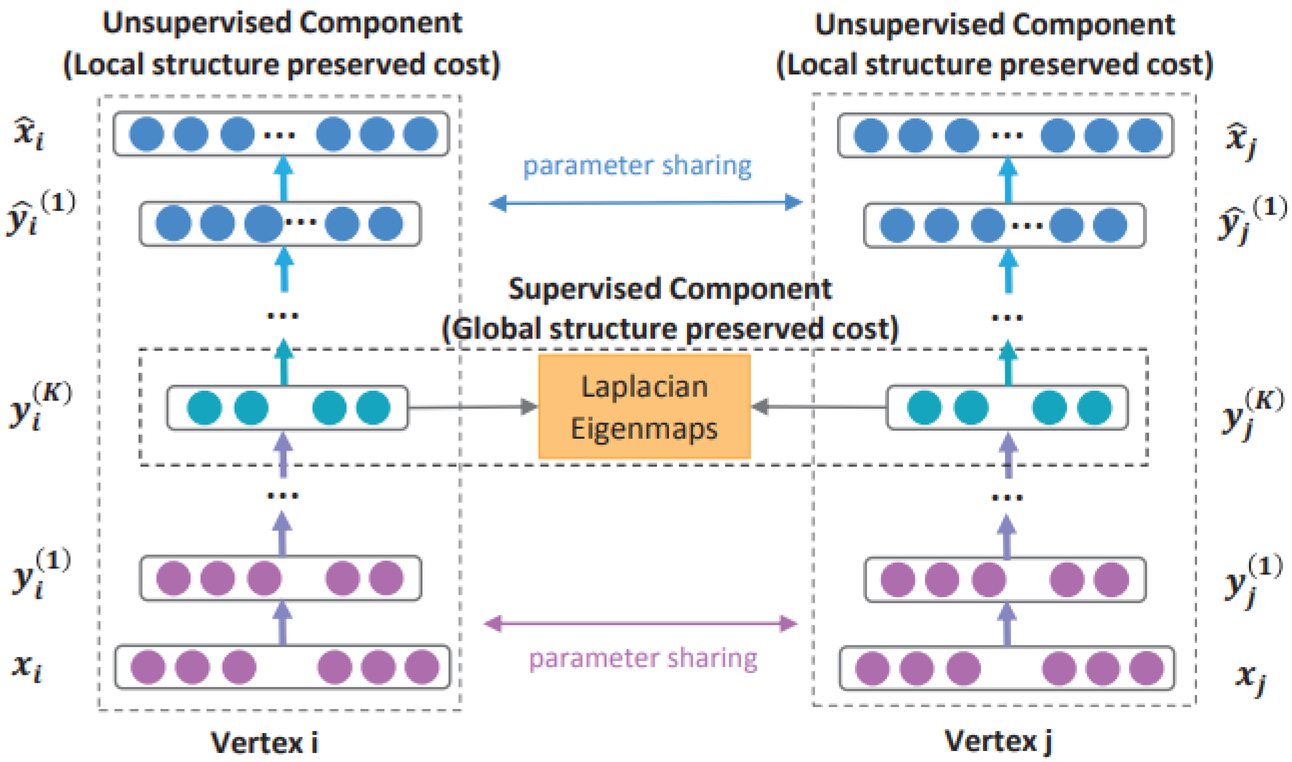


- Structural Deep Network Embedding based on Autoencoder



https://blog.csdn.net/qq_39388410/article/details/103895874

- The framework of the semi-supervised deep model of SDNE

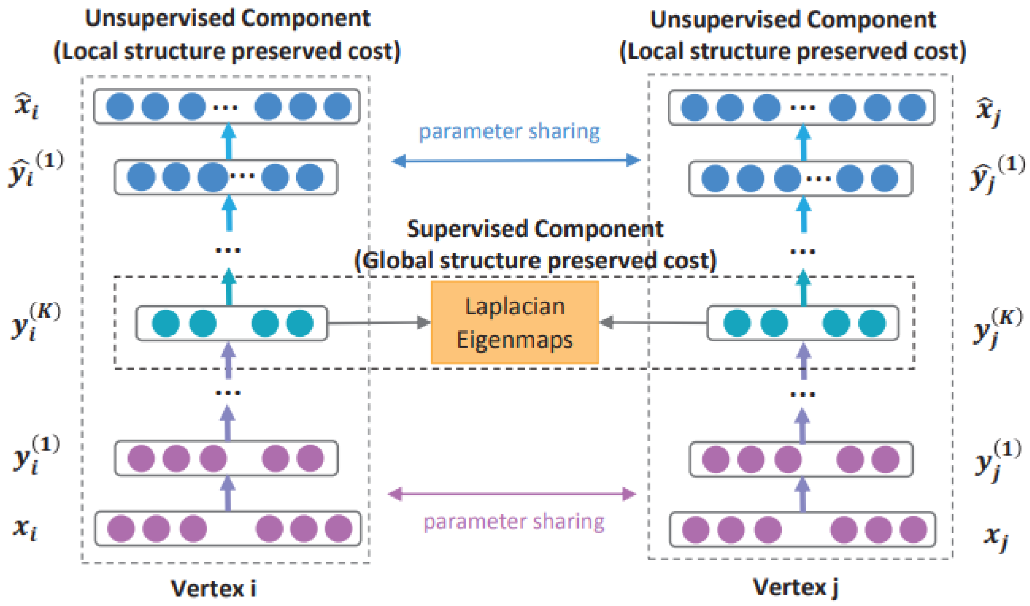


- Then given the input x_i , the hidden representations for each layer are:

$$\mathbf{y}_i^{(1)} = \sigma(W^{(1)}\mathbf{x}_i + \mathbf{b}^{(1)})$$
$$\mathbf{y}_i^{(k)} = \sigma(W^{(k)}\mathbf{y}_i^{(k-1)} + \mathbf{b}^{(k)}), k = 2, ..., K$$

- The goal of the autoencoder is to minimize the reconstruction error of the output and the input.
- The loss function:

$$\mathcal{L} = \sum_{i=1}^{n} \|\hat{\mathbf{x}}_i - \mathbf{x}_i\|_2^2$$

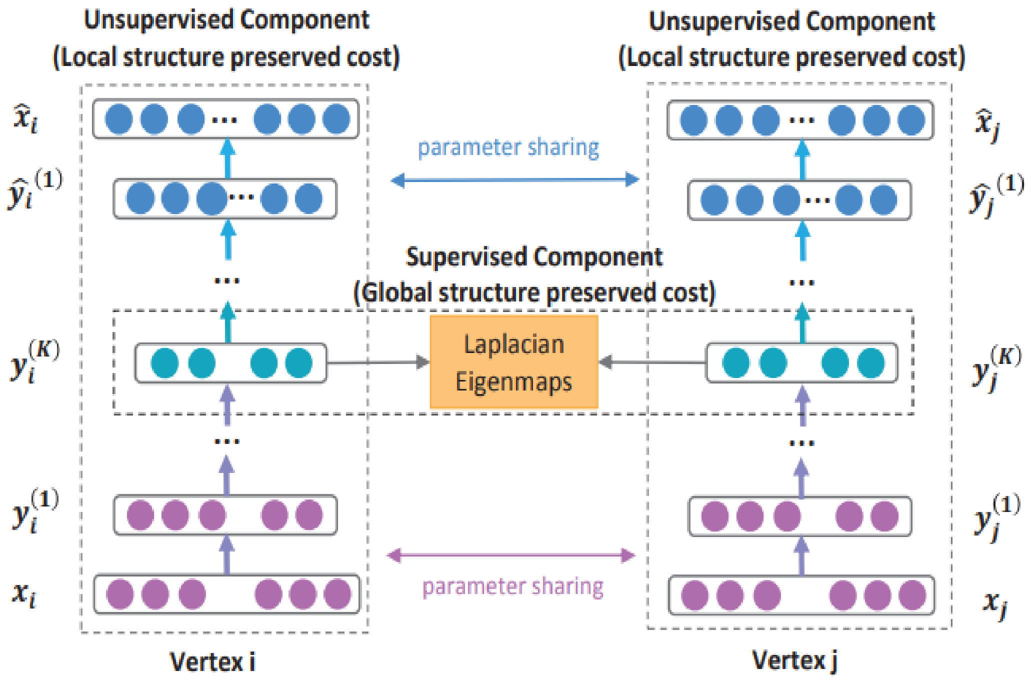


➢ Loss function for first-order proximity:

$$\mathcal{L}_{1st} = \sum_{i,j=1}^{n} s_{i,j} \|\mathbf{y}_i^{(K)} - \mathbf{y}_j^{(K)}\|_2^2 = \sum_{i,j=1}^{n} s_{i,j} \|\mathbf{y}_i - \mathbf{y}_j\|_2^2$$

➢ Impose more penalty to the reconstruction error of the non-zero elements than that of zero elements:

$$\mathcal{L}_{2nd} = \sum_{i=1}^{n} \|(\hat{\mathbf{x}}_i - \mathbf{x}_i) \odot \mathbf{b_i}\|_2^2 = \|(\hat{X} - X) \odot B\|_F^2$$



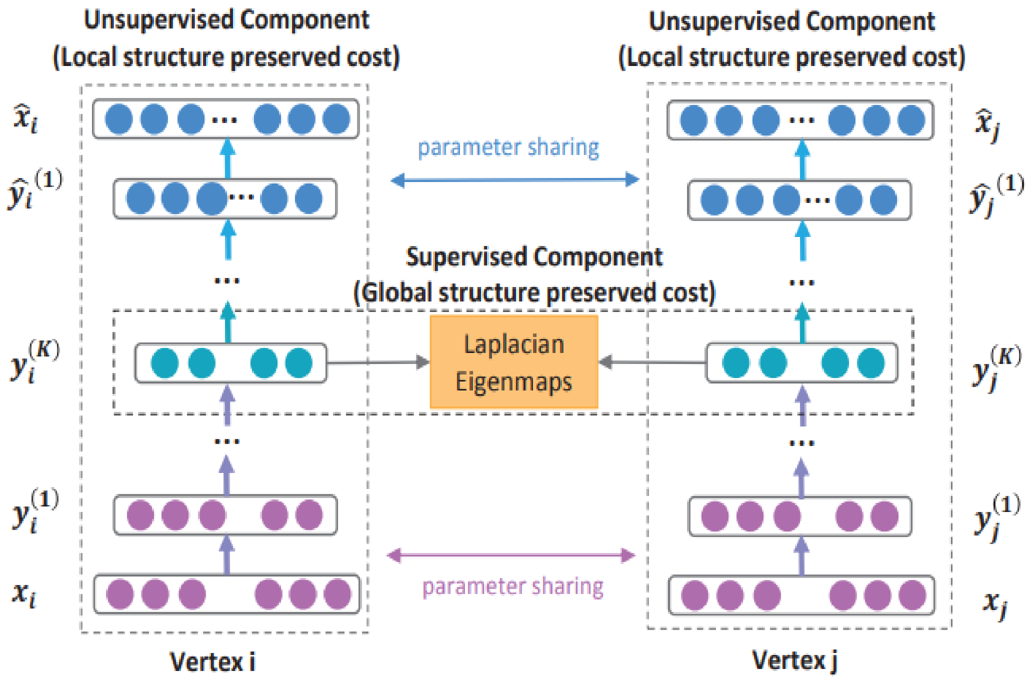
SDNE model

- To preserve the first-order and second-order proximity simultaneously, we need to minimize the joint loss:

$$\mathcal{L}_{mix} = \mathcal{L}_{2nd} + \alpha\mathcal{L}_{1st} + \nu\mathcal{L}_{reg}$$
$$= \|(\hat{X} - X) \odot B\|_F^2 + \alpha \sum_{i,j=1}^{n} s_{i,j}\|\mathbf{y}_i - \mathbf{y}_j\|_2^2 + \nu\mathcal{L}_{reg}$$

- where Lreg is an L2-norm regularizer term to prevent overfitting, which is defined as follows:

$$\mathcal{L}_{reg} = \frac{1}{2}\sum_{k=1}^{K}(\|W^{(k)}\|_F^2 + \|\hat{W}^{(k)}\|_F^2)$$



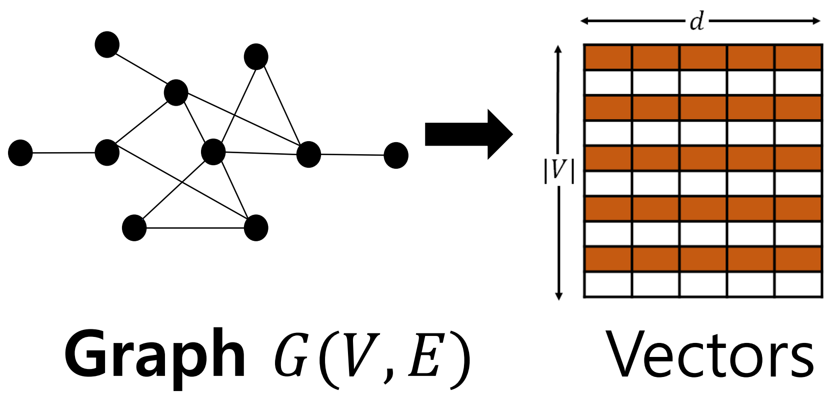
- Popular previous works include

DeepWalk[Perozzi+, KDD2014]

Node2vec[Grover+, KDD 2016]

SDNE[Wang+, KDD 2016]

LINE[Tang+,WWW 2015]



Limited just to the node embeddings

Subgraph2Vec model: Idea

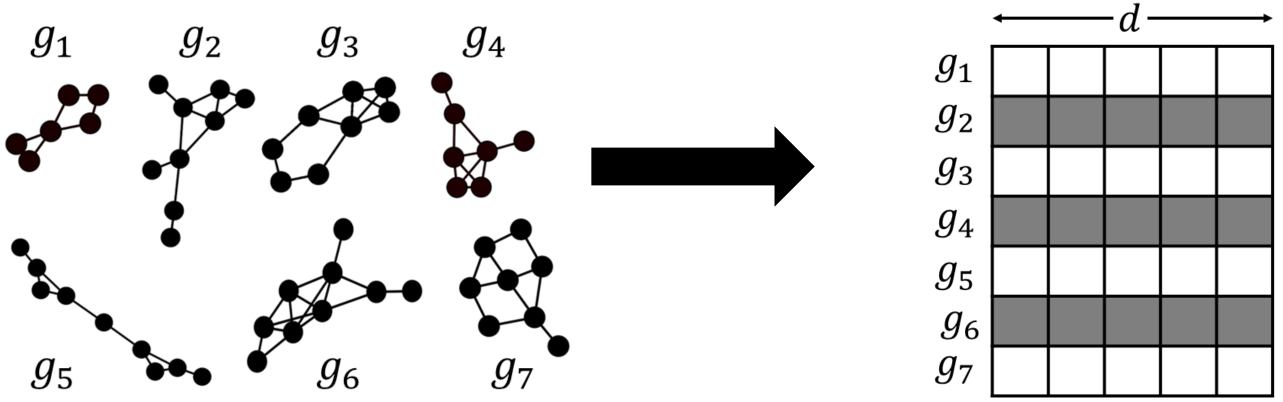
- Learning representation of substructures
  - Extend the WL relabeling strategy to define a proper context for a given subgraph.
  - A modification to the skipgram model enabling it to capture varying length radial contexts

Problem Formulation: Setting

- Given
  - A set S={g_1, g_2, …, g_n } of subgraphs
  - Typically for the same graph
  - An integer d
- Learn
  - d-dimensional embedding for each subgraph
  - Such that pre-defined subgraph property is preserved

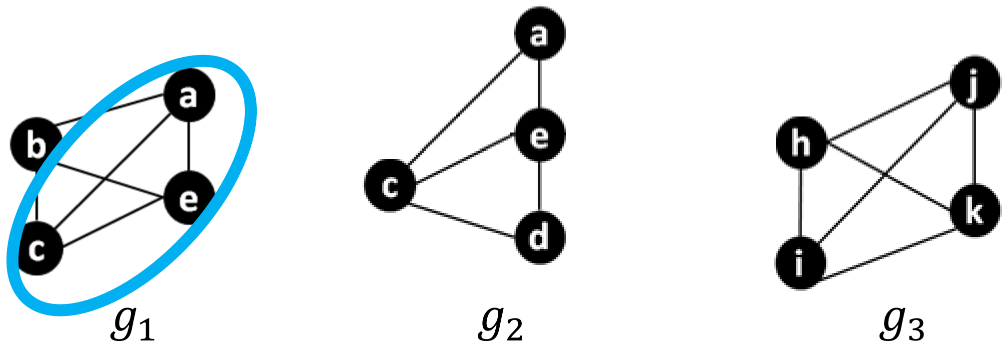
Set of Subgraphs

Subgraph Embedding



Problem formulation: Challenges

- What subgraph property to preserve?
  - Neighbourhood Property:
    - Captures neighbourhood information within the subgraph



  - Subgraph g_1 and g_2 share neighbourhood
  - Subgraph g_3 does not

Algorithm 1: Generate rooted subgraphs

- Generate rooted subgraphs around every node in a given graph
- Considers all the rooted subgraphs (up to a certain degree) of neighbours of r as the context of target subgraphs

Algorithm 2: GETWLSUBGRAPH (v, G, d)

input : v: Node which is the root of the subgraph
$G = (V, E, \lambda)$: Graph from which subgraph has to be extracted
d: Degree of neighbours to be considered for extracting subgraph

output: $sg_v^{(d)}$: rooted subgraph of degree d around node v

begin
  $sg_v^{(d)} = \{\}$
  if $d = 0$ **then**
    $sg_v^{(d)} := \lambda(v)$
  else
    $\mathcal{N}_v := \{v' \mid (v, v') \in E\}$
    $M_v^{(d)} := \{\text{GETWLSUBGRAPH}(v', G, d-1) \mid v' \in \mathcal{N}_v\}$
    $sg_v^{(d)} := sg_v^{(d)} \cup \text{GETWLSUBGRAPH}$ $(v, G, d-1) \oplus sort(M_v^{(d)})$
  return $sg_v^{(d)}$

Algorithm 2: Learn embeddings of those subgraphs

- The skipgram model maximizes co-occurrence probability among the sub-graphs that appear within a given context window.

Algorithm 3: $\textsc{RadialSkipGram}\ (\Phi, sg_v^{(d)}, G, D)$

begin
- $context_v^{(d)} = \{\}$
- **for** $v' \in \textsc{Neighbours}(G, v)$ **do**
  - **for** $\partial \in \{d-1,\ d,\ d+1\}$ **do**
    - **if** $(\partial \geq 0$ *and* $\partial \leq D)$ **then**
      - $context_v^{(d)} = context_v^{(d)} \cup \textsc{GetWLSubgraph}(v', G, \partial)$
- **for** *each* $sg_{cont} \in context_v^{(d)}$ **do**
  - $J(\Phi) = -\log \Pr\left(sg_{cont} | \Phi(sg_v^{(d)})\right)$
  - $\Phi = \Phi - \alpha \frac{\partial J}{\partial \Phi}$

네트워크 과학 연구실
NETWORK SCIENCE LAB

가톨릭대학교
THE CATHOLIC UNIVERSITY OF KOREA