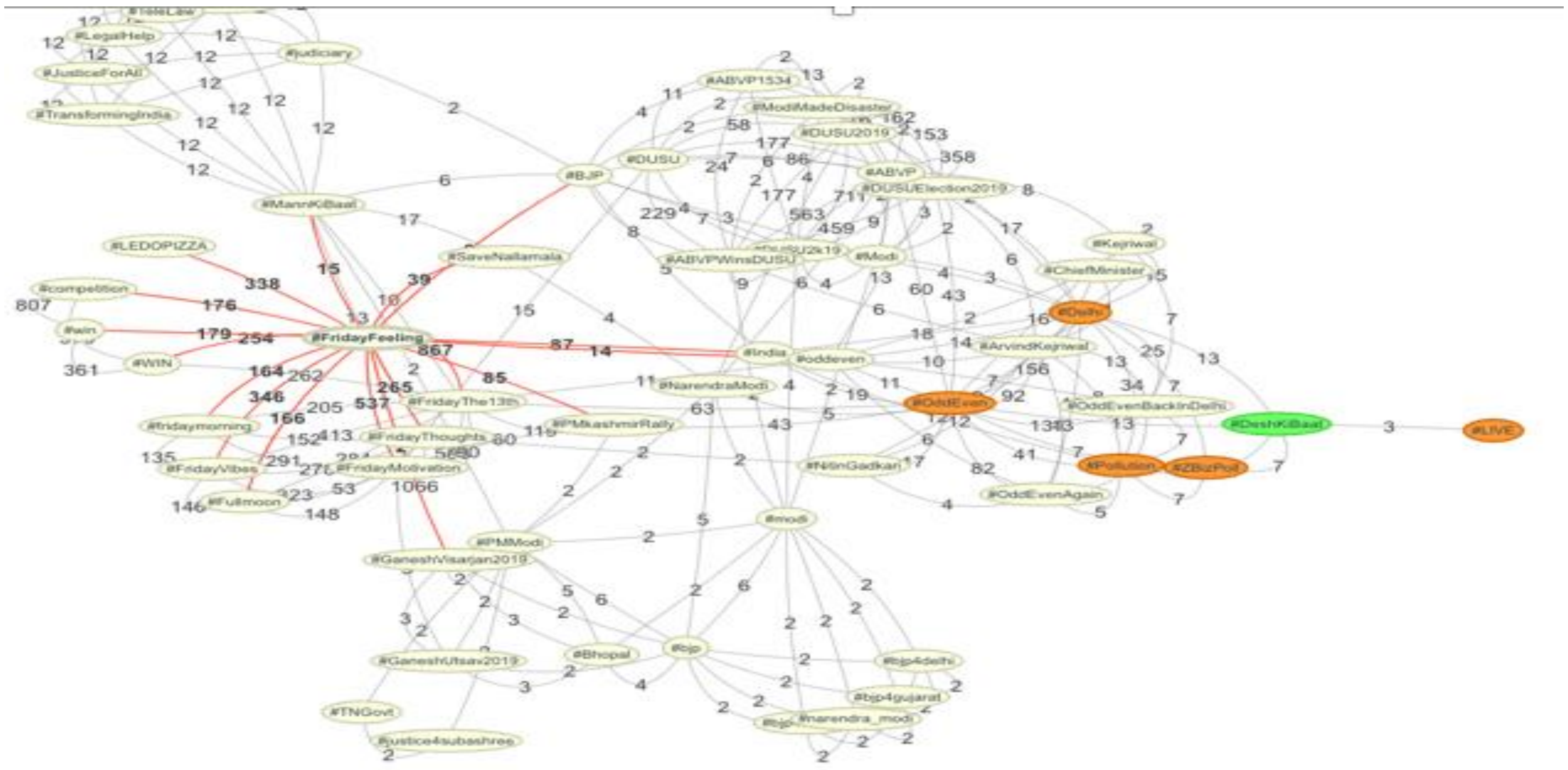
Network Embedding

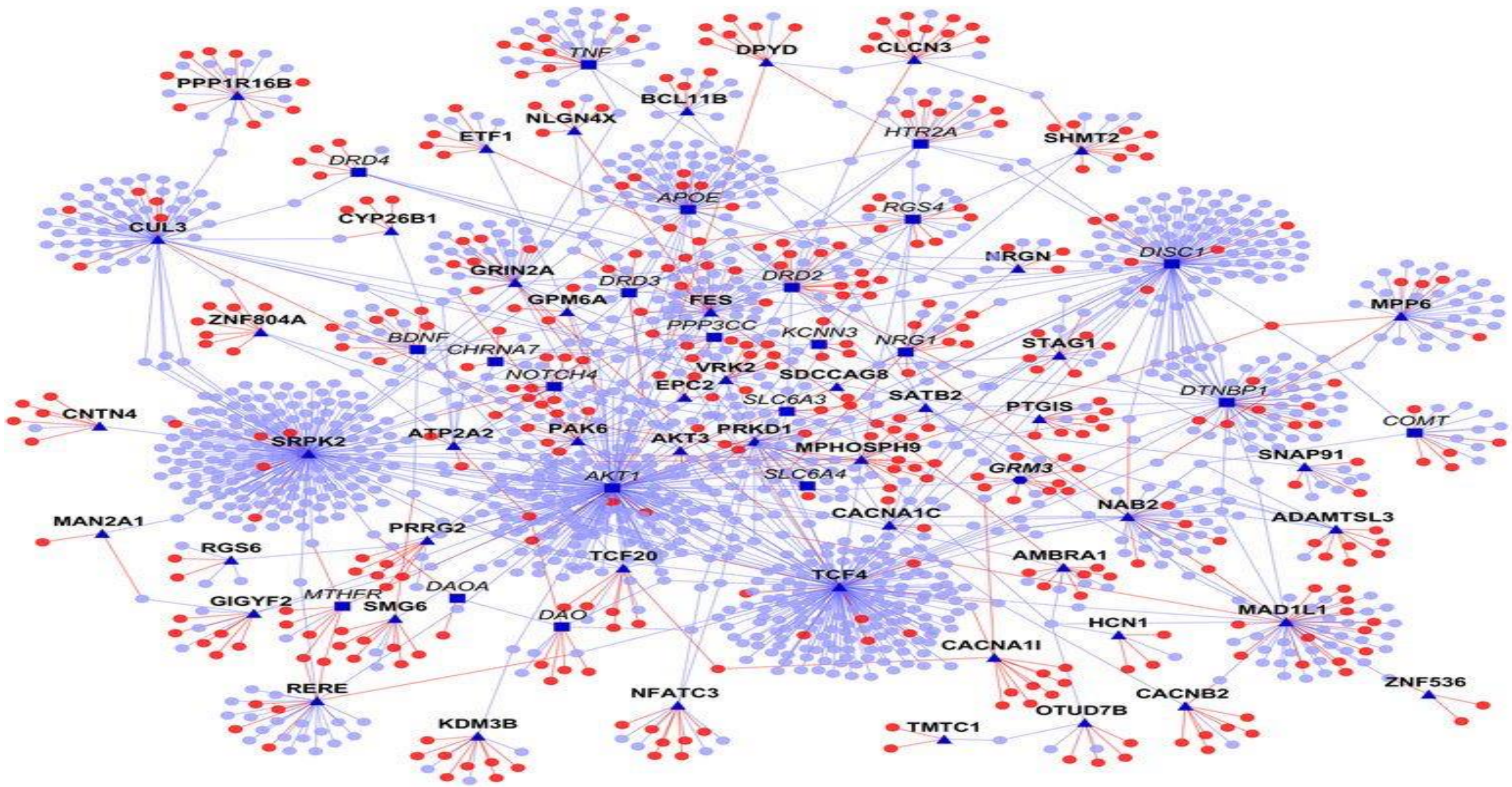
Facebook Friendship Network

Twitter Followers’ Network

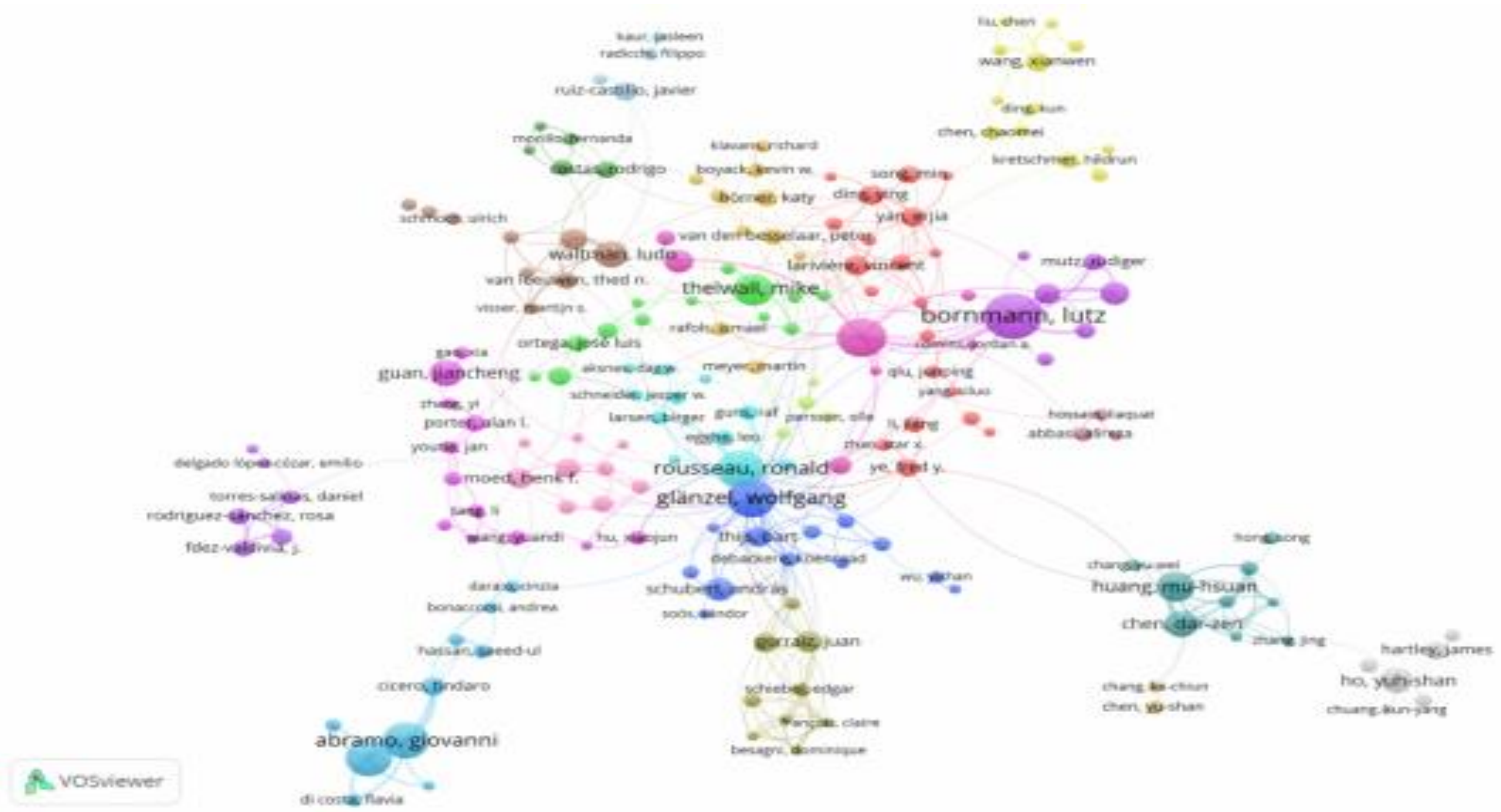
Hashtag Co-occurence Network



Protein-Protein Interaction Network

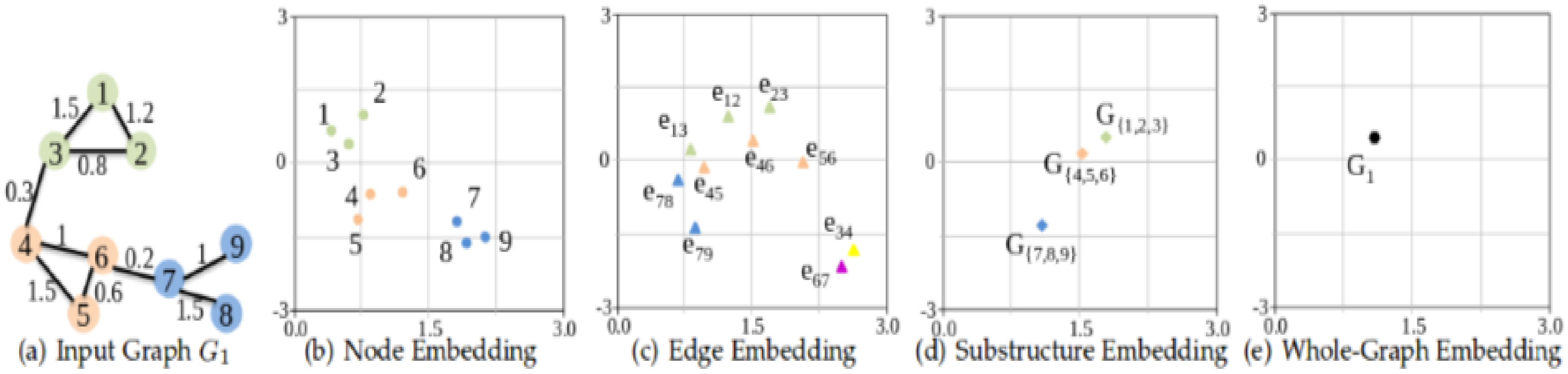


Co-Authorship Network



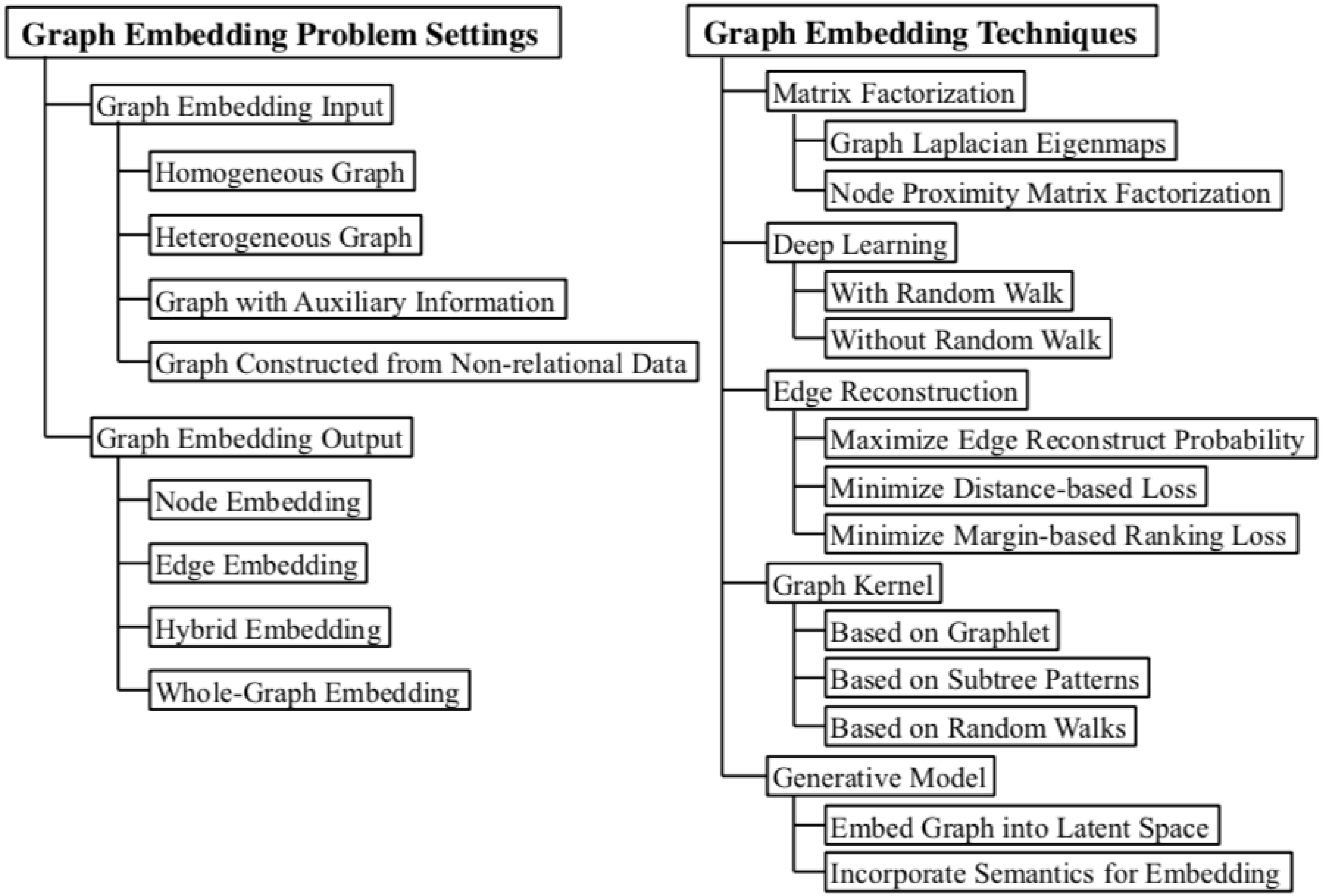
What is Network Embedding?

Suppose *G*(*V*,*E*) represents a network then Network Embedding refers to generating low dimensional network features corresponding to Nodes, Edges, Substructures, and the Whole-Graph.



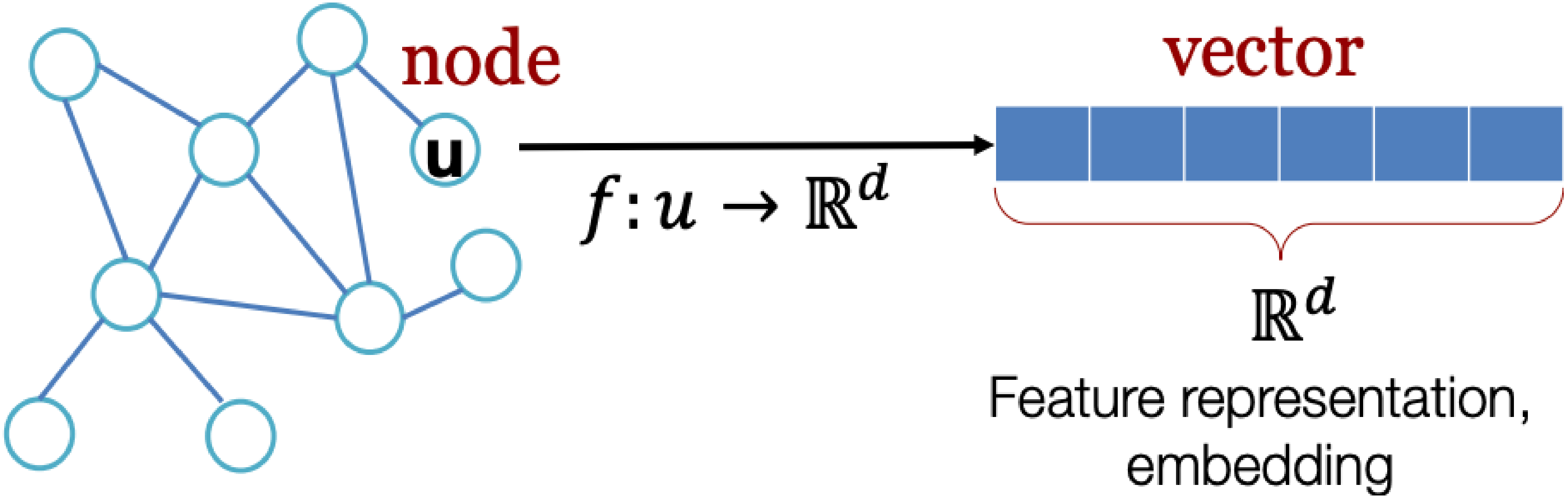
HongYun Cai, Vincent W. Zheng, Kevin Chen-Chuan Chang: **A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications.** IEEE Trans. Knowl. Data Eng. 30(9): 1616-1637 (2018)

Network Embedding - Texonomy



HongYun Cai, Vincent W. Zheng, Kevin Chen-Chuan Chang:**A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications.**IEEE Trans. Knowl. Data Eng. 30(9): 1616-1637 (2018)

Node Embedding

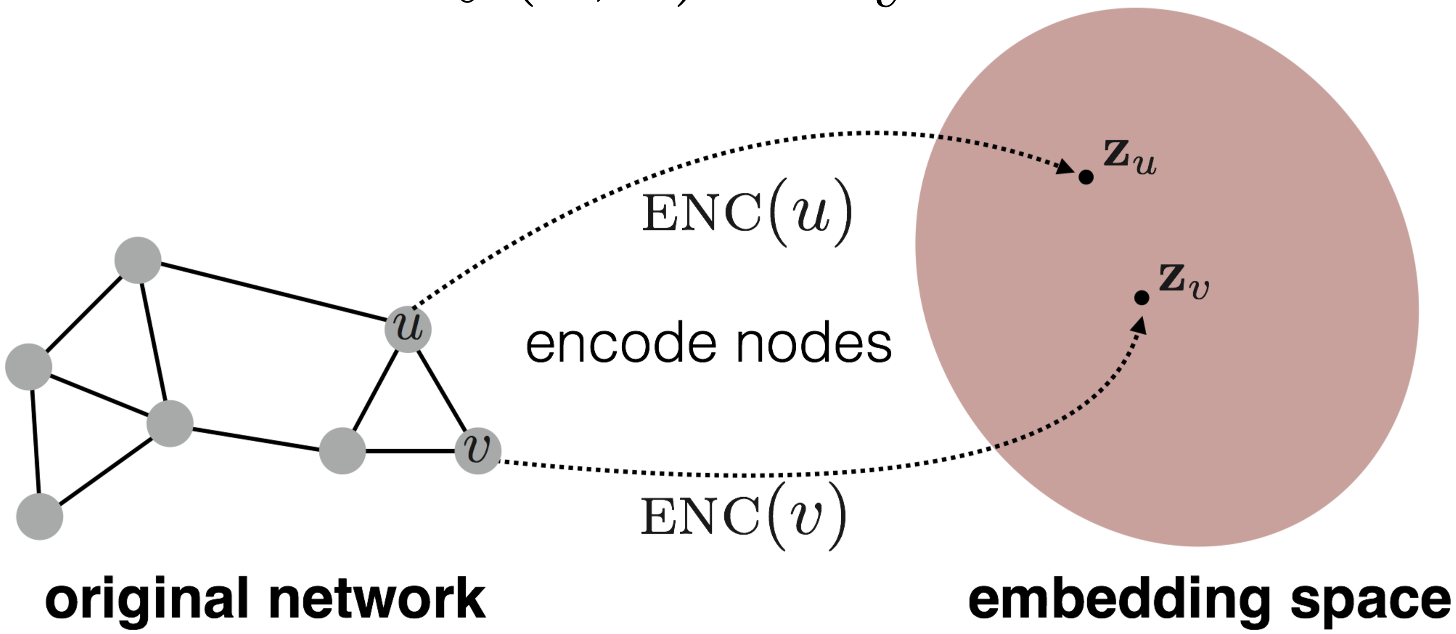


Find embedding of nodes to d-dimensions so that “similar” nodes in the graph have embeddings that are close together.

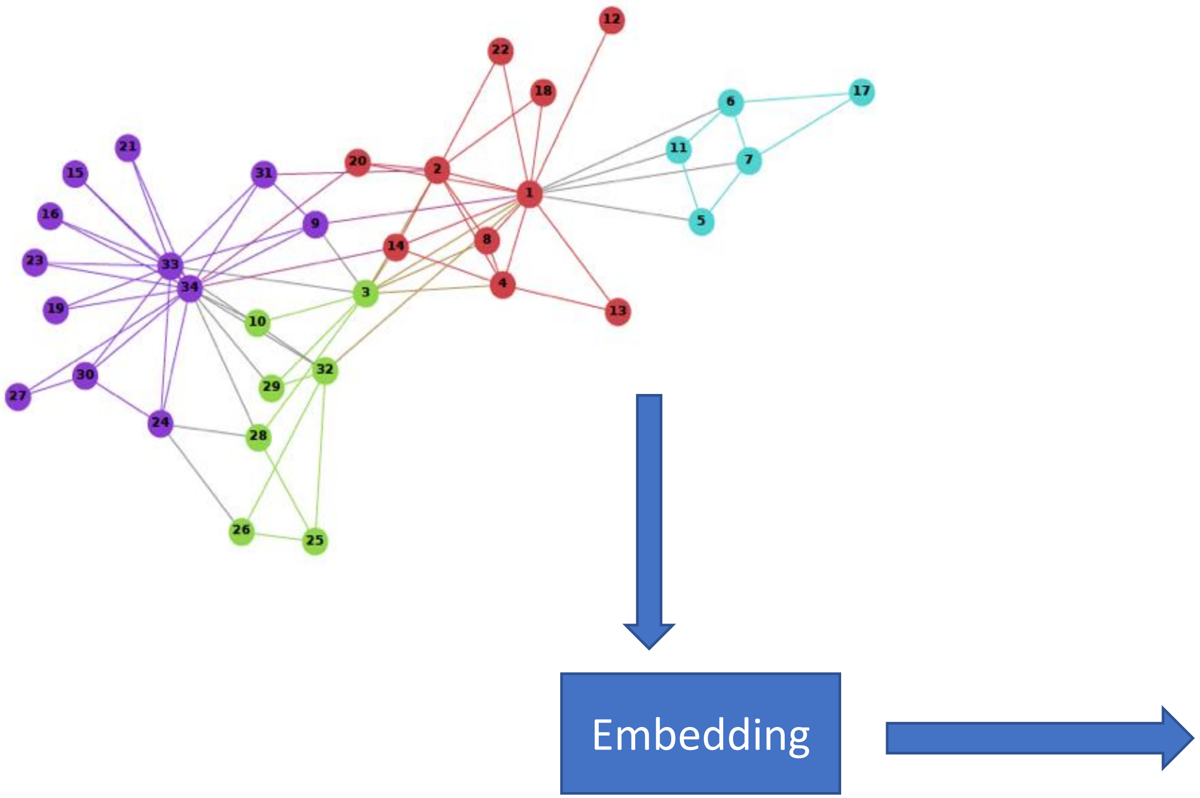
Node Embedding

Goal is to learn an encoding matrix so that **similarity of nodes in the embedding space** approximates **similarity in the original network**.

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

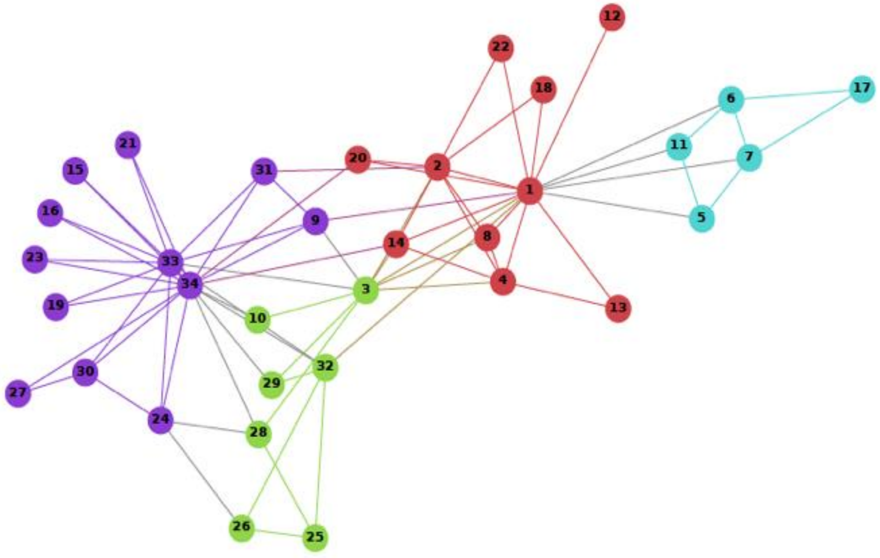


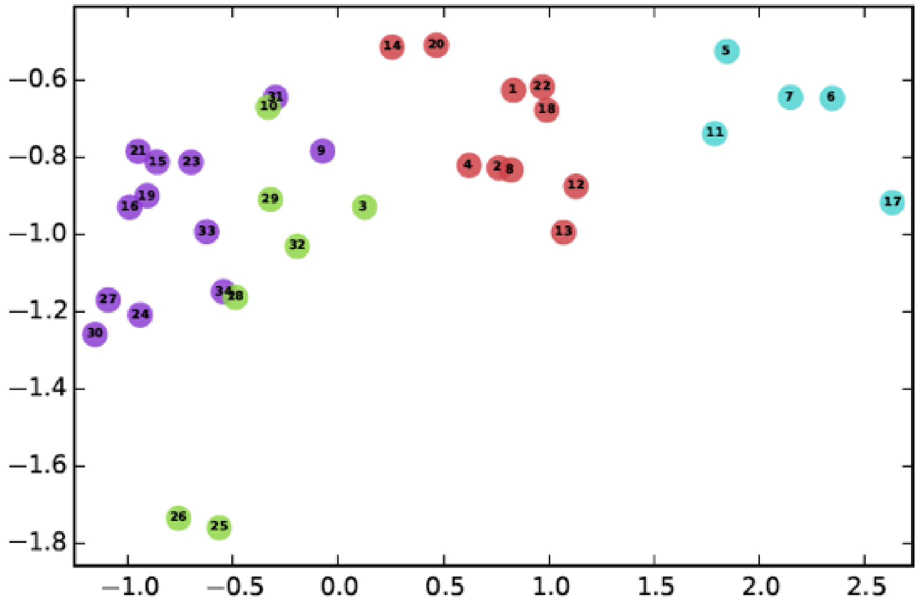
Node Embedding




```
0  :  [1.98479516 2.104902  ]
1  :  [1.37277632 1.17311989]
2  :  [0.6859528  1.08576755]
3  :  [1.27303975 0.41918123]
4  :  [ 0.65722556 -0.36297315]
5  :  [ 0.71494817 -0.39592245]
6  :  [ 0.71494817 -0.39592245]
7  :  [ 1.04116841 -0.05324659]
8  :  [-0.27647113 -0.24436752]
9  :  [ 0.65722556 -0.36297315]
10  :  [ 0.37478198 -0.46299671]
11  :  [ 0.63250419 -0.30641371]
12  :  [ 0.66706122 -0.24109217]
13  :  [ 0.64664541 -0.38866438]
14  :  [ 0.27253823 -0.57650996]
15  :  [ 0.64664541 -0.38866438]
16  :  [-0.5209427  -0.58291863]
17  :  [-0.48328386 -0.17530251]
18  :  [-0.24844813 -0.38526576]
19  :  [-0.5175929  -0.23180349]
20  :  [-0.34294104 -0.09404636]
21  :  [-1.56457792  1.8504155 ]
22  :  [ 0.28271458 -0.28822466]
23  :  [-1.76282103  3.02650113]
24  :  [-0.70912996 -0.57803913]
25  :  [-0.70912996 -0.57803913]
26  :  [-0.70912996 -0.57803913]
27  :  [-0.70912996 -0.57803913]
28  :  [-0.70912996 -0.57803913]
29  :  [-1.08578976 -0.27525378]
30  :  [-0.37477941  0.06842669]
31  :  [-1.03412988 -0.27920574]
32  :  [-0.28044074  0.04244915]
33  :  [-0.58710262 -0.38879992]
```

Node Embedding





Nodes closer in the original network should also be closer in the embedding space

Many ways of Learning Z

1. Matrix Factorization
2. Neural Network
   1. DeepWalk
   2. Node2Vec
   3. GNN
   4. GCN

Matrix Similarity based

Let us assume that A is a matrix representing the network (say adjacency matrix)

Learn the matrix Z so that the loss function is minimum.

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

Matrix Similarity based

Let us assume that A is a matrix representing the network (say adjacency matrix)

Learn the matrix Z so that the loss function is minimum.

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

- Stochastic gradient descent (SGD)
- Low Rank Matrix Factorization (PCA, LSI)

Matrix Similarity based

Let us assume that A is a matrix representing the network (say adjacency matrix)

Learn the matrix Z so that the loss function is minimum.

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

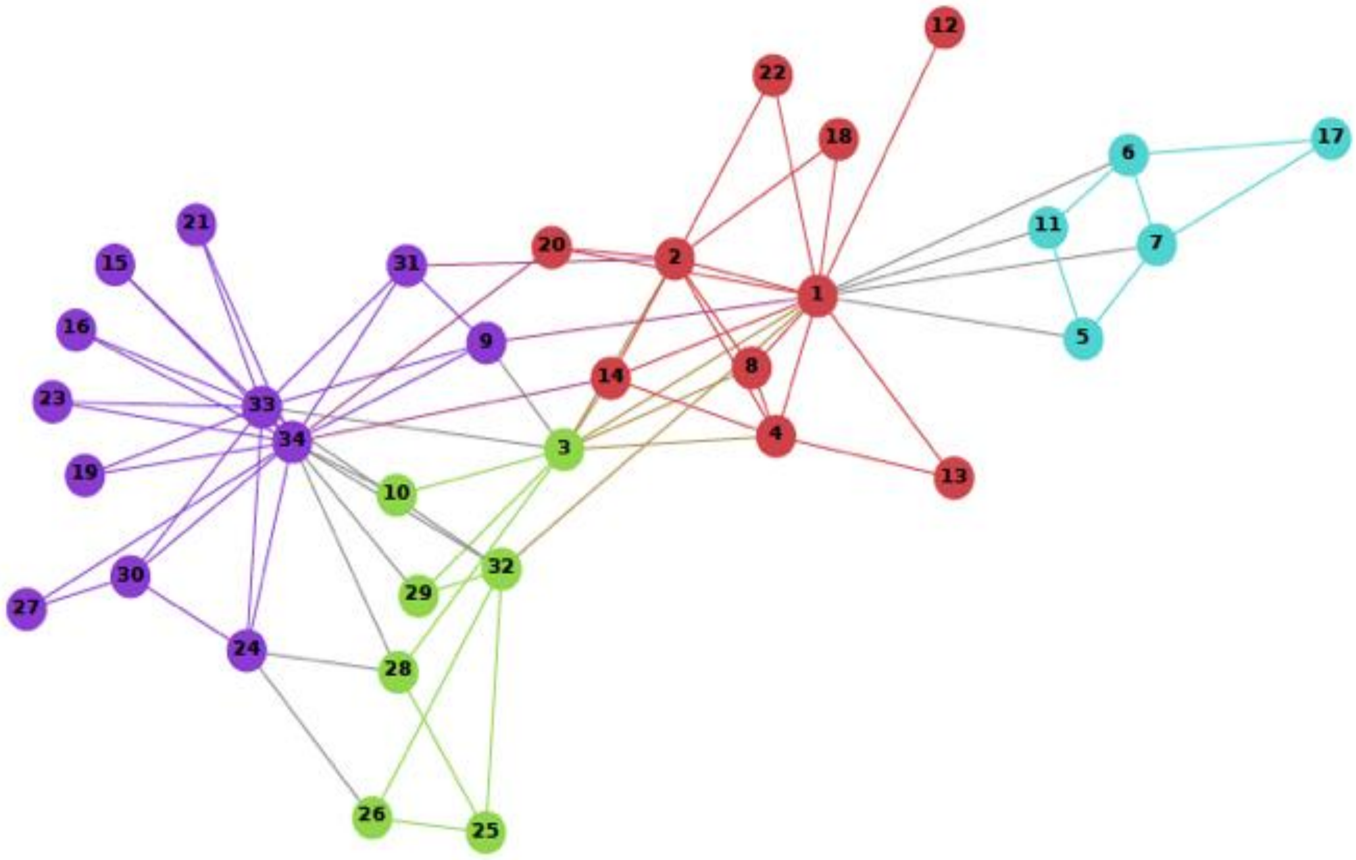
Any derived matrix

- Stochastic gradient descent (SGD)
- Low Rank Matrix Factorization

Random Walk

- **DeepWalk:** Just run fixed-length, unbiased random walks starting from each node
- **Node2Vec:** Use flexible, biased random walks that can trade off between **local** and **global** views of the network.

DeepWalk – unbiased RW



Generate RW node sequence
17, 6, 11, 1, 13
17, 6, 7 , 5, 1
1, 2, 20, 34, 10
…..

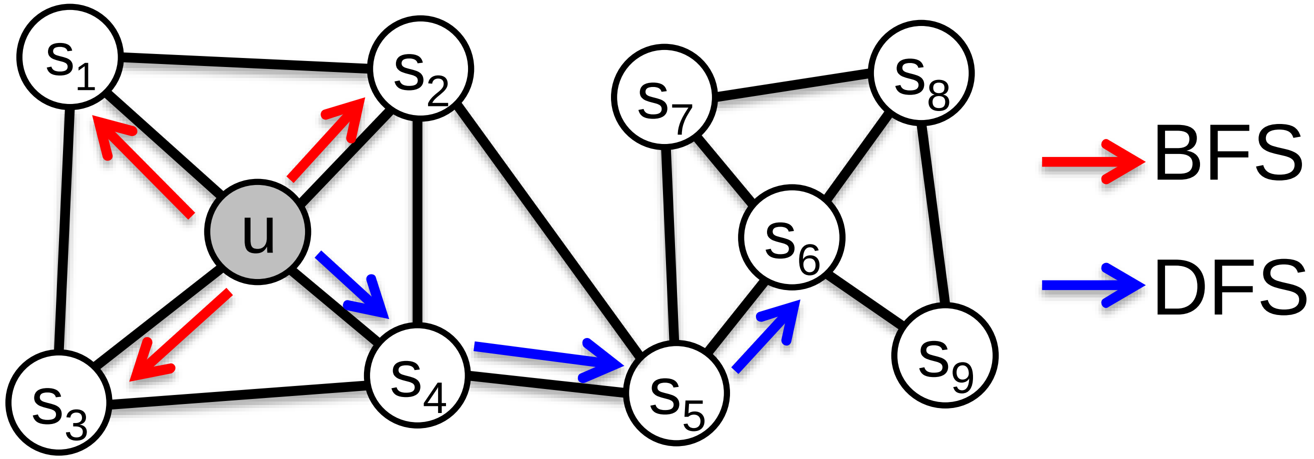
Apply skip-gram to generate the embedding

Node2vec - Biased RW

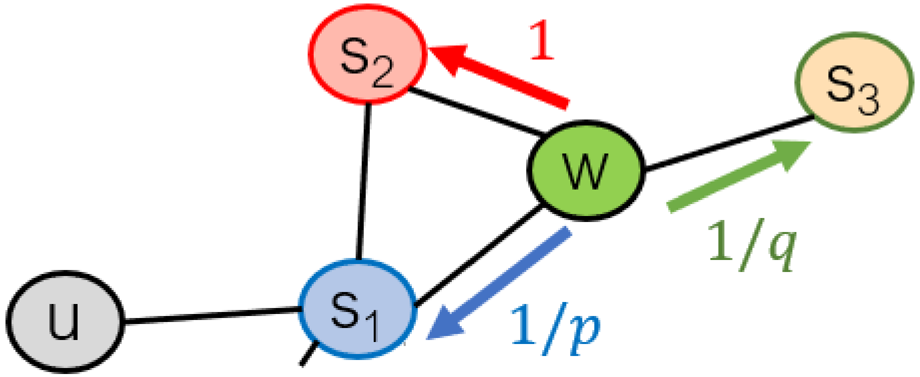
Interpolating BFS and DFS

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

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



Node2vec: two parameters

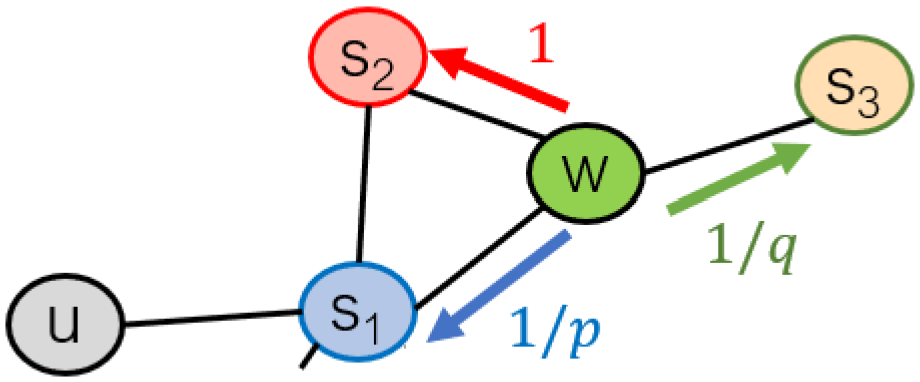


Generate RW node sequence
17, 6, 11, 1, 13
17, 6,7 , 5, 1
1,2, 20,34, 10
.....

Apply skip-gram to generate the embedding

- p, q model transition probabilities
  - p … return parameter
  - q … ”walk away” parameter

Node2vec: two parameters



17, 6, 11, 1, 13
17, 6,7 , 5, 1
1,2, 20,34, 10
.....

Apply skip-gram to generate the embedding

- p, q model transition probabilities
  - p … return parameter
  - q … "walk away" parameter