struc2vec

Learning Node Representations from Structural Identity

Siddharth Ravi, 27th Aug 2020

About Me

ML Research Engineer at Q-Free, Netherlands (~1.5 yrs - ongoing)

Researcher at Tilburg University (~1 yr)

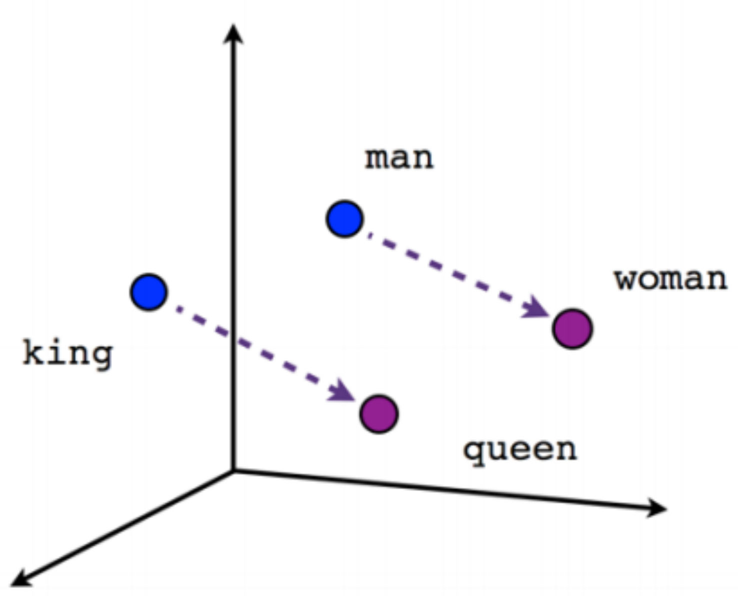
MSc. in Systems and Control (cognitive robotics track), TU Delft.

Summary Slide

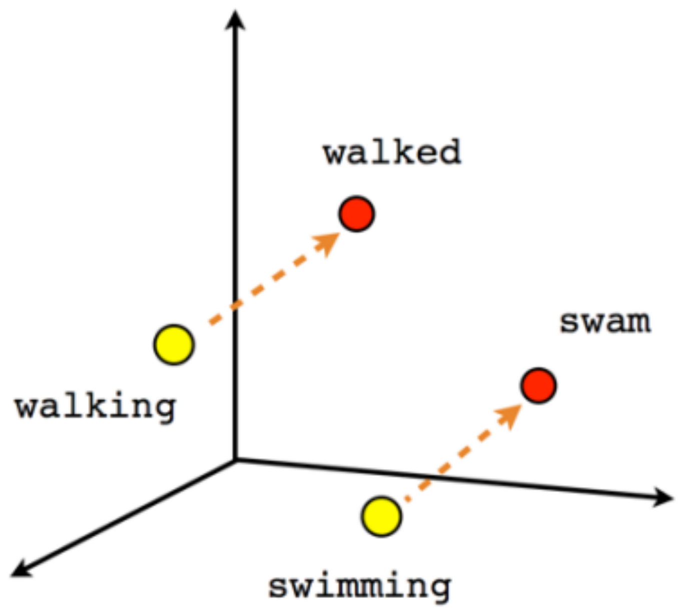
- struc2vec is an embedding that captures the *structural identity* of nodes.
- Structural identity is a concept of symmetry in which network nodes are identified based on their network structure and relationship to other neighbouring nodes.
- Nodes are placed in latent space according to a *structural similarity* measure
- Obtained using a 4-stage training process.

Embeddings

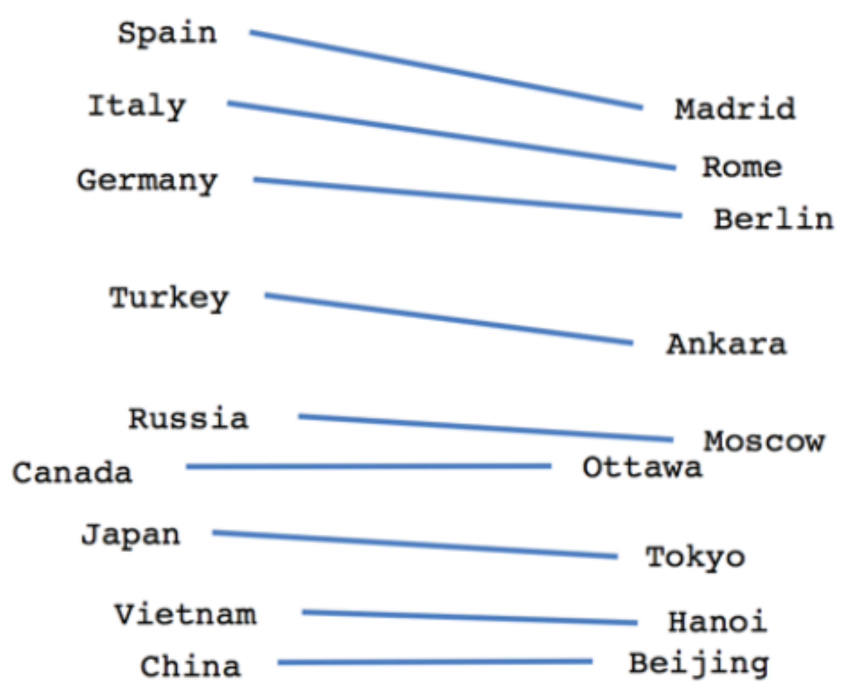
Embeddings. Why?



Male-Female



Verb tense

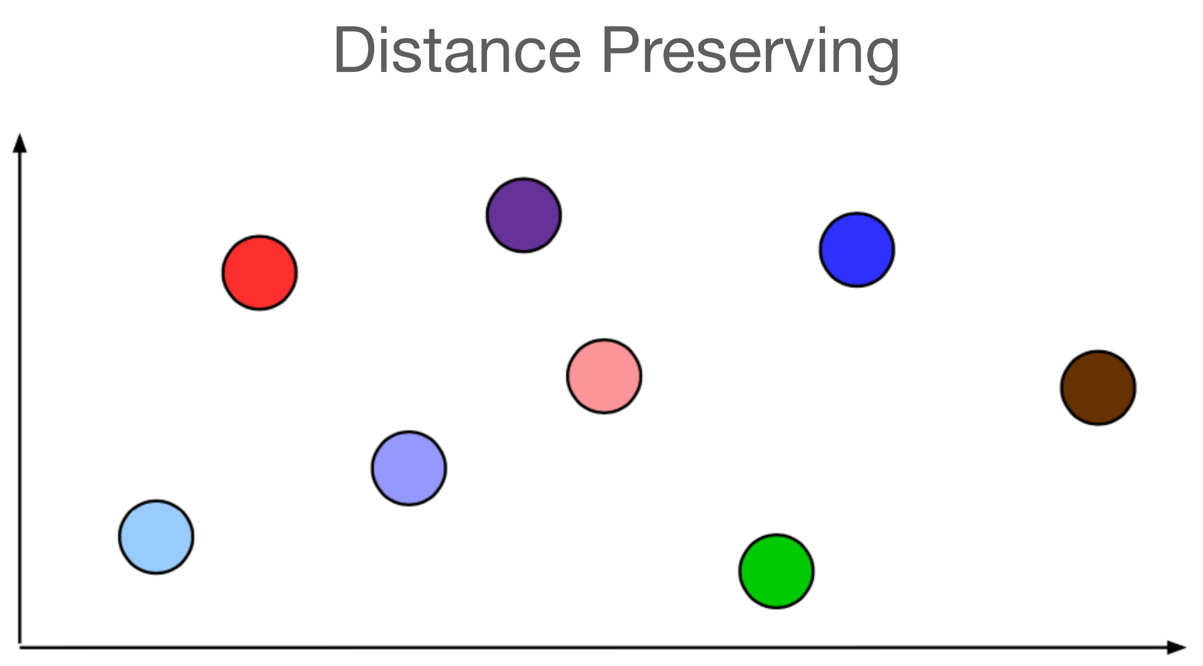


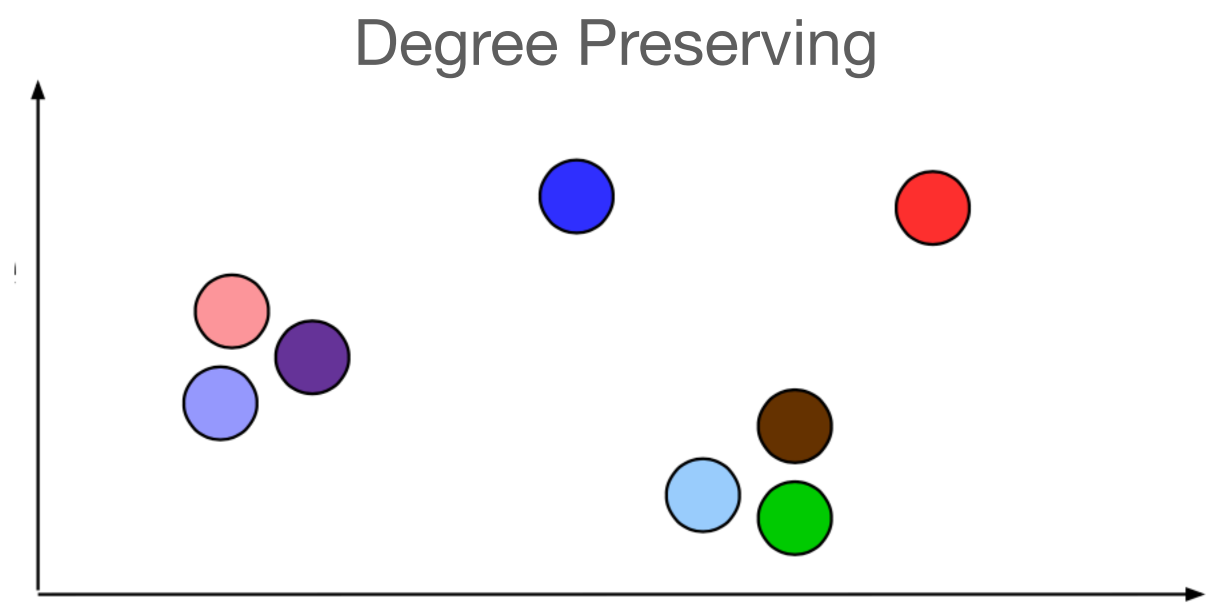
Country-Capital

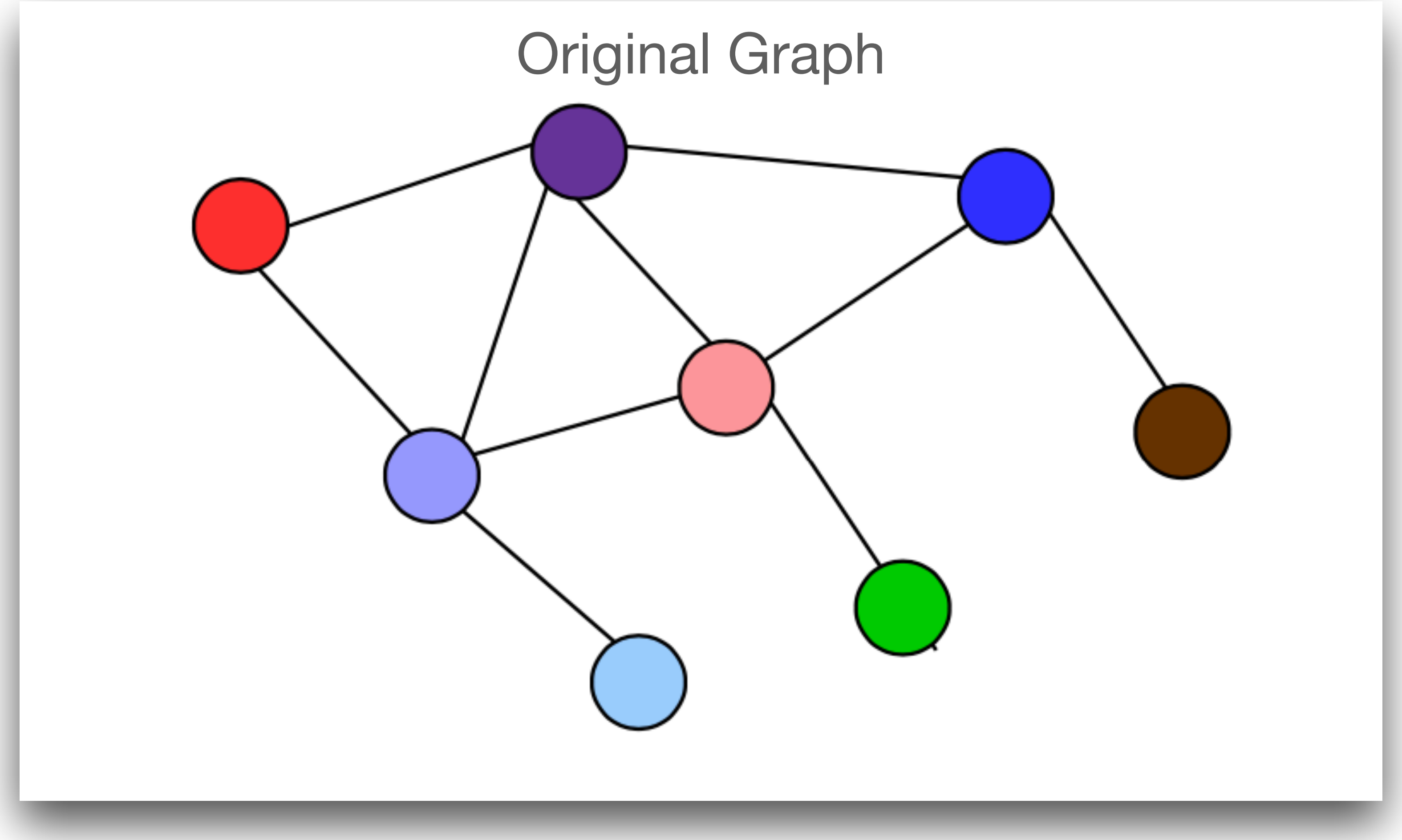
- Puts similar concepts closer in a dimensionally compact representational space. Eg - word2vec creates vectors that clump semantically similar words together.
- Improved results while used as input vectors for models.
- Generally allows for some level of first and second degree relational interpretability. Eg - analogies for word vectors.

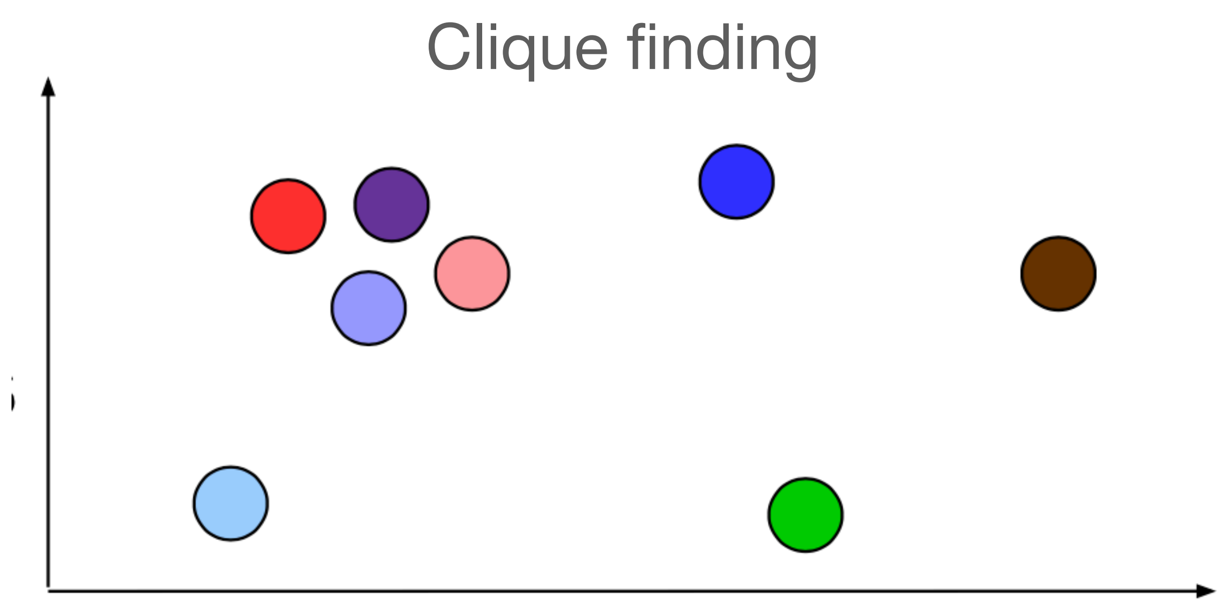
Embeddings for Graphs

- Converts graph data into a low dimensional space which preserves important graph properties.
- Many different ways to approach the task, no one embedding to rule them all.









Source :https://leoribeiro.github.io/struc2vec-slides.pdf

Choosing an embedding

- Choosing the right embedding depends on the application.
- struc2vec is an embedding that preserves the concept of structural identity.

Structural Identity

Structural Identity

- Network nodes usually have specific roles, and are identified on the basis of their structural position (Eg: hierarchy in an organisation).
- In other words, roles provide structural identity to nodes.
- Structural similarity of two nodes is a property that is a function of the neighbourhood of nodes.
- Degree sequences of increasing neighbourhood sizes have high structural similarity (even if they are far apart at the network level).
- Putting nodes in latent space according to structural similarity would result in an embedding where:
    - Distance between latent representations is strongly correlated to their structural similarity.
    - Other non-structural node/edge attributes (eg: node labels, hop distance) would have little to do with their latent representation.

struc2vec

struc2vec

- Four main steps -

    1. Determine similarities between vertex pairs for different neighbourhood sizes.

    2. Construct a weighted multilayer graph.

    3. Generate context for each node in constructed graph using biased random walks through the multilayer graph.

    4. Use techniques like skip-gram to learn latent representations from context.

Calculating Structural Similarity

Let $R_k(u)$ be the set of nodes k hops from node u in a graph.

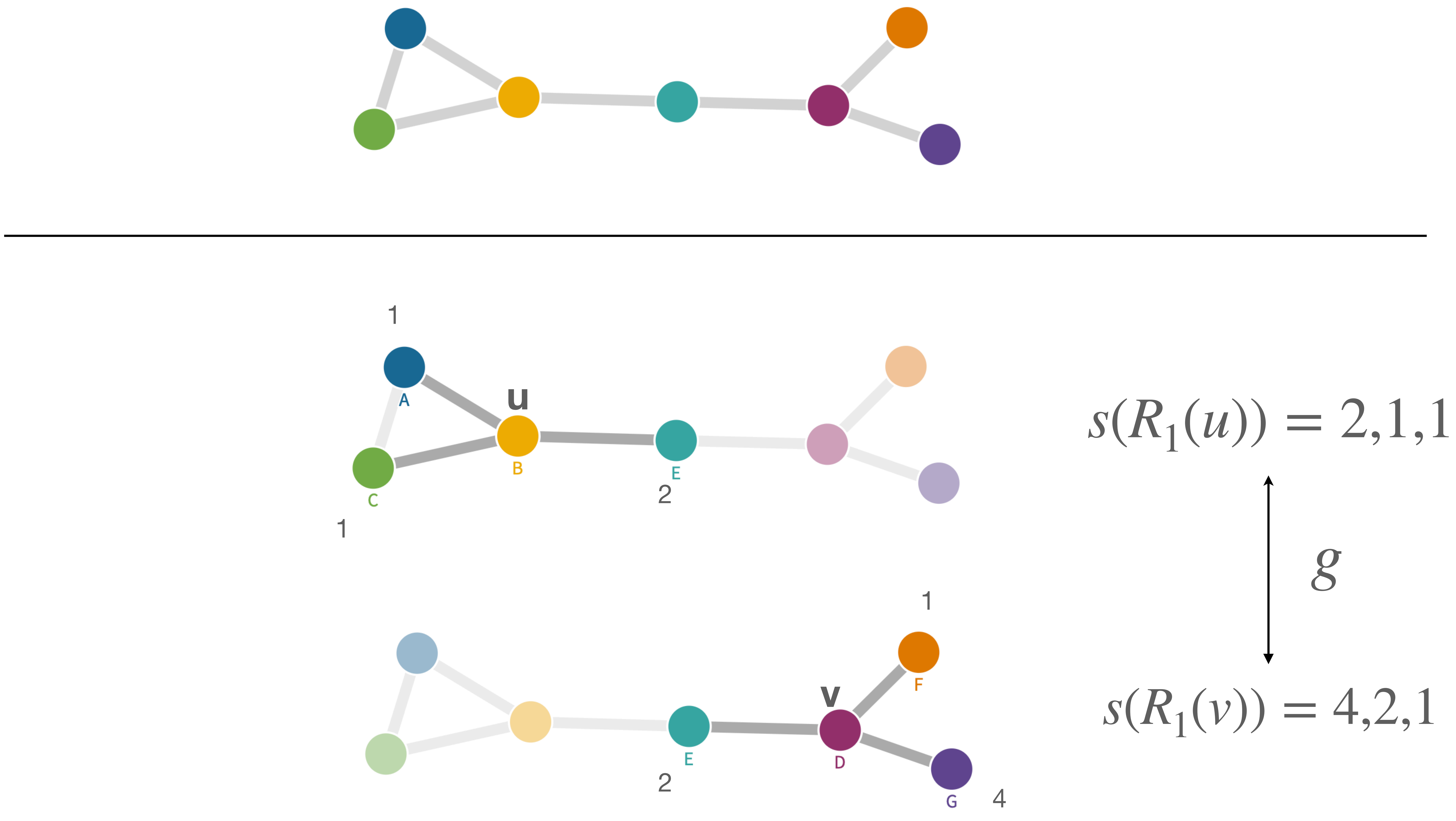
The structural distance between nodes u and v is given by $f_k(u, v)$ at a given distance k.
(only defined when both nodes have atleast one neighbour).

$f_k(u, v)$ has two components-

1. The distance at the $(k-1)$ neighbourhood, $f_{k-1}(u, v)$.

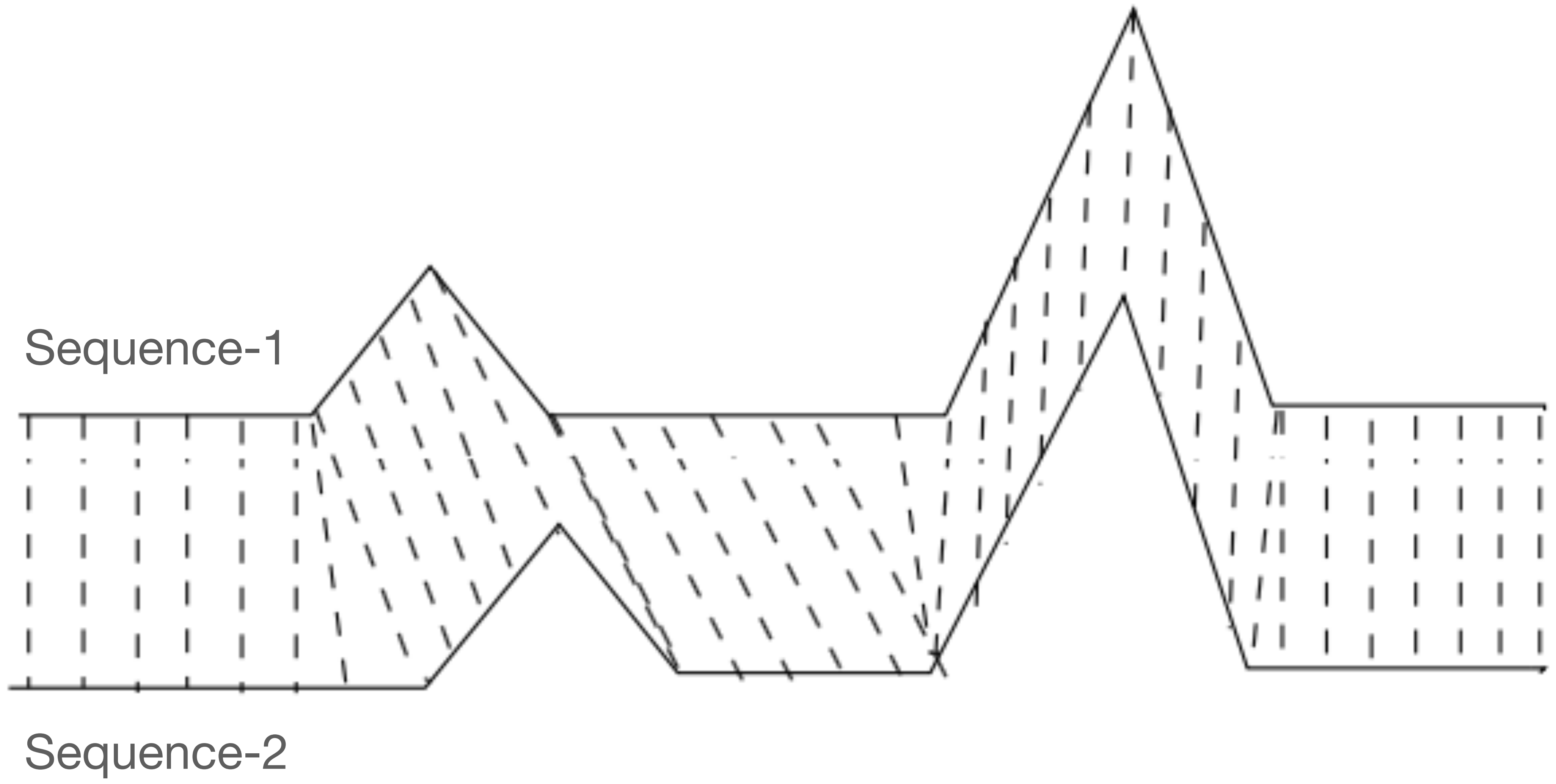
2. The degree sequence distance g between the ordered degree sequence s, of nodes in the k-neighbourhoods of u and v.

$$f_k(u, v) = f_{k-1}(u, v) + g(s(R_k(u), s(R_k(v)))$$



1
u
A
B
E
C
2
1
$s(R_1(u)) = 2,1,1$
g
1
v
F
E
D
2
G
4
$s(R_1(v)) = 4,2,1$

Structural Similarity



- Sequences are treated as a time series, and can be of different length.
- *Dynamic time warping* (DTW) is used to assess their similarity.
- DTW computes a distance function between matched elements in the two sequences.

$$d(a, b) = \frac{max(a, b)}{min(a, b)} - 1$$

Source: https://commons.wikimedia.org/wiki/File:Dynamic_time_warping.png

struc2vec

- Four main steps -

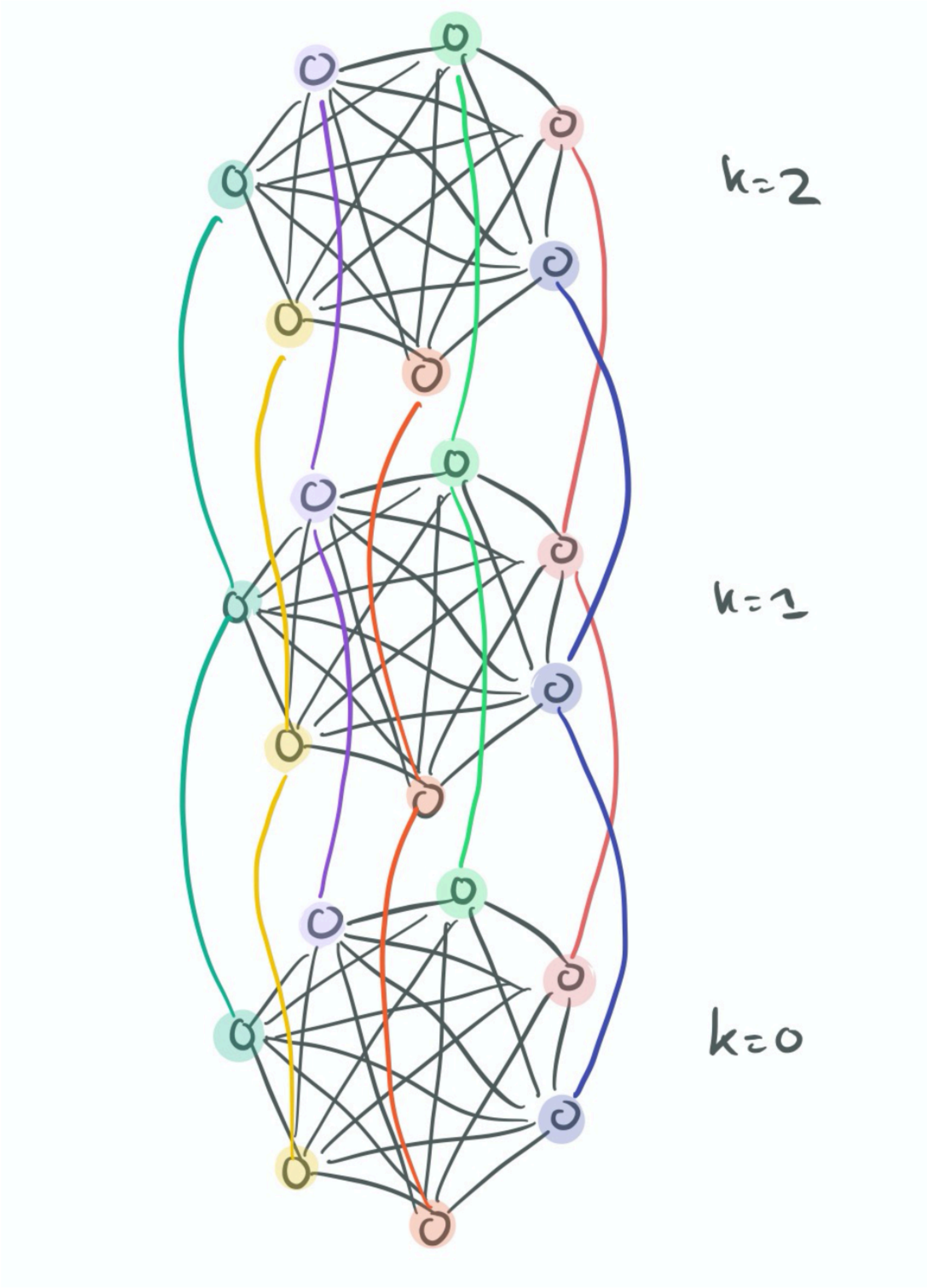
    1. Determine similarities between vertex in pairs for different neighbourhood sizes.

    2. Construct a weighted multilayer graph.

    3. Generate context for each node in constructed graph using biased random walks through the multilayer graph.

    4. Use techniques like skip-gram to learn latent representations from context.

Constructing a Weighted Multilayer Graph



- Each layer in graph is a level in the hierarchy measuring structural similarity.
- Number of layers = diameter of graph k^*.
- Each layer k has one node for each node in original graph.
- Weighted undirected edges are created between nodes in a layer k, with the weights a function of the structural distance in their k-neighbourhood.

$$w_k(u, v) = e^{-f_k(u,v)}$$

  - Edges are defined only when structural similarity is defined.

Source: https://blog.acolyer.org/2017/09/15/struc2vec-learning-node-representations-from-structural-identity

struc2vec

- Four main steps -

  1. Determine similarities between vertex in pairs for different neighbourhood sizes.
  2. Construct a weighted multilayer graph.
  3. Generate context for each node in constructed graph using biased random walks through the multilayer graph.
  4. Use techniques like skip-gram to learn latent representations from context.

Generating Context Through Random Walk

- For node u, context is generated through a number of random walks in the multi layer graph. It is allowed to walk within layer, or move between layers.
- Starting at the corresponding vertex in layer 0, walks are repeated to yield multiple contexts.
- Walk is biased because movement between nodes is dependent on their weights.
- The node sequences generated by the random walks become the context.

Generating Context Through Random Walk

q = probability of staying in the current layer

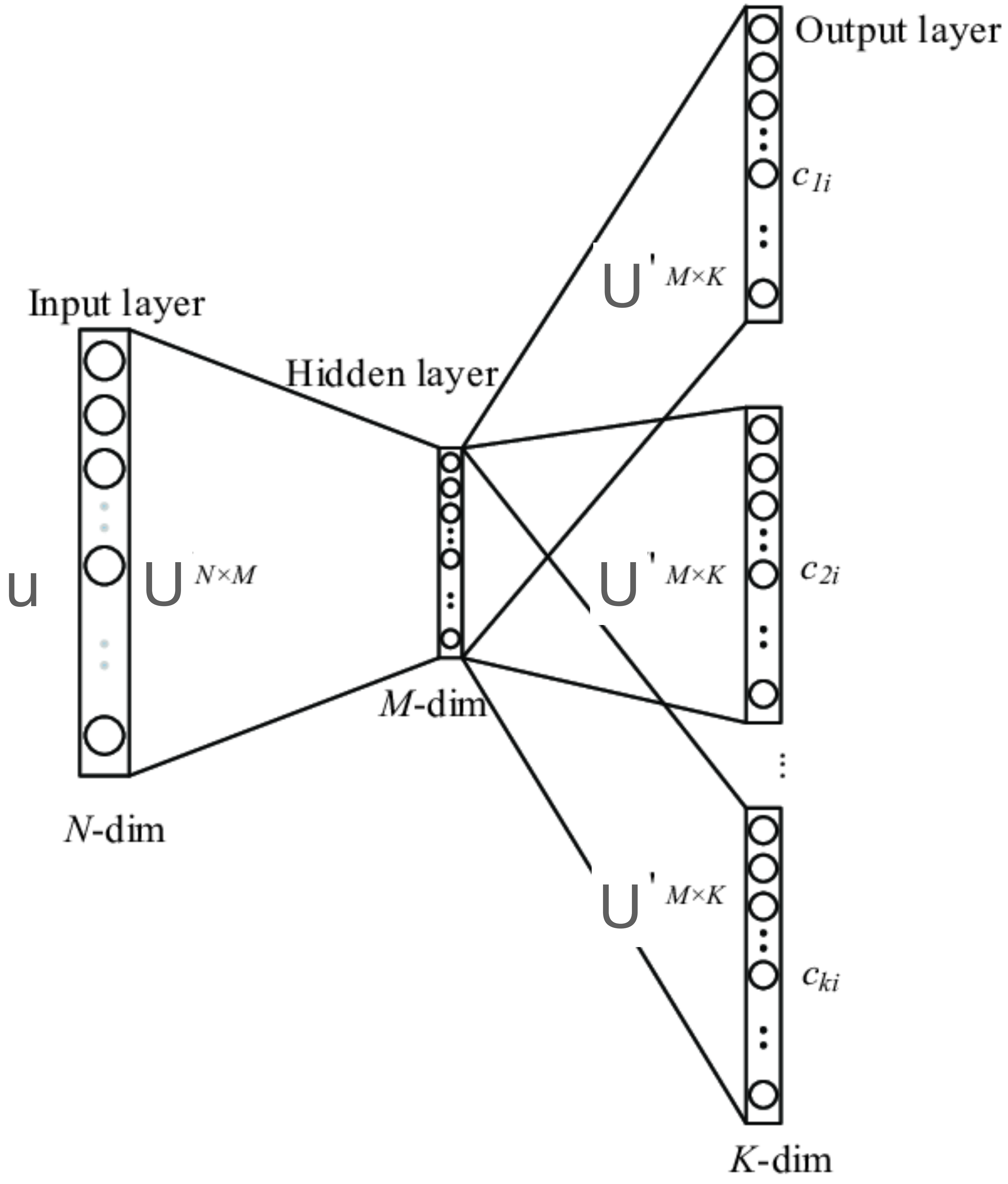
A node v is selected according to the function $p_k(u, v) = \dfrac{w_k(u, v)}{z_k(u)}$, with $z_k(u)$ being a normalisation factor for u in layer k (the sum of $w_k(u, v)$ ∀ $v \neq u$).

We go up a layer with probability $p_k(u_k, u_{k+1}) = \dfrac{w(u_k, u_{k+1})}{w(u_k, u_{k+1}) + w(u_k, u_{k-1})}$, and down with probability $1 - p_k(u_k, u_{k+1})$.

struc2vec

- Four main steps -
    1. Determine similarities between vertex in pairs for different neighbourhood sizes.
    2. Construct a weighted multilayer graph. Each layer = a level in the hierarchy measuring structural similarity; number of layers = diameter of graph.
    3. Generate context for each node in constructed graph using biased random walks.
    4. Use techniques like skip-gram to learn latent representations from context.

Learning Latent Representations



- Use any technique to generate context nodes for a given node u
- Authors made use of skip-gram.
- This creates the latent vectors for every node in the network.

Source: https://upload.wikimedia.org/wikipedia/commons/9/95/Skip-gram.png

Some optimisations used

- Since weights are calculated for a lot of edges, the authors discussed 3 optimisation strategies.

    1. Reducing the length of degree sequences considered

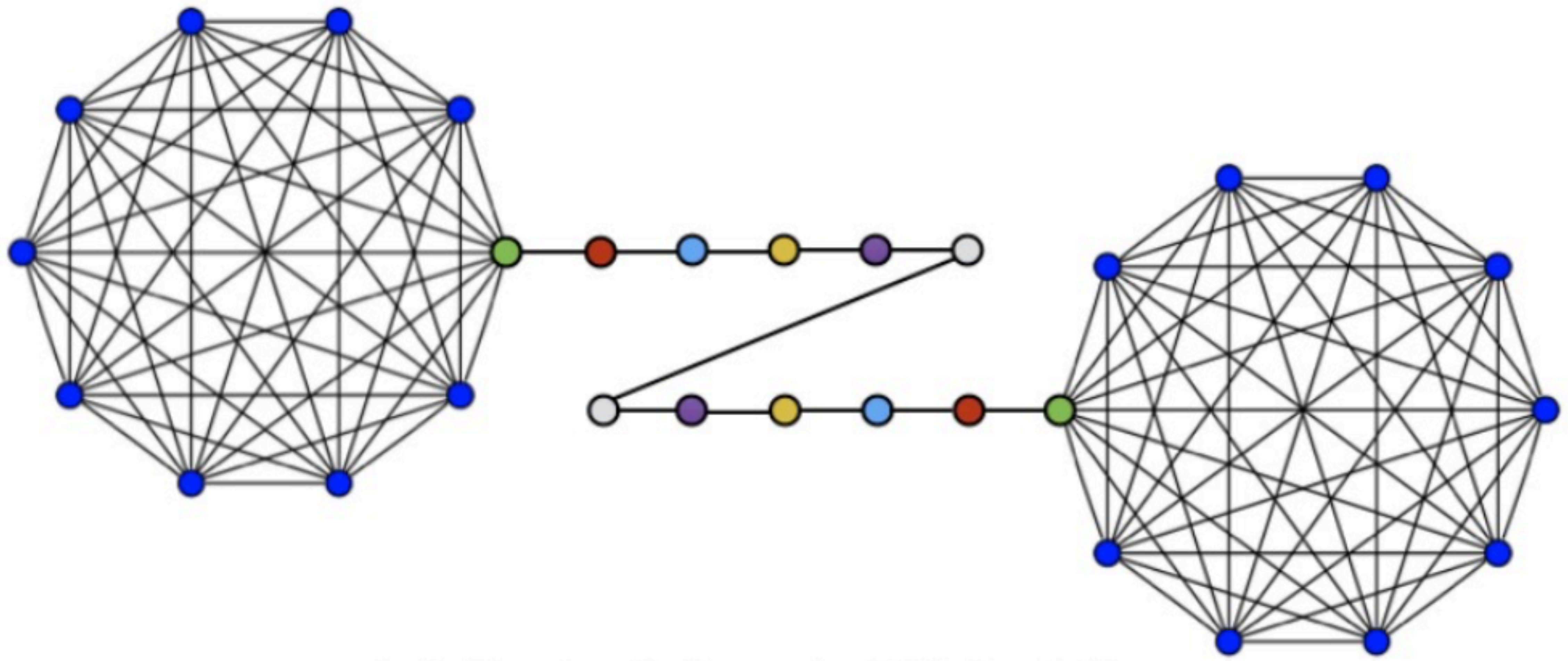
    2. Reducing the number of pairwise similarity calculations

    3. Reducing the number of layers.

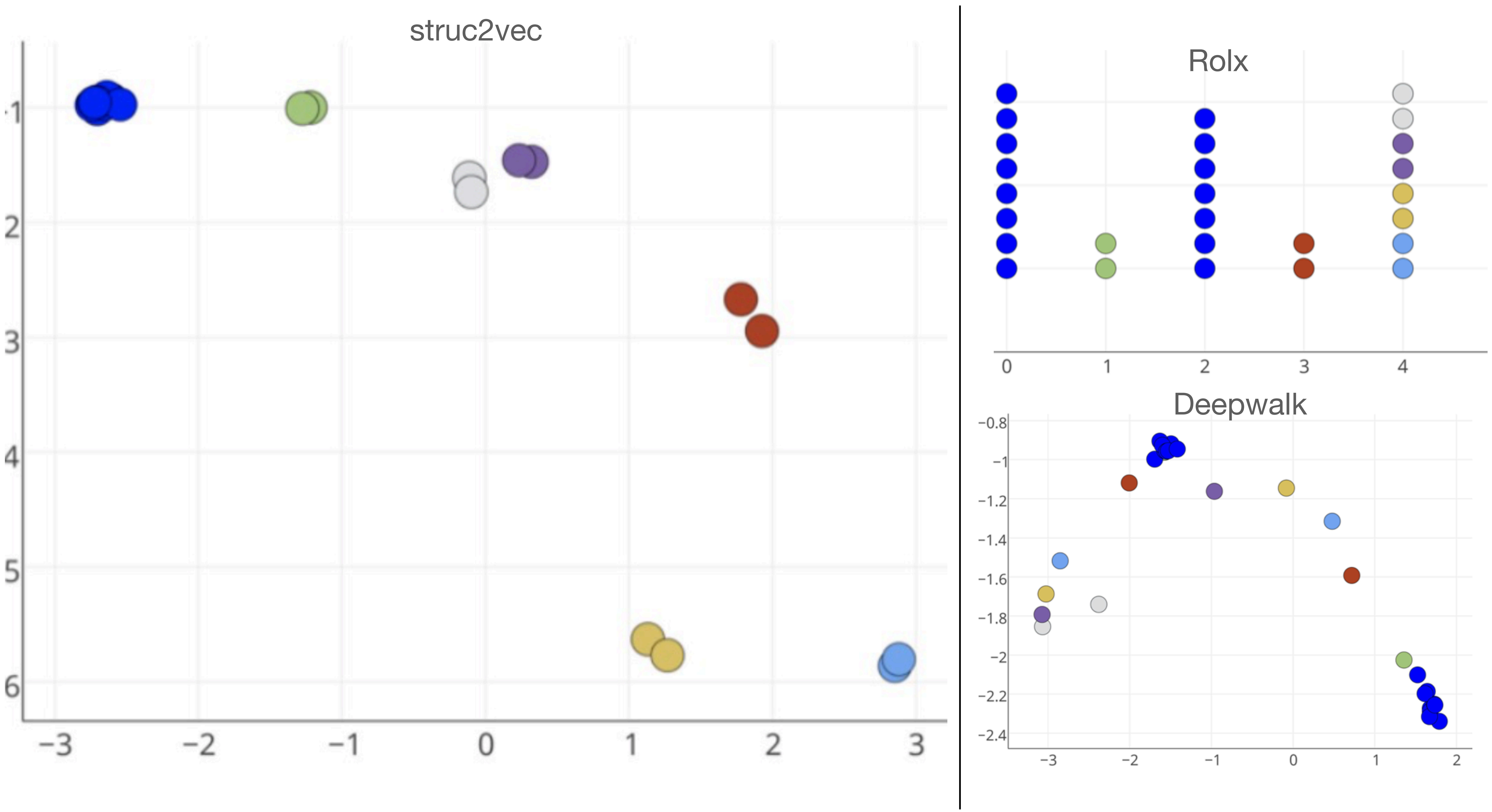
Results & Conclusion

Results

- Barbell graph = two complete graphs, connected by a path graph.



(a) Barbell Graph B(10, 10)


struc2vec
Rolx
Deepwalk

Thank You!