Node Representations

- Map network nodes into Euclidean space
  - aka. network embedding

- Many ways to embed nodes
- Right way depends on application

- Preserve distances
- Find cliques
- Preserve degrees
Nodes in networks have specific roles
- eg., individuals, web pages, proteins, etc

Structural identity
- identification of nodes based on network structure (no other attribute)
- often related to role played by node

Automorphism: strong structural equivalence

- Red, Green: automorphism
- Purple, Brown: structurally similar
Related Work

- **word2vec**: framework to embed words (from sentences) into Euclidean space [arXiv’13]
- **deepwalk**: embed network nodes generating sentences through random walks [KDD’14]
- **node2vec**: use *biased* random walks to generate sentences [KDD’16]

- **rolx**: use node-feature matrix to compute low rank matrix for roles [KDD’12]

Walk on original network to generate context
struc2vec

- Novel framework for node representations based on structural identity
  - structurally similar nodes close in space

- **Key ideas**
  - Structural similarity does not depend on hop distance
    - neighbor nodes can be different, far away nodes can be similar
  - Structural identity as a hierarchical concept
    - depth of similarity varies
  - Flexible four step procedure
    - operational aspect of steps are flexible
Step 1: Structural Similarity

- Hierarchical measure for structural similarity between two nodes
- $R_k(u)$: set of nodes at distance $k$ from $u$ (ring)
- $s(S)$: ordered degree sequence of set $S$

$s(R_0(u)) = 4$
$s(R_0(v)) = 3$
$s(R_1(u)) = 1, 3, 4, 4$
$s(R_1(v)) = 4, 4, 4$
$s(R_2(u)) = 2, 2, 2, 2$
$s(R_2(v)) = 1, 2, 2, 2, 2$
Step 1: Structural Similarity

- $g(D_1, D_2)$: distance between two ordered sequences
  - Cost of pairwise alignment: $\max(a, b) / \min(a, b) - 1$
  - Optimal alignment by DTW in our framework

- $f_k(u, v)$: structural distance between nodes $u$ and $v$ considering first $k$ rings
  - $f_k(u, v) = f_{k-1}(u, v) + g(s(R_k(u)), s(R_k(v)))$

- $s(R_0(u)) = 4$
- $s(R_0(v)) = 3$
- $g(\cdot, \cdot) = 0.33$

- $s(R_1(u)) = 1, 3, 4, 4$
- $s(R_1(v)) = 4, 4, 4$
- $g(\cdot, \cdot) = 3.33$

- $s(R_2(u)) = 2, 2, 2, 2$
- $s(R_2(v)) = 1, 2, 2, 2, 2$
- $g(\cdot, \cdot) = 1$

- $f_0(u, v) = 0.33$
- $f_1(u, v) = 3.66$
- $f_2(u, v) = 4.66$
Step 2: Multi-layer graph

- Encodes structural similarity between all node pairs
  - Each layer is weighted complete graph
    - corresponds to similarity hierarchies
  - Edge weights in layer $k$
    - $w_k(u,v) = \exp\{-f_k(u,v)\}$
  - Connect corresponding nodes in adjacent layers
Step 3: Generate Context

- Context generated by biased random walk
  - walking on multi-layer graph

- Walk in current layer with probability $p$
  - choose neighbor according to edge weight
  - RW prefers more similar nodes

- Change layer with probability $1-p$
  - choose up/down according to edge weight
  - RW prefers layer with less similar neighbors
Step 4: Learn Representation

- For each node, generate set of independent and relative short random walks
  - context for node; sentences of a language

- Train a neural network to learn latent representation for nodes
  - maximize probability of nodes within context
  - Skip-gram (Hierarchical Softmax) adopted
Optimization

- Reduce time to generate/store multi-layer graph and context for nodes
  - OPT1: Reduce length of degree sequences
    - use pairs (degree, number of occurrences)
  - OPT2: Reduce number of edges in multi-layer graph
    - only $\log n$ neighbors per node
  - OPT3: Reduce number of layers in multi-layer graph
    - fixed (small) number of layers
- Scales quasi-linearly
  - over 1 million nodes
Isomorphic nodes very close in space

—is similar with OPTs

Barbell Network

deepwalk

node2vec

struc2vec

rollox

- Isomorphic nodes very close in space
- similar with OPTs
Mirrored Karate Network

Similar roles close in space

node2vec

struc2vec

Similar roles close in space
Airport Classification

- Struc2vec helps classification if labels related to role of nodes

- Air traffic network: airports, commercial flights
  - Brazilian, USA, European (collected from public data)
  - Airport activity measured in number of flights or movement of people
  - Four labels according to quartiles of activity

- Struc2vec (and others) learn node representation from network
  - No labels or activity used here
Airport Classification

- Node representations used to train classifier
  - logistic regression, L2 normalization

- struc2vec superior performance
- 50% improvement in Brazilian network
- Activity related to structure more than neighbors or degree
Conclusion

- Structural identity: symmetry concept based on network, related to node roles
- \textit{struc2vec}: flexible framework to learn representations for structural identity
  - multi-layer graph encodes structural similarity
- \textit{struc2vec} helps classification based on roles
- Yet another useful kind of embedding
  - not necessarily a substitute for others

Find the right embedding for your task!
Thank You!

- Questions and comments?

- struc2vec (source code and datasets)
  https://github.com/leoribeiro/struc2vec
Scalability

- $G(n,p)$ network model, avg. deg 10
  - avg running time over 10 networks, OPTs on

- Time dominated by computing degree sequences of rings (yet to be optimized)
Distances

- Euclidean distance distribution in mirrored Karate network
- Mirrored pairs much closer than all pairs
- Not for node2vec
Robustness

- Structural similarity under edge removal
  - \( G \) is a social network
  - each edge present in \( G_{1,2} \) with prob \( s \)

- Euclidean distance distribution
- Corresponding pairs much closer
- Even when \( s \) is moderate