Deep Learning for Network Biology

Marinka Zitnik and Jure Leskovec

Stanford University

This Tutorial

snap.stanford.edu/deepnetbio-ismb

ISMB 2018

July 6, 2018, 2:00 pm - 6:00 pm

This Tutorial

1) Node embeddings

- Map nodes to low-dimensional embeddings
- *Applications:* PPIs, Disease pathways

2) Graph neural networks

- Deep learning approaches for graphs
- *Applications:* Gene functions

3) Heterogeneous networks

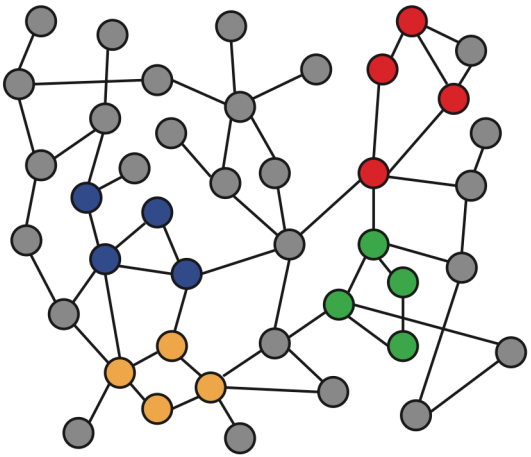
- Embedding heterogeneous networks
- *Applications:* Human tissues, Drug side effects

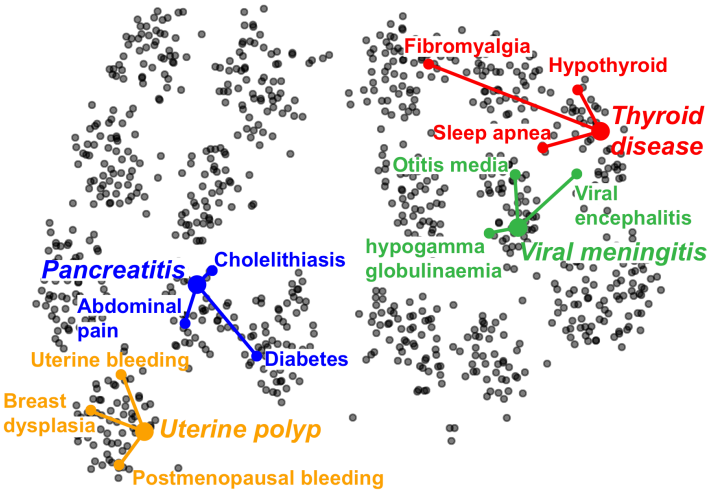
Part 1: Node Embeddings

Some materials adapted from:

- Hamilton et al. 2018. Representation Learning on Networks. *WWW.*

Embedding Nodes





Input **Output**

Intuition: Map nodes to d-dimensional embeddings such that similar nodes in the graph are embedded close together

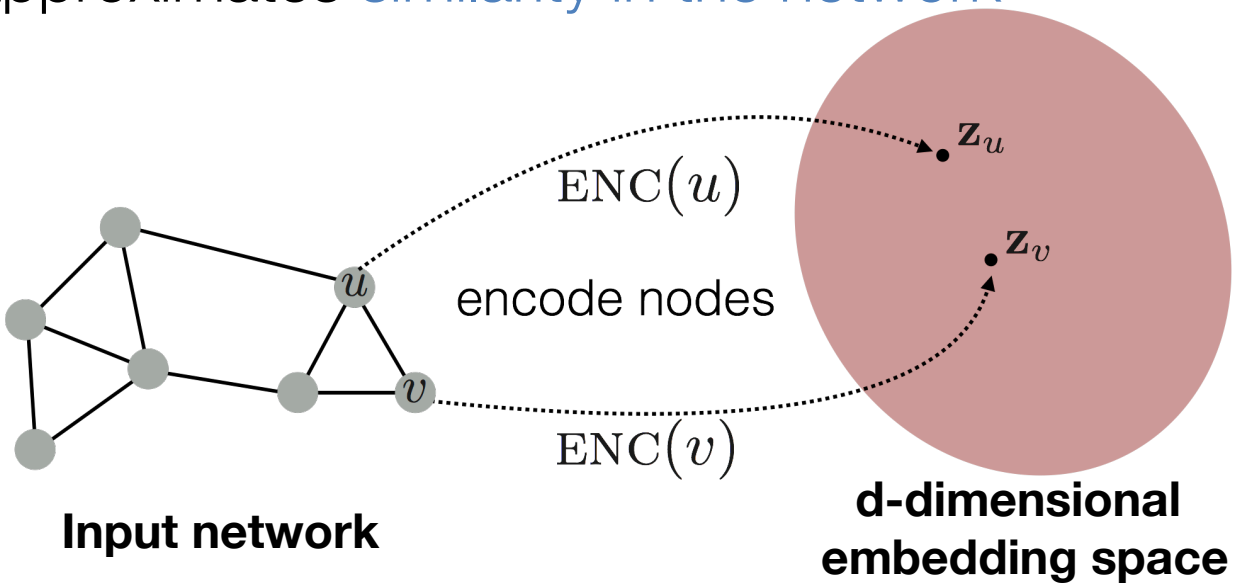
Setup

- Assume we have a graph $\mathbf{G}$:
  - $\mathbf{V}$ is the vertex set
  - $\mathbf{A}$ is the adjacency matrix (assume binary)

  - **No node features or extra information is used!**

Embedding Nodes

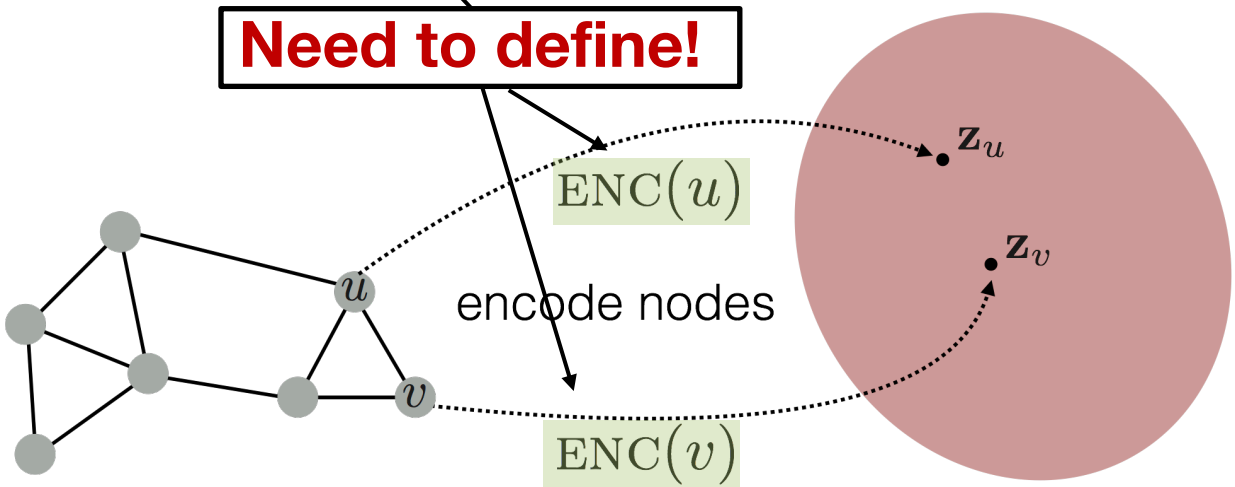
Goal: Map nodes so that similarity in the embedding space (e.g., dot product) approximates similarity in the network



Embedding Nodes

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

Need to define!



Input network

d-dimensional embedding space

Learning Node Embeddings

1. **Define an encoder** (a function ENC that maps node u to embedding $\boldsymbol{z}_u$)
2. **Define a node similarity function** (a measure of similarity in the input network)
3. **Optimize parameters of the encoder so that:**

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

Two Key Components

1. **Encoder** maps a node to a d-dimensional vector:

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

$\mathbf{d}$-dimensional embedding

node in the input graph

2. **Similarity function** defines how relationships in the input network map to relationships in the embedding space:

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

Similarity of u and v in the network

dot product between node embeddings

Embedding Methods

- Many methods use similar encoders:
  - node2vec, DeepWalk, LINE, struc2vec
- These methods use different notions of node similarity:
  - Two nodes have similar embeddings if:
    - they are connected?
    - they share many neighbors?
    - they have similar local network structure?
    - etc.

Outline of This Section

1. Adjacency-based similarity
2. Random walk approaches
3. Biomedical applications

Adjacency-based Similarity

Material based on:

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

Adjacency-based Similarity

- **Similarity function** is 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

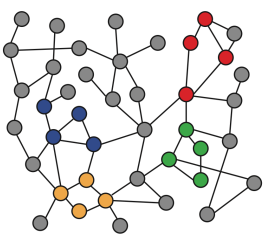
Adjacency-based Similarity

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

- Find embedding matrix $\mathbf{Z} \in \mathbb{R}^{d \times |V|}$ that minimizes the loss $\mathcal{L}$:
  - Option 1: Stochastic gradient descent (SGD)
    - Highly scalable, general approach
  - Option 2: Solve matrix decomposition solvers
    - e.g., SVD or QR decompositions
    - Need to derive specialized solvers

Adjacency-based Similarity

- $O(|V|^2)$ runtime
  - Must consider all node pairs
  - $O([E|)$ if summing over non-zero edges (e.g., Natarajan et al., 2014)
- $O(|V|)$ parameters
  - One learned embedding per node
- Only consider direct connections



Red nodes are obviously more similar to **Green nodes** compared to **Orange nodes**, despite none being directly connected

Outline of This Section

1. Adjacency-based similarity ✓
2. Random walk approaches ☞
3. Biomedical applications

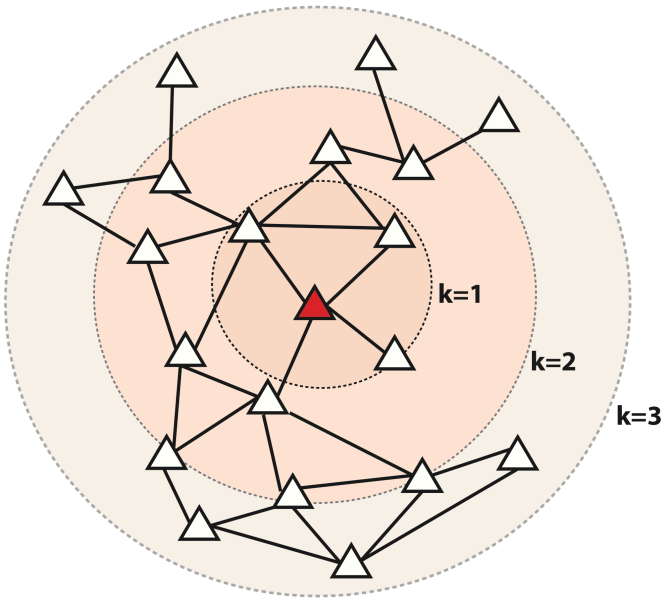
Random Walk Approaches

Material based on:

- Perozzi et al. 2014. DeepWalk: Online Learning of Social Representations. *KDD.*
- Grover et al. 2016. node2vec: Scalable Feature Learning for Networks. *KDD.*
- Ribeiro et al. 2017. struc2vec: Learning Node Representations from Structural Identity. *KDD.*

Multi-Hop Similarity

Idea: Define node similarity function based on higher-order neighborhoods



- **Red:** Target node
- **k=1:** 1-hop neighbors
  - $\mathbf{A}$ (i.e., adjacency matrix)
- **k= 2:** 2-hop neighbors
- **k=3:** 3-hop neighbors

How to stochastically define these higher-order neighborhoods?

Unsupervised Feature Learning

- Intuition: Find embedding of nodes to d-dimensions that preserves similarity
- Idea: Learn node embedding such that nearby nodes are close together
- Given a node u, how do we define nearby nodes?
  - $N_R(u)$ … neighbourhood of u obtained by some strategy R

Feature Learning as Optimization

- Given $G = (V, E)$
- Goal is to learn $f: u \rightarrow \mathbb{R}^d$
  - where f is a table lookup
    - We directly "learn" coordinates $\boldsymbol{z_u} = f(u)$ of u
- Given node u, we want to learn feature representation $f(u)$ that is predictive of nodes in u's neighborhood $N_{\mathrm{R}}(u)$

$$\max_f \sum_{u \in V} \log \Pr(N_{\mathrm{R}}(u) | \boldsymbol{z}_{\mathrm{u}})$$

Unsupervised Feature Learning

Goal: Find embedding $\boldsymbol{z}_u$ that predicts nearby nodes $N_R(u)$:

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

Assume conditional likelihood factorizes:

$$P(N_R(u)|\mathbf{z}_u) = \prod_{n_i \in N_R(u)} P(n_i|\mathbf{z}_u)$$

Random-walk Embeddings

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

Probability that u and v co-occur in a random walk over the network

Why Random Walks?

1. **Flexibility:** Stochastic definition of node similarity:
   - Local and higher-order neighborhoods

2. **Efficiency:** Do not need to consider all node pairs when training
   - Consider only node pairs that co-occur in random walks

Random Walk Optimization

1. Simulate many short random walks starting from each node using a strategy R
2. For each node u, get $\mathrm{N}_R(u)$ as a sequence of nodes visited by random walks starting at u
3. For each node u, learn its embedding by predicting which nodes are in $\mathrm{N}_R(u)$:

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

Random Walk Optimization

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

sum over all nodes u

sum over nodes v seen on random walks starting from u

predicted probability of u and $\mathbf{v}$ co-occuring on random walk, i.e., use softmax to parameterize $P(v|\mathbf{z}_u)$

Random walk embeddings = z_u minimizing $\mathcal{L}$

Random Walk Optimization

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 problem is normalization term in the softmax function?

Solution: Negative Sampling

Solution: Negative sampling (Mikolov et al., 2013)

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

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

sigmoid function

random distribution over all nodes

i.e., instead of normalizing w.r.t. all nodes, just normalize against $\mathbf{k}$ random **negative samples**

Random Walks: Overview

1. Simulate many short random walks starting from each node using a strategy R
2. For each node u, get $\mathrm{N}_R(u)$ as a sequence of nodes visited by random walks starting at u
3. For each node u, learn its embedding by predicting which nodes are in $\mathrm{N}_R(u)$:

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

Can efficiently approximate using negative sampling

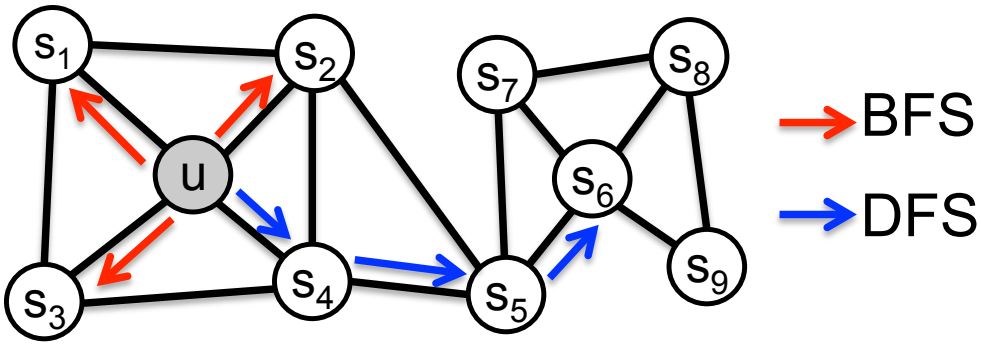
What is the strategy R?

- **So far:**
  - Given simulated random walks, we described how to optimize node embeddings

- **What strategies can we use to obtain these random walks?**
  - Simplest idea:
    - Fixed-length, unbiased random walks starting from each node (i.e., DeepWalk from Perozzi et al., 2013)
  - **Can we do better?**
    - Grover et al., 2016; Ribeiro et al., 2017; Abu-El-Haija et al., 2017 and many others

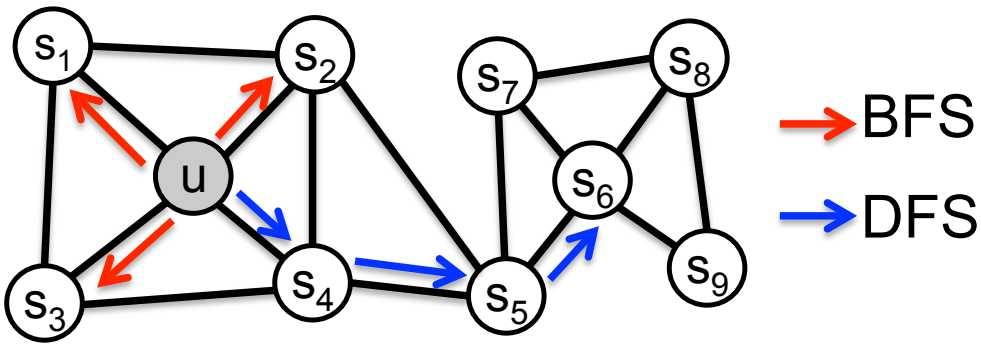
node2vec: Biased Walks

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 $N_R(u)$ 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

Interpolate BFS and DFS

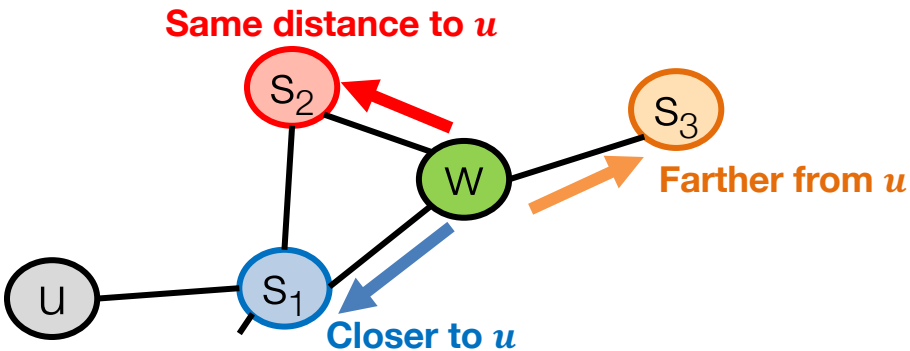
Biased random walk R that given a node u generates neighborhood $N_R(u)$

- Two parameters:
  - Return parameter p:
    - Return back to the previous node
  - In-out parameter q:
    - Moving outwards (DFS) vs. inwards (BFS)

Biased Random Walks

Biased 2nd-order random walks explore network neighborhoods:

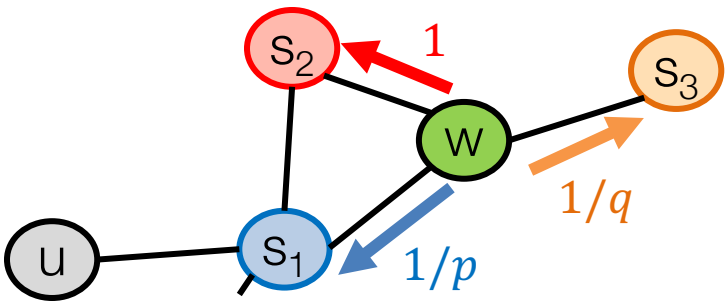
- Rnd. walk started at u and is now at w
- **Insight:** Neighbors of w can only be:



Idea: Remember where that walk came from

Biased Random Walks

- Walker is at $\mathbf{w}$. Where to go next?

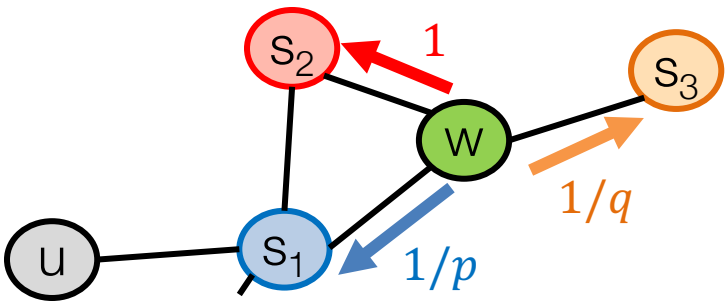


$1/p, 1/q, 1$ are unnormalized probabilities

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

Biased Random Walks

- Walker is at $\mathbf{w}$. Where to go next?



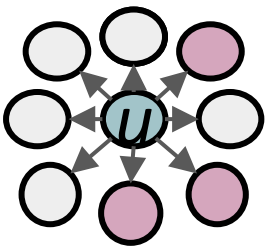
$\mathbf{w} \rightarrow$	s_1	$1/p$
	s_2	1
	s_3	$1/q$

Unnormalized transition prob.

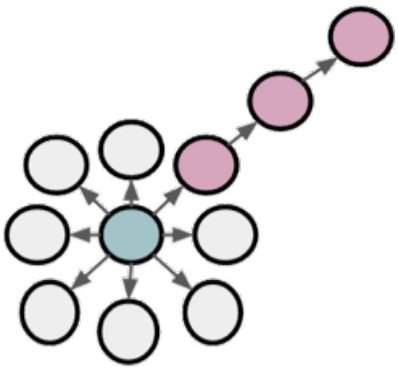
- **BFS-like** walk: Low value of p
- **DFS-like** walk: Low value of q

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

BFS vs. DFS



BFS:
Micro-view of neighbourhood



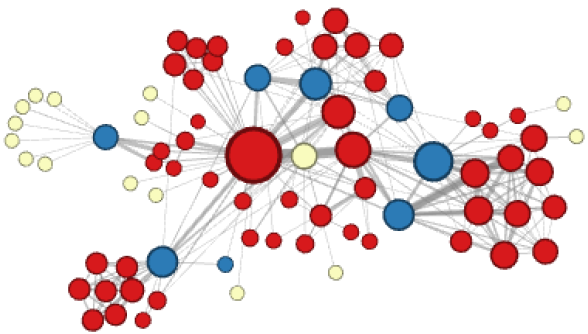
DFS:
Macro-view of neighbourhood

Experiment: Micro vs. Macro

Interactions of characters in a novel:

p=1, q=2

Microscopic view of the network neighbourhood



p=1, q=0.5

Macroscopic view of the network neighbourhood

Summary So Far

- **Idea:** Embed nodes so that distances in the embedding space reflect node similarities in the network

- Different notions of **node similarity**:
  - Adjacency-based (i.e., similar if connected)
  - Random walk approaches:
    - Fixed-length, unbiased random walks starting from each node in the original network (Perozzi et al., 2013)
    - Fixed-length, biased random walks on the original network (node2vec, Grover et al., 2016)

Summary So Far

- **So what method should I use..?**
- No one method wins in all cases….
  - e.g., node2vec performs better on node classification while multi-hop methods performs better on link prediction (Goyal and Ferrara, 2017 survey).
- Random walk approaches are generally more efficient (i.e., $O(|E|)$ vs. $O(|V|^2)$)
- **In general:** Must choose def'n of node similarity that matches application!

Outline of This Section

1. Adjacency-based similarity ✓
2. Random walk approaches ✓
3. Biomedical applications ☞

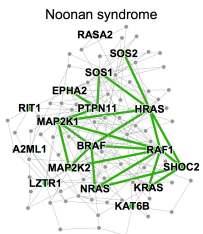
Biomedical Applications

Material based on:

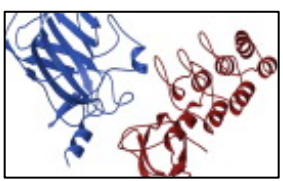
- Grover et al. 2016. node2vec: Scalable Feature Learning for Networks. *KDD.*
- Zitnik and Leskovec. 2017. Predicting Multicellular Function through Multilayer Tissue Networks. *ISMB.*
- Agrawal et al. 2018. Large-scale analysis of disease pathways in the human interactome. *PSB.*

Biomedical Applications

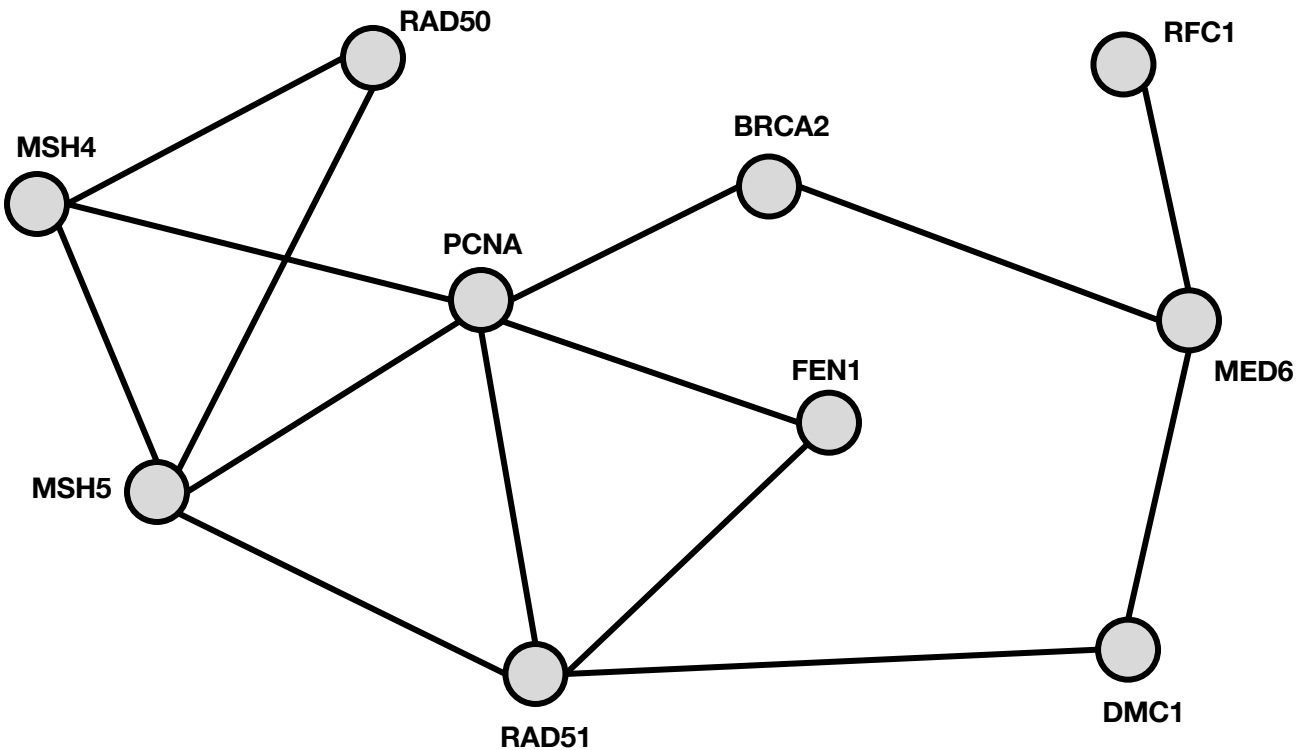
1. Disease pathway detection:
   - Identify proteins whose mutation is linked with a particular disease
   - **Task:** Multi-label node classification



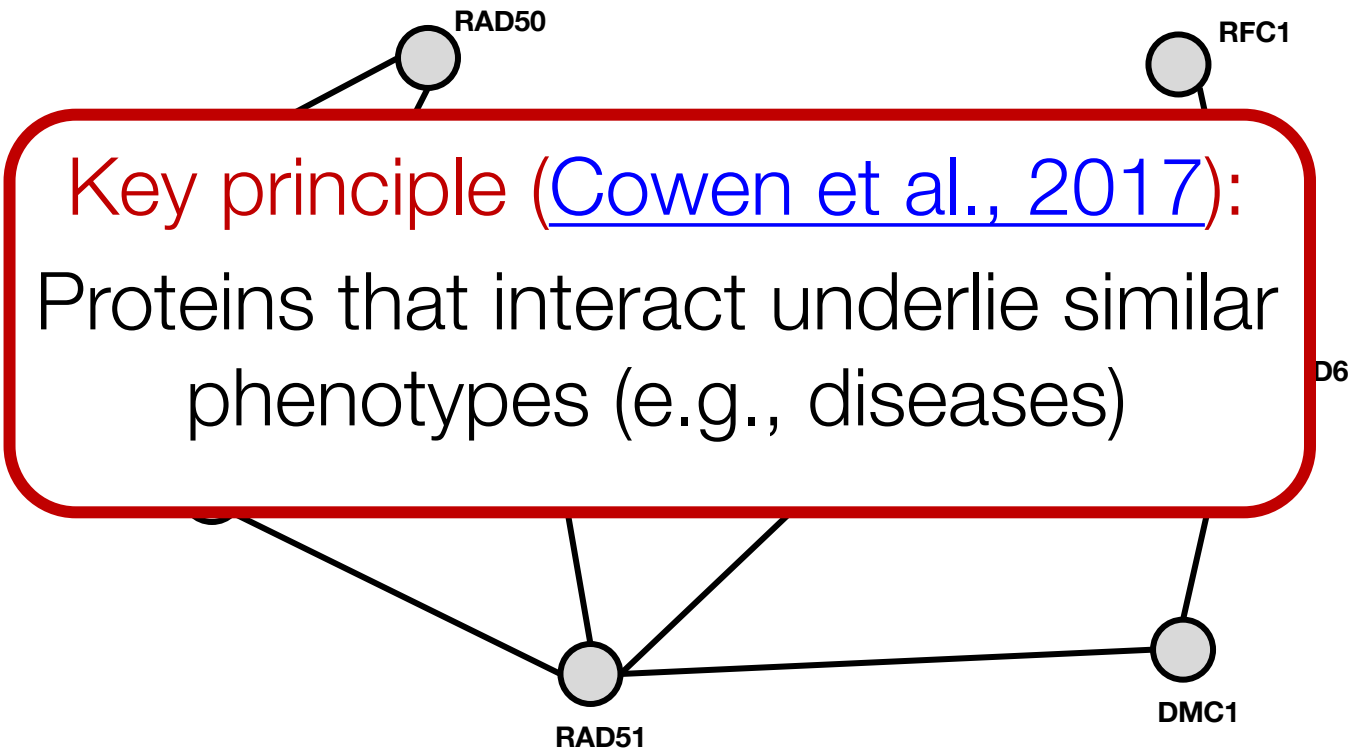
2. Protein interaction prediction:
   - Identify protein pairs that physically interact in a cell
   - **Task:** Link prediction



Human Interactome



Human Interactome

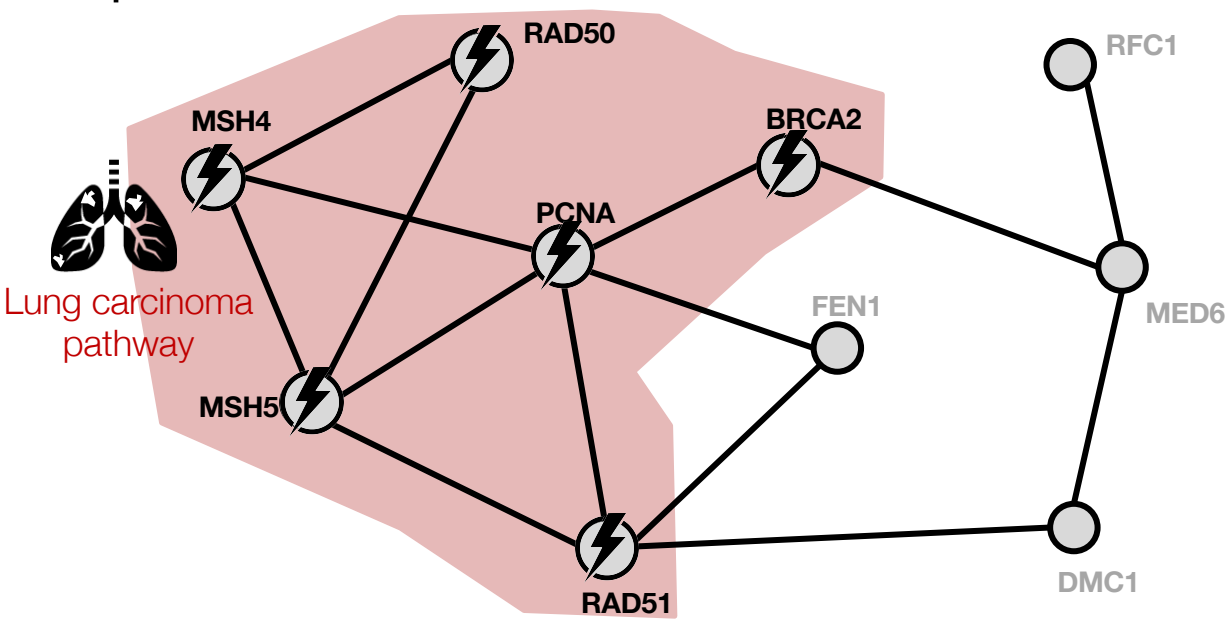


Key principle (Cowen et al., 2017):

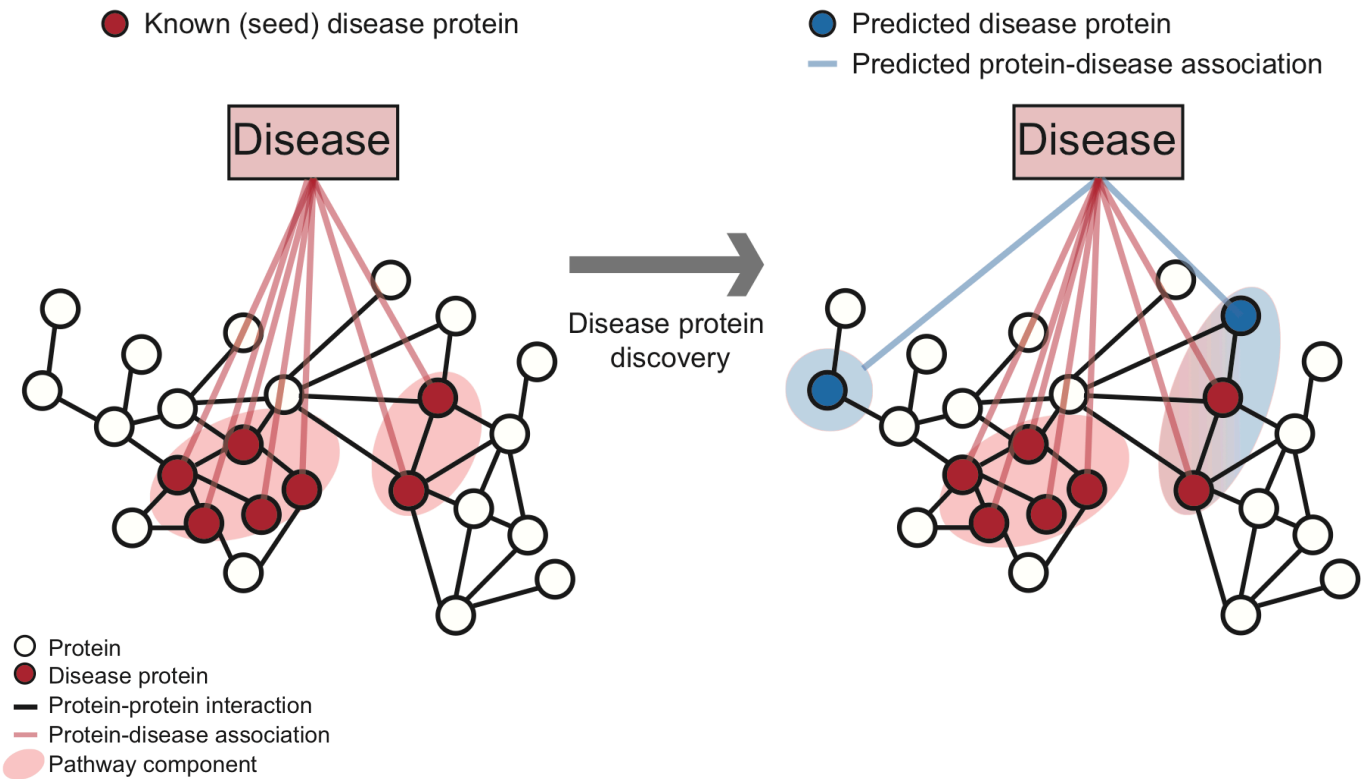
Proteins that interact underlie similar phenotypes (e.g., diseases)

Disease Pathways

- Pathway: Subnetwork of interacting proteins associated with a disease



Disease Pathways: Task



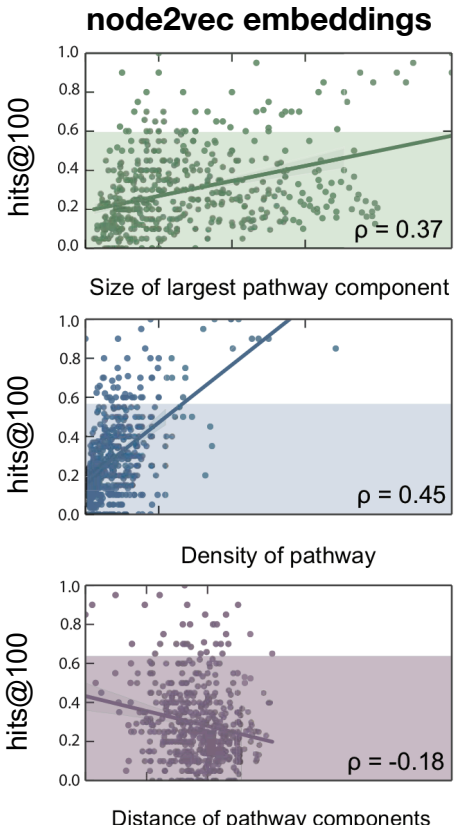
Disease Pathway Dataset

- Protein-protein interaction (PPI) network culled from 15 knowledge databases:
  - 350k physical interactions, e.g., metabolic enzyme-coupled interactions, signaling interactions, protein complexes
  - All protein-coding human genes (21k)
- Protein-disease associations:
  - 21k associations split among 519 diseases
- **Multi-label node classification:** every node (i.e., protein) can have 0, 1 or more labels (i.e., disease associations)

Experimental Setup

- Two main stages:
  1. Take the PPI network and use node2vec to learn an **embedding for every node**
  2. For each disease:
     - Fits a logistic regression classifier that predicts disease proteins based on the embeddings:
       - Train the classifier using training proteins
       - Predict disease proteins in the test test: probability that a particular protein is associated with the disease

Pathways: Results

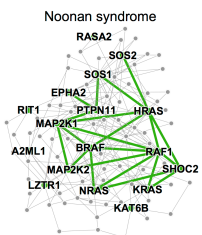


- Best performers:
  - node2vec embeddings
    $\text{hits@100} = 0.40$
  - DIAMOnD
    $\text{hits@100} = 0.30$
  - Matrix completion
    $\text{hits@100} = 0.29$
- Worst performer:
  - Neighbor scoring
    $\text{hits@100} = 0.24$

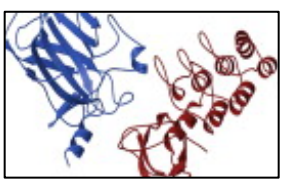
hits@100: fraction of all the disease proteins are ranked within the first 100 predicted proteins

Biomedical Applications

1. Disease pathway detection:
   - Identify proteins whose mutation is linked with a particular disease
   - **Task:** Multi-label node classification



2. Protein interaction prediction:
   - Identify protein pairs that physically interact in a cell
   - **Task:** Link prediction



Protein-Protein Interaction

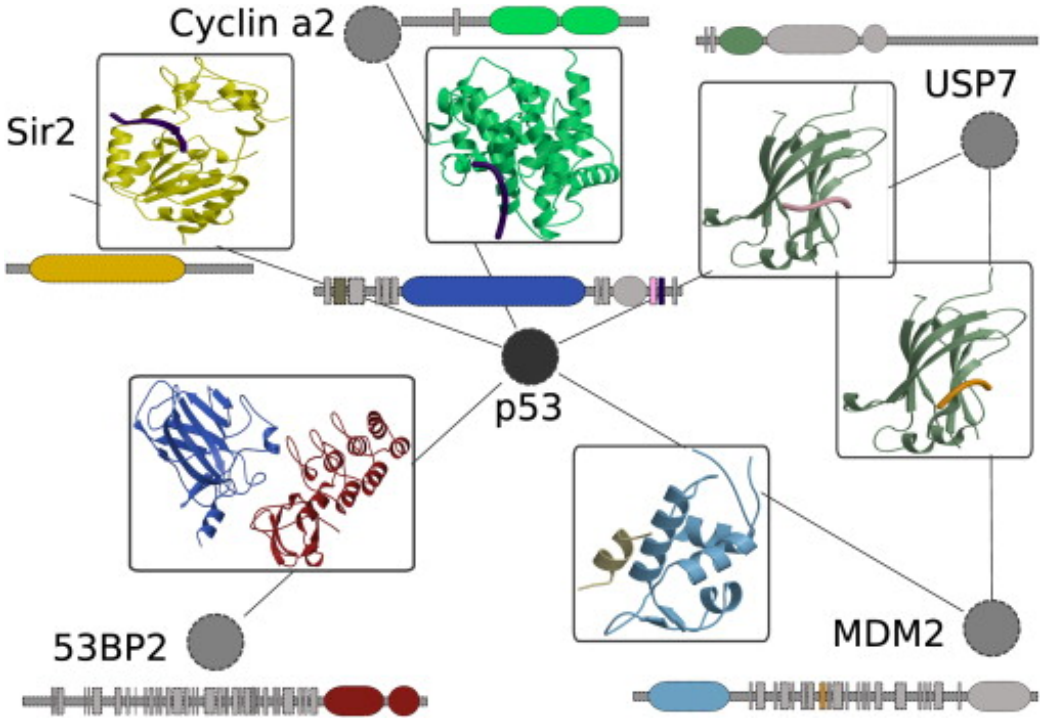
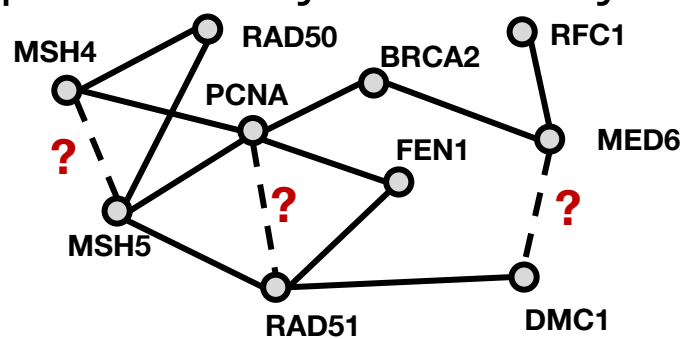


Image from: Perkins et al. [Transient Protein-Protein Interactions: Structural, Functional, and Network Properties]. *Structure*. 2010.

Network Data

- Human PPI network:
  - Experimentally validated physical protein-protein interactions from the BioGRID
- **Link prediction:** Given two proteins, predict probability that they interact



Learning Edge Embeddings

- **So far:** Methods learn embeddings for nodes:
  - Great for tasks involving individual nodes (e.g., node classification)
- **Question:** How to address tasks involving pairs of nodes (e.g., link prediction)?
- **Idea:** Given u and v, define an operator g that generates an embedding for pair (u, v):

$$\boldsymbol{z}_{(u,v)} = g(u, v)$$

Learning Edge Embeddings

How to define operator g?

- **Desiderata:** The operator needs to be defined for any pair of nodes, even if the nodes are not connected
- We consider four choices for g:

Scoring node pairs	Definition
(a) Average	$[\mathbf{z}_u \boxplus \mathbf{z}_v]_i = \frac{\mathbf{z}_u(i)+\mathbf{z}_v(i)}{2}$
(b) Hadamard	$[\mathbf{z}_u \boxdot \mathbf{z}_v]_i = \mathbf{z}_u(i) * \mathbf{z}_v(i)$
(c) Weighted-L1	$\lVert \mathbf{z}_u \cdot \mathbf{z}_v \rVert_{\bar{1}i} = \lvert \mathbf{z}_u(i) - \mathbf{z}_v(i) \rvert$
(d) Weighted-L2	$\lVert \mathbf{z}_u \cdot \mathbf{z}_v \rVert_{\bar{2}i} = \lvert \mathbf{z}_u(i) - \mathbf{z}_v(i) \rvert^2$

Experimental Setup

- We are given a PPI network with a certain fraction of edges removed:
  - Remove about 50% of edges
  - Randomly sample an equal number of node pairs at random which have no edge connecting them
  - Explicitly removed edges and non-existent (or false) edges form a balanced test data set
- Two main stages:
  1. Use node2vec to learn an **embedding for every node** in the filtered PPI network
  2. Predict a score for every protein pair in the test set based on the embeddings

PPI Prediction: Results

Op	Algorithm	Dataset		
		Facebook	PPI	arXiv
	Common Neighbors	0.8100	0.7142	0.8153
	Jaccard's Coefficient	0.8880	0.7018	0.8067
	Adamic-Adar	0.8289	0.7126	0.8315
	Pref. Attachment	0.7137	0.6670	0.6996
(a)	Spectral Clustering	0.5960	0.6588	0.5812
	DeepWalk	0.7238	0.6923	0.7066
	LINE	0.7029	0.6330	0.6516
	node2vec	0.7266	0.7543	0.7221
(b)	Spectral Clustering	0.6192	0.4920	0.5740
	DeepWalk	**0.9680**	0.7441	0.9340
	LINE	0.9490	0.7249	0.8902
	node2vec	**0.9680**	**0.7719**	**0.9366**
(c)	Spectral Clustering	0.7200	0.6356	0.7099
	DeepWalk	0.9574	0.6026	0.8282
	LINE	0.9483	0.7024	0.8809
	node2vec	0.9602	0.6292	0.8468
(d)	Spectral Clustering	0.7107	0.6026	0.6765
	DeepWalk	0.9584	0.6118	0.8305
	LINE	0.9460	0.7106	0.8862
	node2vec	0.9606	0.6236	0.8477

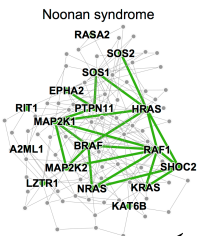
F1 – scores are in [0,1], higher is better

- Learned embeddings drastically outperform heuristic scores
- **Hadamard operator**:
  - Highly stable
  - Best average performance

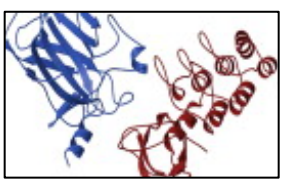
Scoring node pairs	Definition
(a) Average	$[\mathbf{z}_u \boxplus \mathbf{z}_v]_i = \frac{\mathbf{z}_u(i)+\mathbf{z}_v(i)}{2}$
(b) Hadamard	$[\mathbf{z}_u \boxdot \mathbf{z}_v]_i = \mathbf{z}_u(i) * \mathbf{z}_v(i)$
(c) Weighted-L1	$\|\mathbf{z}_u \cdot \mathbf{z}_v\|_{\bar{1}i} = \|\mathbf{z}_u(i) - \mathbf{z}_v(i)\|$
(d) Weighted-L2	$\|\mathbf{z}_u \cdot \mathbf{z}_v\|_{\bar{2}i} = \|\mathbf{z}_u(i) - \mathbf{z}_v(i)\|^2$

Biomedical Applications

1. Disease pathway detection:
   - Identify proteins whose mutation is linked with a particular disease
   - **Task:** Multi-label node classification



2. Protein interaction prediction:
   - Identify protein pairs that physically interact in a cell
   - **Task:** Link prediction



Outline of This Section

1. Adjacency-based similarity ✓
2. Random walk approaches ✓
3. Biomedical applications ✓

PhD Students


Claire Donnat


Mitchell Gordon


David Hallac


Emma Pierson


Geet Sethi


Himabindu Lakkaraju


Rex Ying


Tim Althoff


Will Hamilton


Alex Porter

Post-Doctoral Fellows


Baharan Mirzasoleiman

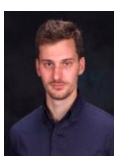

Marinka Zitnik


Michele Catasta


Srijan Kumar

Research Staff


Stephen Bach


Adrijan Bradaschia


Rok Sosic

Industry Partnerships

Funding

Collaborators

Dan Jurafsky, Linguistics, Stanford University
Christian Danescu-Miculescu-Mizil, Information Science, Cornell University
Stephen Boyd, Electrical Engineering, Stanford University
David Gleich, Computer Science, Purdue University
VS Subrahmanian, Computer Science, University of Maryland
Sarah Kunz, Medicine, Harvard University
Russ Altman, Medicine, Stanford University
Jochen Profit, Medicine, Stanford University
Eric Horvitz, Microsoft Research
Jon Kleinberg, Computer Science, Cornell University
Sendhill Mullainathan, Economics, Harvard University
Scott Delp, Bioengineering, Stanford University
Jens Ludwig, Harris Public Policy, University of Chicago

WE'RE HIRING!

Many interesting high-impact projects
in Machine Learning and Large Biomedical Data

Applications: Precision Medicine & Health, Drug Repurposing,
Drug Side Effect modeling, Network Biology, and many more