Comp790-166: Computational Biology

Lecture 4

January 28, 2021

Important Upcoming Dates

- Wellness Days – No class
  - February 16
  - March 11
- Homework 1
  - Assigned February 4
  - Due February 18
- Homework 2
  - Assigned April 2
  - Due April 15
- Course Projects
  - March 9
  - Project Proposal Presentations on March 9, and March 16

Project Opportunity

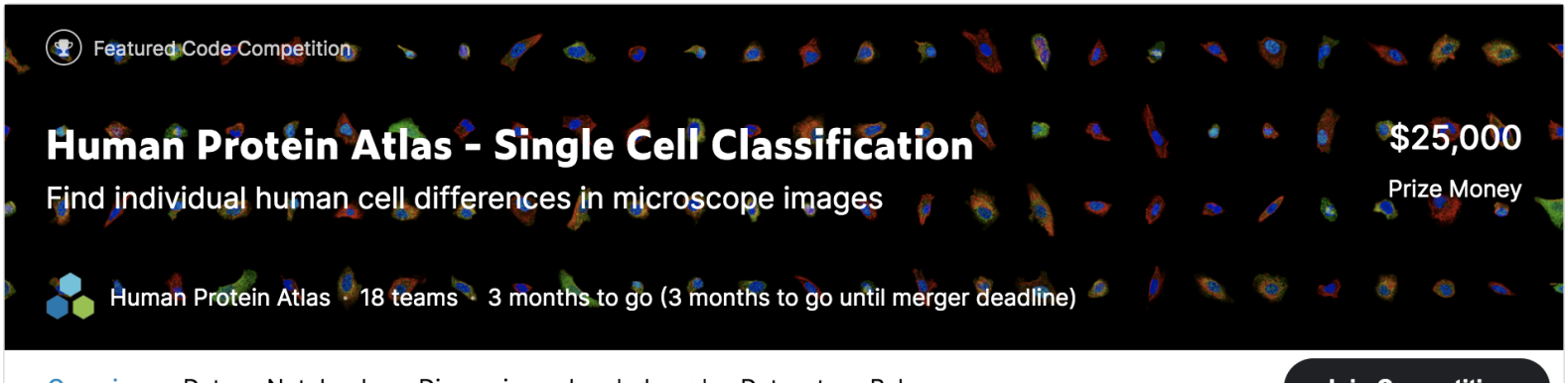


Figure: `https://www.kaggle.com/c/hpa-single-cell-image-classification/overview`

Today

- Overflow graph partitioning from last time (sorry again about going over time!)
  - Stochastic Block Model
  - Affiliation Model
  - Higher order structures and clique counting.
- Graph Embeddings
  - Node2Vec
  - Is there a point to graph embedding for biological networks?
  - Leveraging local information in biological networks
- Signal Processing on Graphs

Stochastic Block Model (SBM)

- **Intuition:** Members of a community should be connected to themselves and to members of other communities in the same way.
- **Model:** Assuming we have N nodes and K communities, we infer two main parameters
  - θ, a matrix of between-community edge probabilities
  - $\mathbf{z}$, a vector of node-to-community assignments

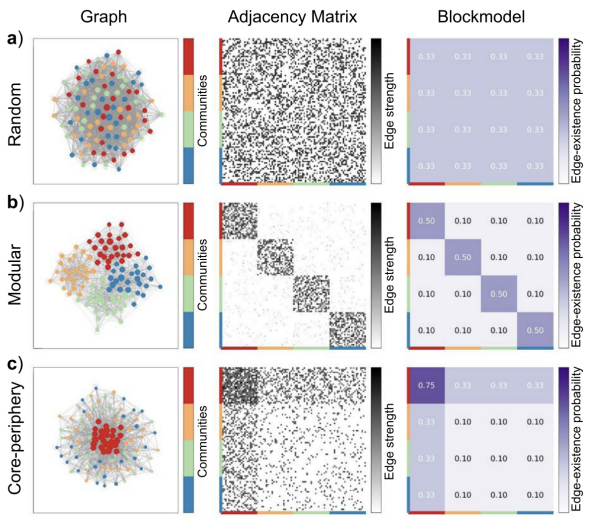


Figure: from Fastkowitz *et al.* Scientific Reports. 2018.

Learning Parameters

Let $\mathbf{Z}$ by an $N \times K$ indicator matrix with $Z_{ik} = 1$ if a node is assigned to community k and $Z_{ik} = 0$ otherwise. $\mathbf{A}$ is our binary $N \times N$ adjacency matrix.

$$A_{ij} \sim \text{Bernoulli}(\theta_{z_i, z_j}) \tag{1}$$

In general write the complete data log-likelihood of the observed graph ($\mathbf{A}$) and the node-to-community assignments ($\mathbf{Z}$) can be written as,

$$\mathcal{L}(\mathbf{A}, \mathbf{Z}) = \log \mathcal{L}(\mathbf{Z}) + \log \mathcal{L}(\mathbf{A} \mid \mathbf{Z}) \tag{2}$$

SBM Complete Data Log Likelihood

$$\log \mathcal{L}(\mathbf{A}, \mathbf{Z}) = \sum_i \sum_q Z_{iq} \log \alpha_q + \frac{1}{2} \sum_{i \neq j} \sum_{q,l} Z_{iq} Z_{jl} \log b(A_{ij}, \theta_{ql}) \qquad (3)$$

- α_q is the probability (in general) of being in community q.
- $b(a, \pi) = \pi^a (1 - \pi)^{1-a}$

SBM Complete Data Log Likelihood, Continued

$\sum_i \sum_q Z_{iq} \log \alpha_q + \frac{1}{2} \sum_{i \neq j} \sum_{q,l} Z_{iq} Z_{jl} \log b(A_{ij}, \theta_{ql})$ can be written completely as,

$$\sum_i \sum_q Z_{iq} \log \alpha_q + \frac{1}{2} \sum_{i \neq j} \sum_{q,l} Z_{iq} Z_{jl} [A_{ij} \log(\theta_{ql}) + (1 - A_{ij}) \log(1 - \theta_{ql}))] \quad (4)$$

Fitting Parameters

- I will not go through it here, but you can either use expectation maximization (EM), belief propagation, or MCMC methods.
  - See → `https://arxiv.org/abs/1207.2328` for EM and BP
  - See → `https://journals.aps.org/pre/abstract/10.1103/PhysRevE.89.012804` for MCMC
- The fastest implementation of SBM model parameters that is most readily scalable to large graphs can be found in GraphTool → `https://graph-tool.skewed.de`

An Application of the SBM in Single-Cell Data

Using single-cell RNA-seq data, the authors build a graph between cells based on gene expression and inferred phenotypically homogeneous groups of cells with an SBM. The authors claim that the modularity-based optimization approach can prevent the identification of small communities.

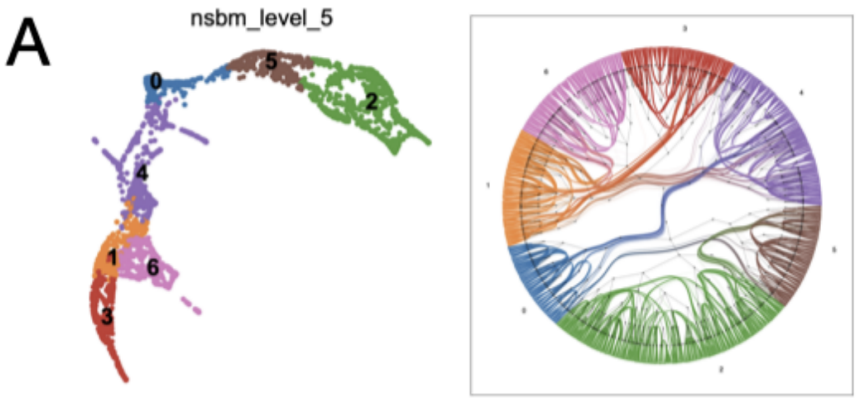


Figure: from
`https://www.biorxiv.org/content/10.1101/2020.06.28.176180v2.full`

SBM in the Analysis of the Human Microbiome

Here nodes are different microbial species and their interactions collected from different bodysites. Learning an ensemble of stochastic block models, where each member of the ensemble characterized several body sites according to similar microbial species interactions.

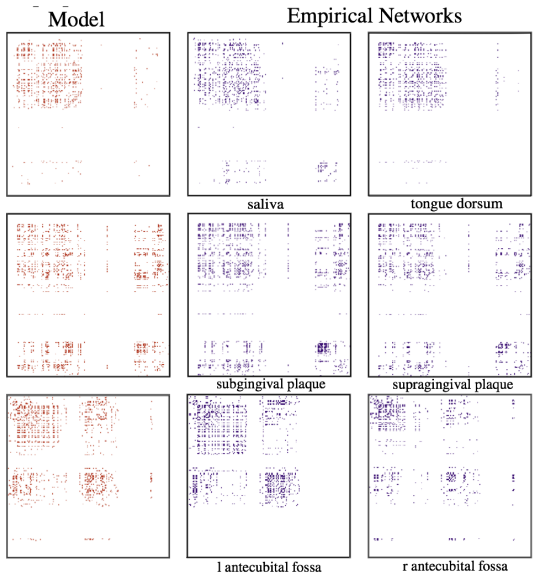


Figure: from Stanley *et al.* IEEE TNSE. 2016.

Affiliation Model for Community Structure

- This is a *soft* clustering approach where overlapping communities are allowed
- Instead of learning a hard node-to-community partition, we learn a node-to-community *propensity*

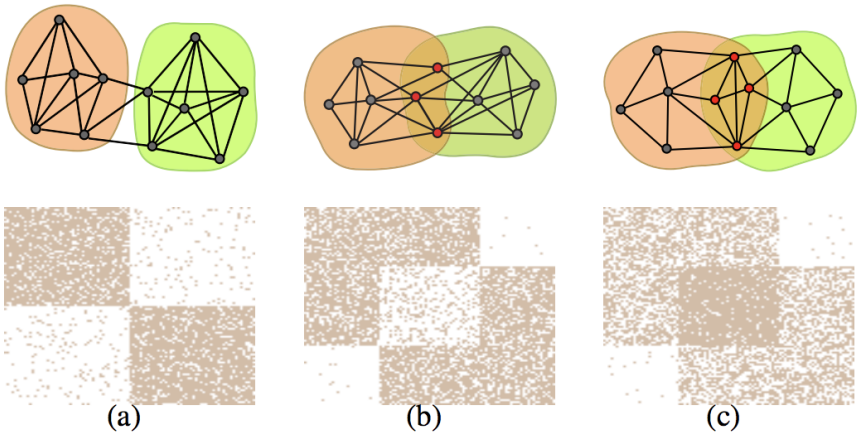


Figure: from Yang *et al.* ICDM. 2012

'BigClam' Approach to Overlapping Communities

Model the existence of an edge between nodes u and v based on the inner product of propensities, F_u and F_v.

DEFINITION 1. *Let F be a nonnegative matrix where F_{uc} is a weight between node $u \in V$ and community $c \in C$. Given F, the* BIGCLAM *generates a graph $G(V, E)$ by creating edge (u, v) between a pair of nodes $u, v \in V$ with probability $p(u, v)$:*

$$p(u, v) = 1 - \exp(-F_u \cdot F_v^T), \tag{1}$$

where F_u is a weight vector for node u ($F_u = F_{u\cdot}$).

Figure: from Yang and Leskovec. WSDM. 2013.

Edge Probability vs Number of Shared Memberships

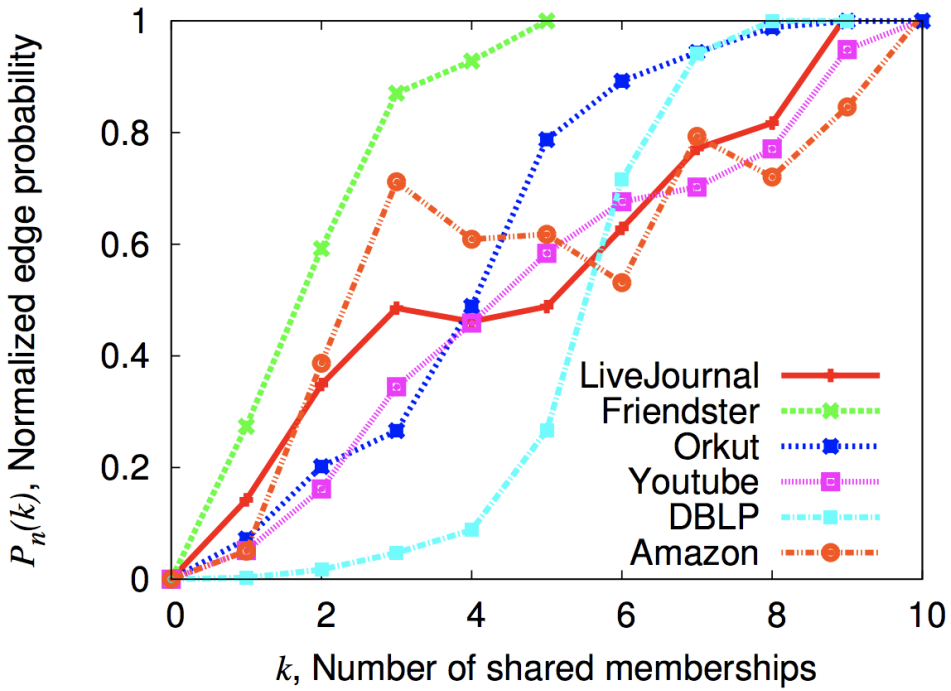


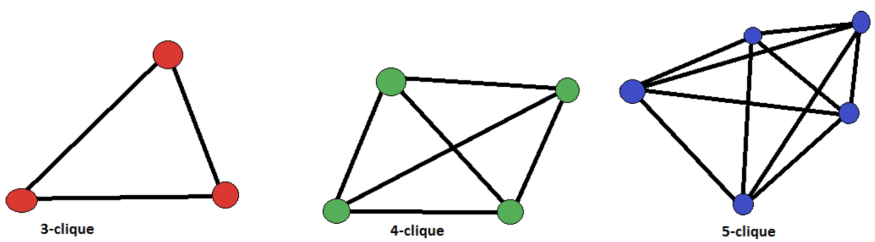
Figure: from Yang and Leskovec. WSDM. 2013.

The Future of Graph Partitioning

- I think overall, this is a pretty solved problem.
- Graph embedding people are still playing, but the partitions of nodes don't look super different than Louvain....
- In one new direction, we want to prune away some of the graph and explore structure without dealing with the whole graph.
- We can also think about defining communities based on motifs, or higher-order edge connections

Higher Order Idea One: Counting Cliques

- **Definition:** A k-clique is set, S, of k nodes, such that all pairs in S are connected by an edge.
- In some applications, you might want to study cliques. Or maybe you want to study the behavior of cliques as you make decisions in constructing your graph.
- Maybe it is easy to count the number of triangles in the graph but as k gets larger, this can become quite difficult
- If this is interesting to you, there is some very nice theoretical work for approximating the number of cliques in a graph `https://arxiv.org/pdf/1611.05561.pdf`



For those who count cliques

The Turan Shadow algorithm is implemented in Julia
`https://github.com/nassarhuda/TuranShadow.jl`

Graph Representation Learning

- There is a lot of recent hype in the graph world about feature learning from graphs or learning a representation for each node in a continuous vector space
- For a particular node, u, the objective is to learn some representation in d dimensions, $f(u)$, with $f(u) \in \mathbb{R}^d$

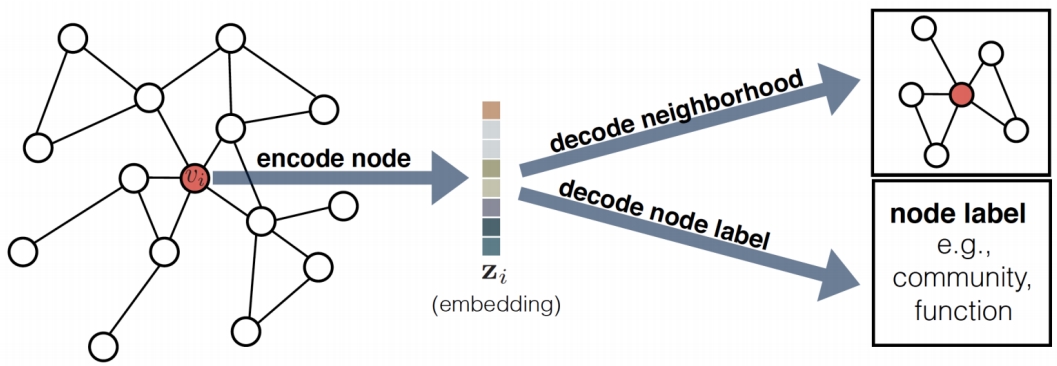


Figure: from Hamilton *et al.* ArXiv. 2018

Get Graph Partitioning for Free from Embedding

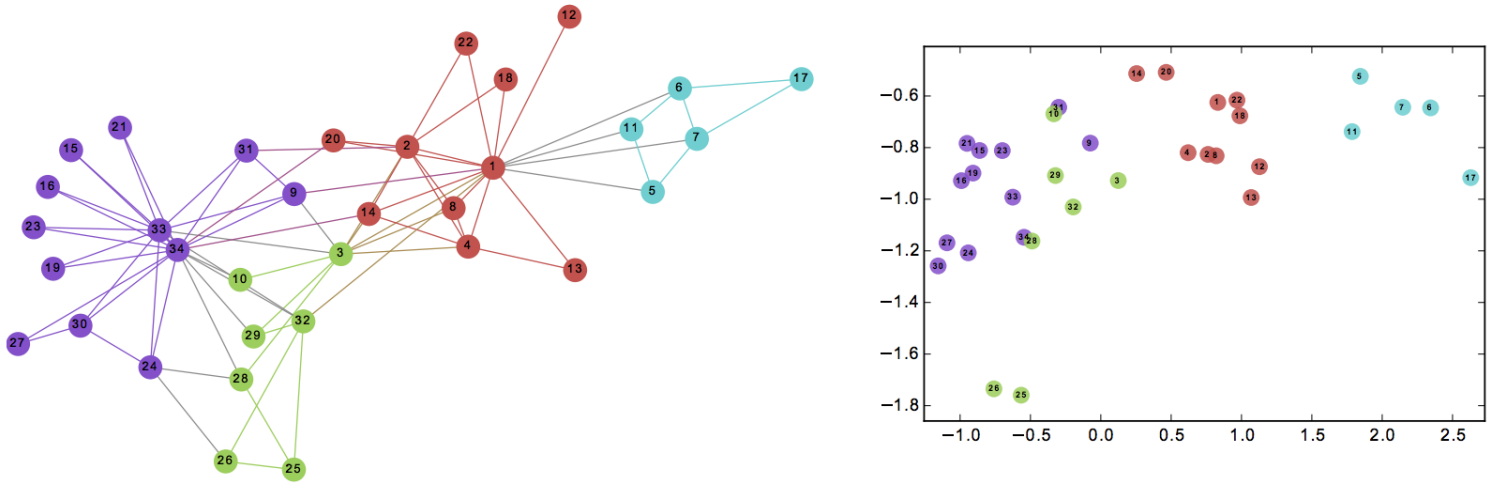
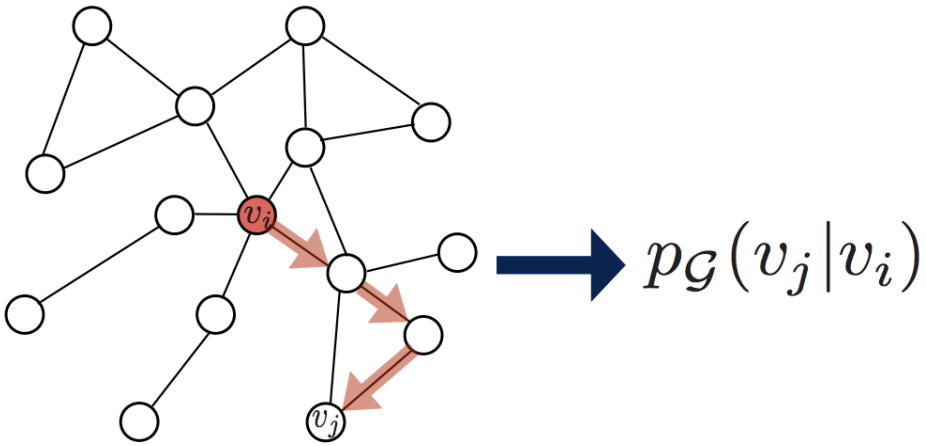
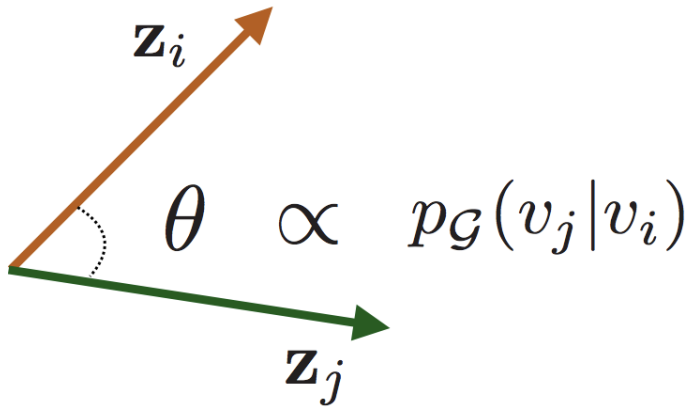


Figure: from Perozzi *et al.* KDD. 2014

The Random Walk Based Approach



1. Run random walks to obtain co-occurrence statistics.



2. Optimize embeddings based on co-occurrence statistics.

Figure: from Hamilton *et al.* ArXiv. 2018

Encoding the 'Role' of a Node with Node2Vec

- Node2Vec allows for control of whether the embedding captures between-node structural similarity (neighborhoods) or role (the types of nodes in connection with)

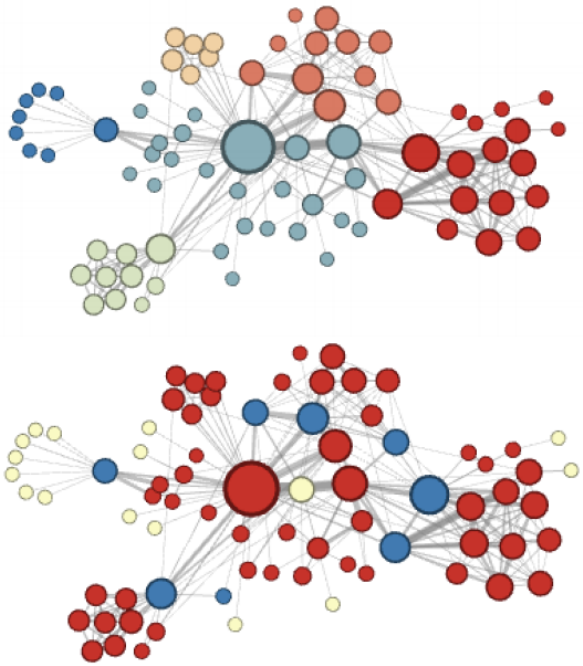


Figure: from Grover and Leskovec. KDD. 2016

Other Similar Approaches

- LINE → `https://arxiv.org/abs/1503.03578`
- DeepWalk → `https://arxiv.org/abs/1403.6652`
- Node2Vec → `https://arxiv.org/abs/1607.00653`
- Splitter → `https://arxiv.org/pdf/1905.02138.pdf`

Flexible Notion of Neighborhood (Breadth First)

When defining a neighborhood set N_s of some node u, there are two possible strategies. The following example assumes sampling $k = 3$ nodes.

- **Breadth-first Sampling:** Find the most immediate neighbors of node u and sequentially sample additional nodes according to their distance from u. In the image below, if we are selecting k neighbors, this selects s_1, s_2, s_3.

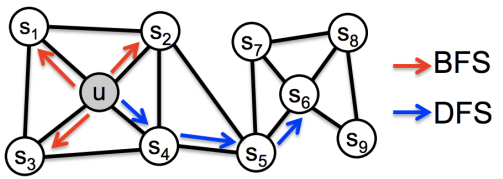


Figure 1: BFS and DFS search strategies from node u ($k = 3$).

Figure: from Leskovec and Grover. KDD. 2016

Flexible Notion of Neighborhood (Depth First)

- **Depth-first Sampling:** Select nodes sampled at sequentially further distances from the source node, u. According to the example below, nodes s_4, s_5, s_6 are selected.

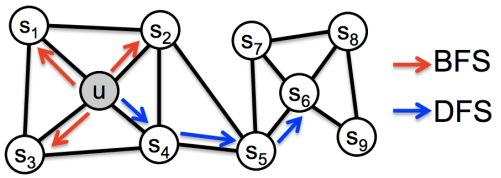


Figure 1: BFS and DFS search strategies from node u ($k = 3$).

Figure: from Leskovec and Grover. KDD. 2016

Node2Vec Objective

Starting with a graph, G with $G = (V, E)$, then the task if to learn a mapping function, f with $f : V \rightarrow \mathbb{R}^d$. Here, d is the number of dimensions of of f. Ultimately, f is a matrix of size $|V| \times d$. $\mathcal{N}_s(u)$ is further defined as the nodes in the neighborhood of node u. Then the objective is to find an f such that,

$$\sum_{u \in V} \log P(\mathcal{N}_s(u) \mid f(u)) \tag{5}$$

is as large as possible. Here, $\mathcal{N}_s(u)$ is conditioned on the feature representation of u, $f(u)$.

Conditional Independence Assumption

$$P(\mathcal{N}_S(u)|f(u)) = \prod_{n_i \in \mathcal{N}_s(u)} P(n_i \mid f(u)). \tag{6}$$

Conditional Likelihood for each Source, Neighborhood Pair

To model each source, neighborhood pair and given the conditional independence assumption introduced on the previous slide,

$$P(n_i \mid f(u)) = \frac{\exp(f(n_i) \cdot f(u))}{\sum_{v \in V} \exp(f(v) \cdot f(u))} \tag{7}$$

Varying Neighborhood According to DFS or BFS

As we have pointed out, $\mathcal{N}_S(u)$ is not restricted to the immediate neighbors of u. The authors define a flexible notion of a random walk on the graph that allows for sampling in either a DFS or BFS way. Assuming the random walk has just traversed edge (t, v) and now resides at node v. The transition probability from v to x (π_{vx}) can be written as,

$$\pi_{vx} = \alpha_{pq}(t, x) \cdot w_{vx} \tag{8}$$

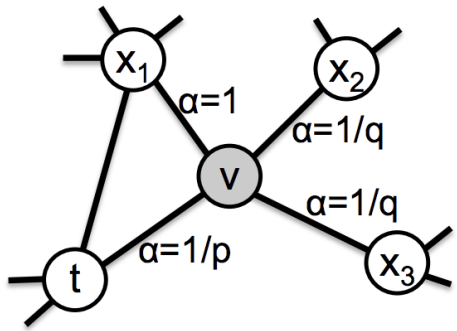


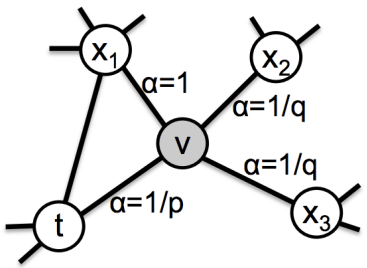
Figure: from Grover and Leskovec, KDD. 2016

Varying Neighborhood According to DFS or BFS

$$\pi_{vx} = \alpha_{pq}(t, x) \cdot w_{vx} \tag{9}$$

Here w_{vx} is the edge weight between nodes v and x or simply $\{0, 1\}$ if the graph is unweighted. d_{tx} denotes shortest path distance between t and x (we are at v). A user can specify p or q.

$$\alpha_{p,q}(t, x) = \begin{cases} 1/p & \text{if} d_{tx} = 0 \\ 1 & \text{if} d_{tx} = 1 \\ 1/q & \text{if} d_{tx} = 2 \end{cases}$$



Intuition Behind p

- p controls the likelihood of immediately revisiting a node in the walk.
  - A **low** value of p would keep the walk local, close to the starting node, u. A low value of p would encourage local walks next to a source node, u.

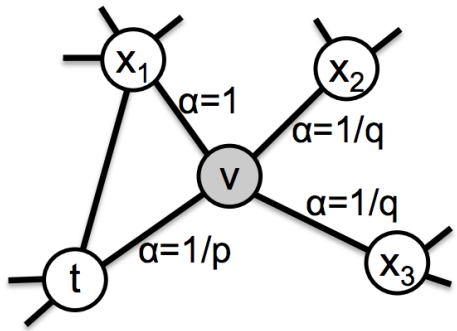


Figure: from Grover and Leskovec, KDD. 2016

Intuition Behind q

item q allows for differentiation between inward and outward nodes

- For $q < 1$, the walk is inclined to visit nodes that are further from node t. (Depth First)

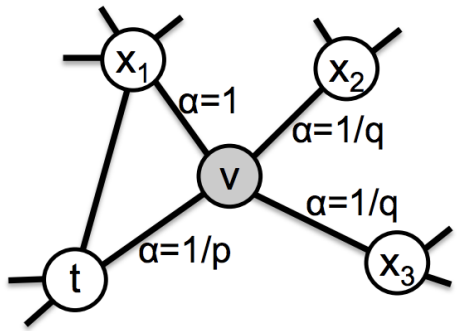
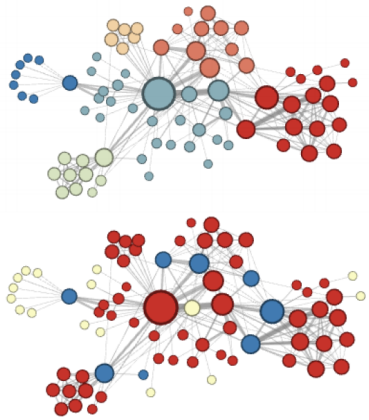


Figure: from Grover and Leskovec, KDD. 2016

Allow for Variation in What Defines Nodes as Being Similar

- **Application of Capturing Local Similarities:** Identify proteins involved in a common pathway. Or parts of the brain that are correlated to each other.
- **Usefulness of Capturing More Global Similarities:** Identify proteins that are regulators of many other proteins but are not related extensively among them selves.
- Validating a graph representation (**project idea!!!**).



Embedding as a Tool for a Variety of Tasks on Graphs

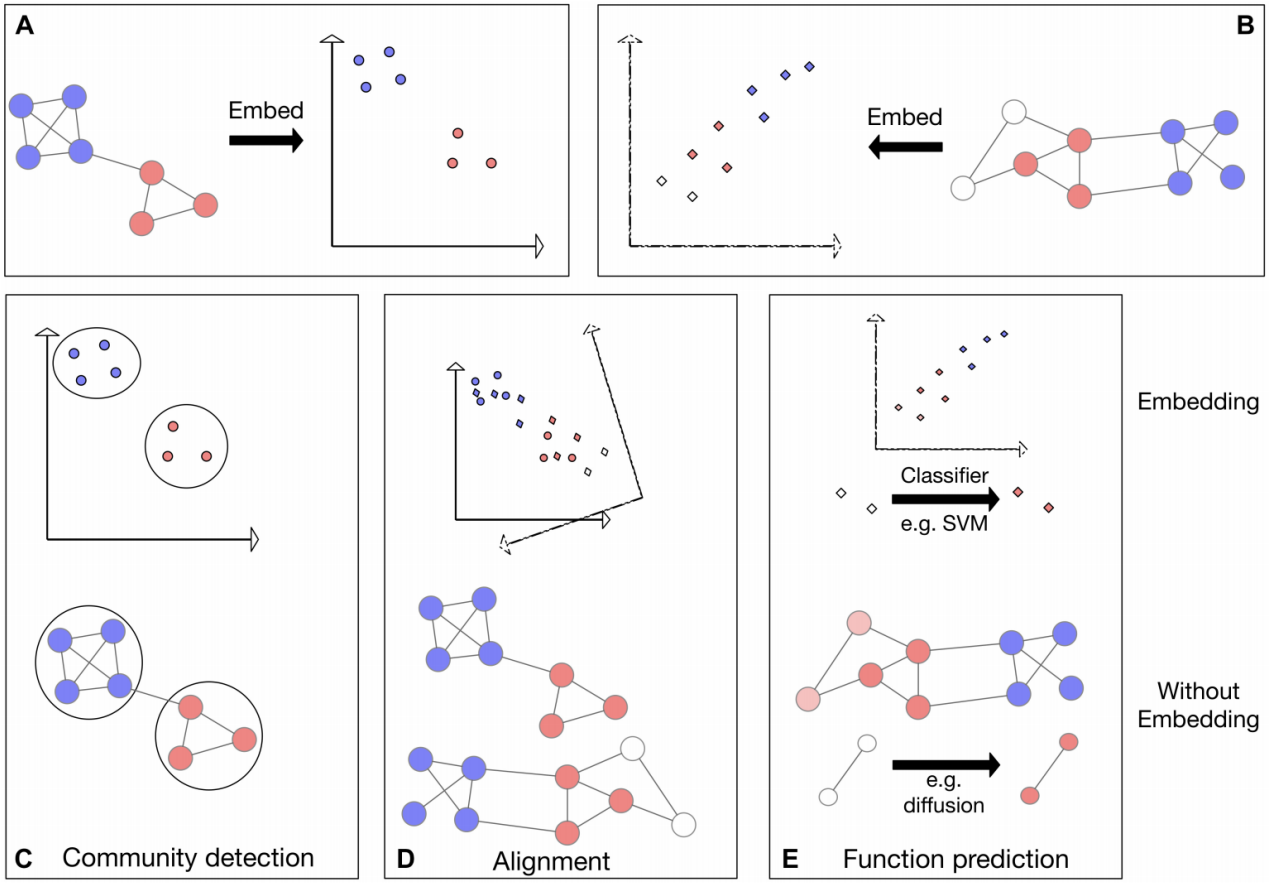


Figure: from Nelson *et al.* Frontiers in Genetics. 2019. Protein families (C). Cross-Species Alignment (D), Function Prediction (E)

Benchmarking Tasks of Working Directly on the Network vs on an Embedded Representation

TABLE 1 | Comparative analysis of direct vs. embedding methods across a range of problems in network biology.

	IsoRank ($\alpha = 0.5$)	**MuNK** ($\lambda = 0.05$)
A. Network alignment		
EC	**39.0%**	21.9%
GOC		
$K = 20$	**63.4%**	57.6%
$K = 50$	**20.7%**	17.9%
$K = 100$	**1.2%**	1.0%
GOC (AUPR)	0.721	**0.746**
Runtime	26 min 40 s (incl. grid search)	1 min 52 s (incl. alignment)
	densityCut ($K = 4$, $\alpha = 0.9$)	**Vicus** ($K = 10$, $\sigma = 0.5$)
B. Community detection		
Buettner ($C = 2$, $C_t = 11$)	0.256	**0.316**
Kolodziejczyk ($C = 5$, $C_t = 4$)	0.325	**0.552**
Pollen ($C = 13$, $C_t = 11$)	**0.931**	0.928
Usoskin ($C = 9$, $C_t = 4$)	0.373	**0.591**
Avg. Runtime	1 min 15 s (incl. parameter grid search)	<5 s

STRING v9.1 *Homo sapiens*. Included with the Mashup distribution.

Figure: from Nelson *et al.* Frontiers in Genetics. 2019

Recap of Why we Love Graphs in Biology

- Graphs are useful for representing relational information
  - Cells, Patients, etc.
- There are a lot of well-defined tasks on graphs that we can use to answer our questions
  - Graph partitioning
  - Diffusion for smoothing (which we will see later), and partitioning if we were to work with Graph Laplacian
  - Visualization- connecting visualization with the scientific problem
  - Other tasks not discussed: link prediction, collaborative filtering, label propagation.

Graph Signal Processing

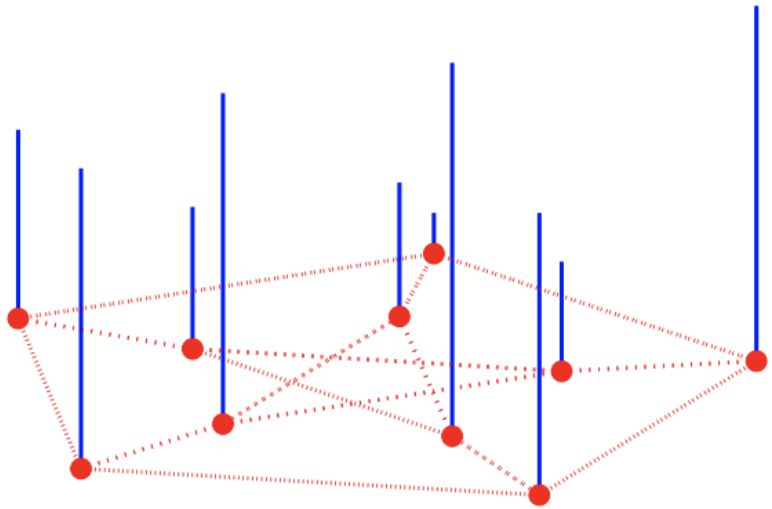
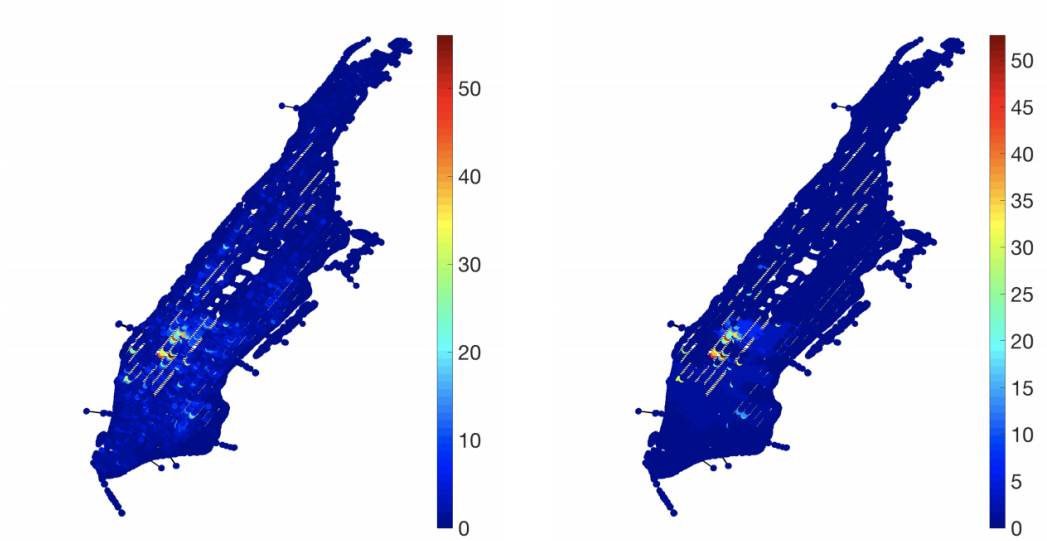


Figure: from Shuman *et al.* ArXiv. The purpose is to study the interplay between some signal and graph connectivity.

Piecewise Smooth Assumption

Translation: Nodes that are close (in terms of geodesic distance) on the graph should have similar signals. You can approximate the signal of a node, based on its neighbors.



(a) Actual data (b) Approximation

Figure: from `https://arxiv.org/abs/1712.00468`. Here the graph is street intersections in Manhattan and the signal is taxi pickups.

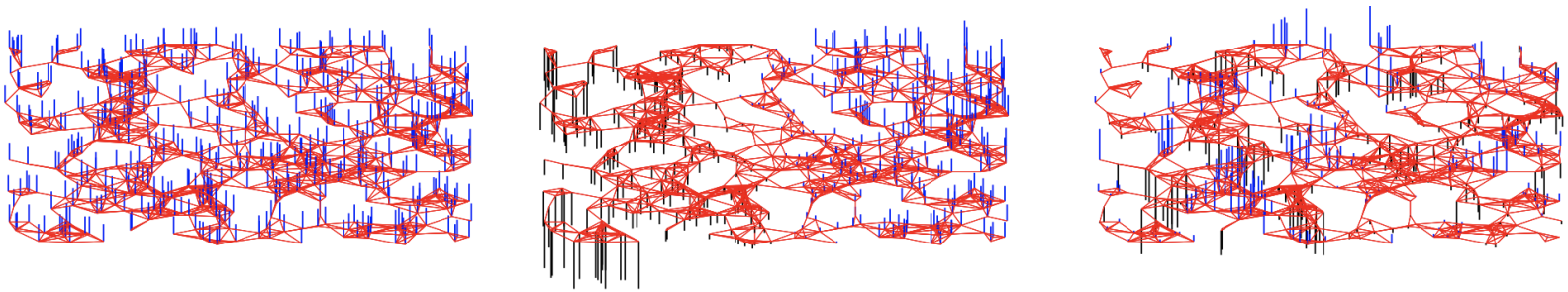
How Localized is the Signal?

Remember, our friend Graph Laplacian ($\mathbf{L} = \mathbf{D} - \mathbf{A}$),

- Some very nice theory falls out about the eigenvalues of the Laplacian matrix in terms of how 'localized' a graph signal, $\mathbf{f}$, is. For example $\mathbf{f}$ could be an expression of some protein.
  - First re-write $\mathbf{f}$ in terms of eigenvectors of the Laplacian
  - The eigenvectors corresponding to the first few eigenvalues of $\mathbf{L}$ are considered **low frequency**, and hence entries of the eigenvector entries corresponding to nodes that are connected should be similar
  - For higher **high frequencies** corresponding to 'later' eigenvalues, the values of the eigenvectors of adjacent nodes will be more different.

Signal Specificity

The mess of a signal on a graph can be interpreted as the number of connections between nodes with whose signals have different signs.



$\mathbf{u}_0$ $\mathbf{u}_1$ $\mathbf{u}_{50}$

Figure: from GSP Review https://arxiv.org/abs/1211.0053

Similarly

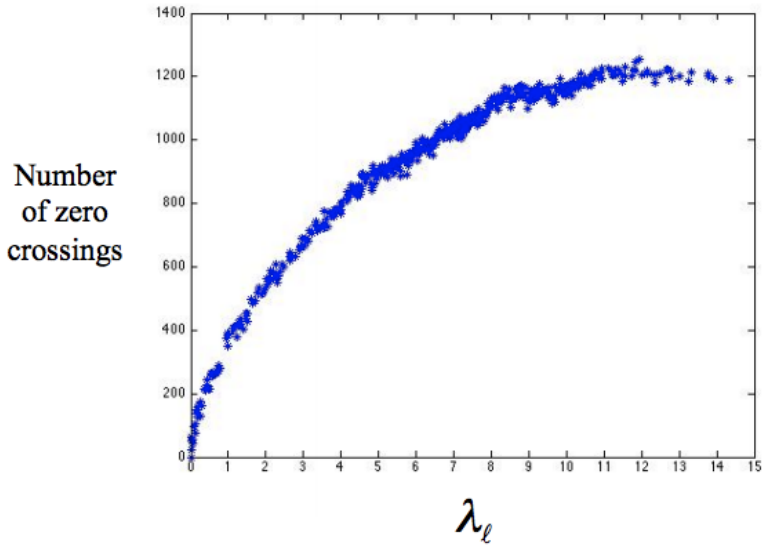


Figure: from GSP Review https://arxiv.org/abs/1211.0053

- You survived all of the graph and linear algebra stuff!
- Next time.... intro to single cell bioinformatics