CSE 427 Computational Biology

Lecture 8
Graph diffusion
1st-order proximity only models local structure

Q: Which node is topologically similar to the red node?

Direct neighbors: two nodes are considered similar if they share an edge
Neighbors' neighbors: two nodes are considered similar if they share many neighboring/adjacent nodes.
Key idea: high-order (nth-order) proximity models global structure

Q: Which node is topologically similar to the red node?

All nodes in the graph: two nodes are considered similar if their distances to all other nodes are similar.
The Math:
use diffusion model on each node to calculate topological similarity

Input: adjacency matrix of the hierarchy (undirected)

\[ S^{t+1} = (1 - p_r)S^t B + p_r I \]
The Math:
use diffusion model on each node to calculate topological similarity

Input: adjacency matrix of the hierarchy (undirected)

\[ S^{t+1} = (1 - p_r)S^tB + p_rI \]

Node \( i \) and node \( j \) are topologically similar if \( s_i \) is similar to \( s_j \)
**The Math:**

use diffusion model on each node to calculate topological similarity

Iterate $t = \{0, 1, \ldots \}$ until $S^{t+1} \approx S^t$.

**Output:** equilibrium distribution $s_i$ starting from node $i$

Node $i$ and node $j$ are topologically similar if $s_i$ is similar to $s_j$
The Math:
use diffusion model on each node to calculate topological similarity

Iterate $t = \{0, 1, \ldots\}$ until $S^{t+1} \approx S^t$.

Output: equilibrium distribution $s_i$ starting from node $i$

$$S^{t+1} = (1 - p_r)S^tB + p_rI$$

Input: adjacency matrix of the hierarchy (undirected)
Random walk to neighbors
Restart to the original node
Motivation

- Given a set of web pages with links between them, we would like to rank the pages in order of importance.
- We can model this as a graph problem where web pages are vertices and links are edges.

1. A link from an important edge is more significant than a link from an unimportant web page.

2. Being linked from a page with many outgoing links is less significant than being linked from a page with few outgoing
Random walk

Every node votes for its neighbors and gets votes from neighbors

- Each link’s vote is proportional to the importance of its source node
- If node $i$ with importance $r_i$ has $d_i$ out-links, each link gets $r_i / d_i$ votes
- Node $i$’s own importance $r_i$ is the sum of the votes on its in-links

$$r_i = \frac{r_2}{d_2} + \frac{r_3}{d_3} + \frac{r_4}{d_4}$$
Random walk

Every node votes for its neighbors and gets votes from neighbors

\[ r_i = \frac{r_2}{d_2} + \frac{r_3}{d_3} + \frac{r_4}{d_4} \]

How can you solve so many equations together?

\[ r_5 = \frac{r_i}{d_i} + \ldots \]

\[ r_6 = \frac{r_i}{d_i} + \ldots \]

\[ r_7 = \frac{r_i}{d_i} + \ldots \]
What is a Random Walk

Given a graph and a starting node, we select a neighbor of it at random, and move to this neighbor.
What is a Random Walk

We select a neighbor of it at random, and move to this neighbor.
What is a Random Walk

Then we select a neighbor of this node and move to it, and so on.
What is a Random Walk

The (random) sequence of nodes selected this way is a random walk on the graph.
A transition matrix is a stochastic matrix where each element $a_{ij}$ represents the probability of moving from $i$ to $j$, with each row summing to 1.
Markov chains

• A Markov chain describes a discrete time stochastic process over a set of states

\[ S = \{s_1, s_2, \ldots, s_n\} \]

according to a transition probability matrix

\[ P = \{P_{ij}\} \]

\[ P_{ij} = \text{probability of moving to state } j \text{ when at state } i \]

• Markov Chains are memoryless: The next state of the chain depends only at the current state
Stationary Distribution

- $x_t(i) = \text{probability that the surfer is at node } i \text{ at time } t$
- $x_{t+1}(j) = \sum_i x_t(i) \cdot P_{ij}$
- $x_{t+1} = x_t P = x_{t-1} P P = x_0 P^t$

- What happens when the surfer keeps walking for a long time?
  - We get a stationary distribution
Stationary Distribution

- The stationary distribution at a node is related to the amount of time a random walker spends visiting that node.
- When the surfer keeps walking for a long time, the distribution does not change any more: $x_{t+1}(i) = x_t(i)$.
- For “well-behaved” graphs this does not depend on the start distribution.

Source: Ahmed Hassan Random Walks on Graphs Classification, Clustering, and Ranking
Hitting Time

• How long does it take to hit node $b$ in a random walk starting at node $a$?

• Hitting time from node $i$ to node $j$
  • Expected number of hops to hit node $j$ starting at node $i$.
  • Not symmetric
  • $h(i,j) = 1 + \sum_{k \in \text{adj}(i)} P(i,k) \cdot h(k,j)$
Commuting Time

- How long does it take to hit node $b$ in a random walk starting at node $a$ and come back to $a$?

- Commute time from node $i$ to node $j$
  - Expected number of hops to hit node $j$ starting at node $i$ and come back to $i$.
  - Symmetric
  - $c(i, j) = h(i, j) + h(j, i)$
• Random walk
  • Starting from different node results in the same probability distribution
• Random walk with restart
  • Starting from different node results in different probability distribution

Iterate $t = \{0, 1, \ldots\}$ until $S^{t+1} \approx S^t$.

Output: equilibrium distribution $s_i$ starting from node $i$

$$S^{t+1} = (1 - p_r)S^tB + p_rI$$

Input: adjacency matrix of the hierarchy (undirected)

Random walk to neighbors

Restart to the original node

Source: Network-based stratification of tumor mutations
Random Walk

Then we select a neighbor of this node and move to it, and so on.
Random walk

The flow equations can be written:

\[ r = Mr \]

\[ \begin{array}{c}
1/d_1 & 0 & \cdots & 1/d_N \\
1/d_1 & 1/d_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 1/d_2 & \cdots & 1/d_N \\
\end{array} \]

\[ \begin{array}{c}
r_1 \\
r_2 \\
\vdots \\
r_j \\
r_N \\
\end{array} \]

\[ \begin{array}{c}
r_1 \\
r_2 \\
\vdots \\
r_j \\
r_N \\
\end{array} \]

\[ M \] is a Markov matrix since each column sums equal to 1
How to solve this?

Power Iteration method

Initialize: \( r^0 = \left[ \frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N} \right]^T \)

While \( ||r^{k+1} - r^k||_2 > 0.0001 \):

\[ r^{k+1} = Mr^k \]
Random walk interpretation

The vector \( \mathbf{r} \) can be reinterpreted as a probability vector to visit each website

- Imagine a random web surfer
  - At any time \( k \), surfer has a probability vector \( \mathbf{r}^k \) to visit a web page following the out-link.
  - Process repeats indefinitely

\[
\begin{align*}
\mathbf{r}_1 &= 0, \frac{1}{d_2}, \ldots, \frac{1}{d_N} \\
\mathbf{r}_2 &= \frac{1}{d_1}, 0, \ldots, 0 \\
\vdots &= \vdots \\
\mathbf{r}_j &= \vdots, \vdots, \frac{1}{d_1}, \frac{1}{d_2}, \ldots, \frac{1}{d_N} \\
\vdots &= \vdots \\
\mathbf{r}_N &= \vdots, \vdots, \vdots, \frac{1}{d_1}, \frac{1}{d_2}, \ldots, \frac{1}{d_N}
\end{align*}
\]

\[ \mathbf{r} = \mathbf{M} \mathbf{r} \]
Problem of random walk

Dead ends

\[ r_j^{(t+1)} = \sum_{i \rightarrow j} \frac{r_i^{(t)}}{d_i} \]

Dead-ends are a problem because the matrix is not column stochastic so our initial assumptions are not met.

**Example:**

<table>
<thead>
<tr>
<th>Iteration:</th>
<th>0,</th>
<th>1,</th>
<th>2,</th>
<th>3...</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r_a )</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( r_b )</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Random walk has stationary distribution when the graph is irreducible and aperiodic.

- **Irreducible**: There is a path from every node to every other node.
Aperiodic: The GCD of all cycle lengths is 1. The GCD is also called period.

Periodicity is 3

Aperiodic

The greatest common divisor of a set of whole numbers is the largest integer which divides them all.

Example: The greatest common divisor of 12 and 15.
\[ \text{gcd}(12, 15). \]

Divisors of 12: 1, 2, 3, 4, 6, 12.
Divisors of 15: 1, 3, 5, 15.
Common divisors: 1, 3.
Greatest common divisor is 3.
\[ \therefore \text{gcd}(12, 15) = 3. \]
Solution: jump to a random node

At each time step, the random surfer has two options

- With prob. $\beta$, follow a link at random
- With prob. $1 - \beta$, jump to a random page
- Common values for $\beta$ are in the range 0.8 to 0.9

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1 - \beta) \frac{1}{N}$$
Difference from random walk

Random walk

\[ \begin{align*}
   r_1 & = \beta \left( \frac{1}{d_1}, 0, \ldots, \frac{1}{d_1}, \frac{1}{d_2}, \ldots, 0, \frac{1}{d_2}, \ldots \right) \\
   r_2 & \vdots \vdots \vdots \vdots \vdots \\
   r_j & \vdots \vdots \vdots \vdots \vdots
\end{align*} \]

\[ + (1 - \beta) \left( \frac{1}{N}, \frac{1}{N}, \ldots, \frac{1}{N} \right) \]

Random walk with restart

\[ \begin{align*}
   r_1 & = \beta \left( \frac{1}{d_1}, 0, \ldots, \frac{1}{d_1}, \frac{1}{d_2}, \ldots, 0, \frac{1}{d_2}, \ldots \right) \\
   r_2 & \vdots \vdots \vdots \vdots \vdots \\
   r_j & \vdots \vdots \vdots \vdots \vdots
\end{align*} \]

\[ + (1 - \beta) \left( c_1, c_2, \ldots, c_j, 0, 0, \ldots \right) \]
Matrix representation v.s. vector representation

Iterate $t = \{0, 1, \ldots\}$ until $S^{t+1} \approx S^t$.

**Output:** equilibrium distribution $s_i$ starting from node $i$

$S^{t+1} = (1 - p_r)S^tB + p_rI$

Input: adjacency matrix of the hierarchy (undirected)

Node $i$ and node $j$ are topologically similar if $s_i$ is similar to $s_j$.

Random walk to neighbors

Restart to the original node
Random walk with restart

• Random walk with restart is the same as random walk other than the fact that jumps are back to one of a given set of starting vertices.

• In a way, the walk in Random walk with restart is biased towards (or personalized for) this set of starting vertices and is more localized compared to the random walk.
Functions of random walk

1. Smooth the whole graphs

2. Assign importance score

3. Quantify the distance of two nodes

4. Want to integrate information beyond the neighbors
Guilt-by-association rule

- Assign a label to a node using its neighbor’s labels

Source: Aging research in the post-genome era: New technologies for an old problem
Guilt-by-association rule

- Fast and scalable to large networks
- But only utilize information of 1st-order neighbors

Source: Broad network-based predictability of Saccharomyces cerevisiae gene loss-of-function phenotypes
Drug target identification using guilt-by-association

Guilt-by-association rule

Source: Drug repositioning by integrating target information through a heterogeneous network model
Network embedding and graph neural network
SimCLR: Contrastive Learning using data augmentation

(a) Original  (b) Crop and resize  (c) Crop, resize (and flip)  (d) Color distort. (drop)  (e) Color distort. (jitter)

(f) Rotate \{90°, 180°, 270°\}  (g) Cutout  (h) Gaussian noise  (i) Gaussian blur  (j) Sobel filtering
Using network is like data augmentation
The Math:
use diffusion model on each node to calculate topological similarity

Input: adjacency matrix of the hierarchy (undirected)

\[ S^{t+1} = (1 - p_r)S^tB + p_rI \]
The Math:
use diffusion model on each node to calculate topological similarity

Input: adjacency matrix of the hierarchy (undirected)

Node $i$ and node $j$ are topologically similar if $s_i$ is similar to $s_j$

$S^{t+1} = (1 - p_r)S^tB + p_rI$

Random walk to neighbors

Restart to the original node
The Math:
use diffusion model on each node to calculate topological similarity

Iterate $t = \{0, 1, \ldots\}$ until $S^{t+1} \approx S^t$.

Output: equilibrium distribution $s_i$ starting from node $i$

\[ S^{t+1} = (1 - p_r)S^tB + p_rI \]

Node $i$ and node $j$ are topologically similar if $s_i$ is similar to $s_j$
The Math:
use diffusion model on each node to calculate topological similarity

Iterate $t = \{0, 1, \ldots\}$ until $S^{t+1} \approx S^t$.

Output: equilibrium distribution $s_i$ starting from node $i$

$$S^{t+1} = (1 - p_r)S^t B + p_r I$$

Input: adjacency matrix of the hierarchy (undirected)

Random walk to neighbors

Restart to the original node
From advanced matrix to random walk probability matrix

Adjacency matrix

\[
\begin{pmatrix}
1 & 1 & 0 & \cdots & 1 & 0 \\
1 & 0 & 1 & \cdots & 1 & 0 \\
0 & 1 & 0 & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
1 & 1 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 \\
\end{pmatrix}
\]

| V |

RWR

Probability similarity matrix \((\text{sim}_G)\)

\[
\begin{pmatrix}
0.7 & 0.15 & 0.02 & \cdots & 0.1 & 0.01 \\
0.09 & 0.7 & 0.05 & \cdots & 0.12 & 0.03 \\
0.01 & 0.11 & 0.7 & \cdots & 0.05 & 0.04 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0.05 & 0.02 & 0.08 & \cdots & 0.7 & 0.01 \\
0.02 & 0.04 & 0.01 & \cdots & 0.01 & 0.7 \\
\end{pmatrix}
\]

| V |

Embeddings

\[
\begin{pmatrix}
1.25 & 0.35 & 0.27 \\
5.2 & 1.6 & 0.7 \\
1.1 & 3.67 & 4.7 \\
\vdots & \vdots & \vdots \\
2.22 & 0.8 & 1.4 \\
4.1 & 0.78 & 3.51 \\
\end{pmatrix}
\]

d < |V|

Image adapted from MultiVERSE: a multiplex and multiplex-heterogeneous network embedding approach
Network embedding: decomposing diffusion matrix

**Optimization goal:** find class embedding \( \{x_i\} \) and context embedding \( \{u_i\} \) so that \( \{\hat{s}_i\} \) is close to \( \{s_i\} \).

\[
\hat{s}_{ij} = \frac{e^{x_i^T u_j}}{\sum_k e^{x_i^T u_k}} \approx e^{x_i^T u_j}
\]

Drop the partition function for fast approximation.

Loss function:

\[
\min_{x,u} C(S, \hat{S}) = \sum_{i=1}^{n} \sum_{j=1}^{n} (x_i^T u_j - \log s_{ij})^2
\]
Compact Integration of Multi-Network Topology for Functional Analysis of Genes
Recap

- Guilt-by-association
  - Only use 1st-order neighbors’ information
  - A good baseline that works very well in many applications.
- Random walk
  - Find the most important node. Consider all nodes in the graph.
  - Might not converge due to dead end issue
- Random walk with restart
  - Solved dead end issue.
  - Does not have node features. Only one importance score for each node.
- Network embedding
  - Reduce RWR matrix to low-dimensional
  - Better for large and noisy network
Goal

• Tumor stratification: to divide a heterogeneous population into clinically and biologically meaningful subtypes based on molecular profiles
Somatic mutation profile

- Compare the mutations of tumors
- Sparse

Source: Network-based stratification of tumor mutations
Overview of network-based stratification

Source: Network-based stratification of tumor mutations
Performing random walk with restart for each patient

\[ F_{t+1} = \alpha F_t A + (1 - \alpha) F_0 \]

The new mutation status for patient 1, gene j
First consider the mutations for all the genes of patient 1
Then consider all the neighbors of gene j
The original patient mutation matrix

Source: Network-based stratification of tumor mutations
The Math:
use diffusion model on each node to calculate topological similarity

Iterate $t = \{0,1,...\}$ until $S^{t+1} \approx S^t$.

**Output:** equilibrium distribution $s_i$ starting from node $i$

$$S^{t+1} = (1 - p_r)S^t B + p_r I$$
Network smoothing

\[ F_{t+1} = \alpha F_t A + (1-\alpha)F_0 \]

\( F_0 \): patients * genes matrix

\( A \): adjacency matrix of the gene interaction network (STRING, HumanNet and PathwayCommons)

\( \alpha \): tuning factor that determines how far a mutation signal can diffuse

Source: Network-based stratification of tumor mutations
**C** Network NMF: \[
\min_{W,H>0} \|F - WH\|_F + \gamma \|W^1L\|_F
\]

\[
F = \text{genes (patients x genes)} \text{ post-smoothing matrix}
\]

\[
W = \text{cluster prototypes}
\]

\[
H = \text{cluster assignments}
\]

\[
L = \text{network influence constraint}
\]

**d** Network-based stratification
**Biological significance**

Different clusters of patients have different survival rates!

Source: Network-based stratification of tumor mutations
How to incorporate node features into RWR

• The restart probability on each node is a vector now
• \( F_{t+1} = \alpha F_t A + (1-\alpha)F_0 \)

\( F_0 \): patients * genes matrix

\( A \): normalized adjacency matrix of the gene interaction network

\( \alpha \): restart probability

Question: what is the limitation of incorporating node features in this way?
How to incorporate node features into RWR

• The restart probability on each node is a vector now
• $F_{t+1} = \alpha F_t A + (1 - \alpha) F_0$

$F_0$: patients * genes matrix
A: normalized adjacency matrix of the gene interaction network
$\alpha$: restart probability

Question: what is the limitation of incorporating node features in this way?
Different node features are modeled independently.
This is why we need graph neural network which will model the dependency between features
Deep Learning on Graphs

Random value

Labels
Deep Learning on Graphs

- Each node has its own features
- The embedding of each node is a function of its own features and its neighbors’ features.
Stack multiple layers

For the second layer, the feature is the hidden output by the first layer
One common mistake: Graphs are not Images

Adjacent matrix of a Graph

Images

Label
But Images can be modeled as Graphs