THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學

AMA 2233

Data Analytics and Visualization

Yancheng Yuan

Department of Data Science and Artificial Intelligence

Recap of last lecture

The Page Rank Algorithm

Notation:
Denote $\boldsymbol{x_i}$ as the number of backlinks for page i.
Denote $\boldsymbol{n_j}$ as the number of outgoing links from page j.
Denote $\boldsymbol{L_k}$ as the set of pages with a link to page k.

Score calculation: In the page rank algorithm, if page j contains $\boldsymbol{n_j}$ outgoing links, one of which links to page k, then we will boost page k's score by $\frac{x_j}{n_j}$. Therefore,

$$x_k = \sum_{j \in L_k} \frac{x_j}{n_j}$$

Page Rank Algorithm

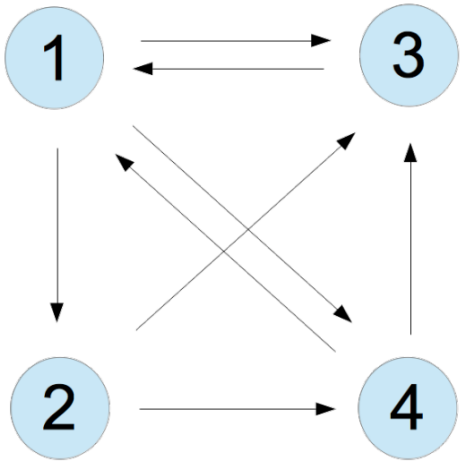
Key equation: Assume there are n pages in total. Denote $x = [x_1, \cdots, x_n]^T$. Then

$$x = Ax,$$

where

$$A_{ij} = \begin{cases} \frac{1}{n_j}, & \text{if } j \in L_i, \\ 0, & \text{otherwise.} \end{cases}$$

Example



$$A = \begin{pmatrix} 0 & 0 & 1 & \frac{1}{2} \\ \frac{1}{3} & 0 & 0 & 0 \\ \frac{1}{3} & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{3} & \frac{1}{2} & 0 & 0 \end{pmatrix}$$

Damping Factor

Damping factor: The PageRank theory holds that an imaginary surfer who is randomly clicking on links will eventually stop clicking. The probability, at any step, that the person will continue following links is a damping factor d. The probability that they instead jump to any random page is 1 - d.

$$x_k = \frac{1-d}{n} + d \sum_{j \in L_k} \frac{x_j}{n_j}$$

In other words, x is the solution to the following linear system

$$x = \frac{1-d}{n} * 1_{\mathrm{n}} + \mathrm{d} * \mathrm{Ax}.$$

A motivation question

Question: How to cluster the data points with only the similarity matrix (also called affinity matrix) A?

A commonly used similarity matrix for a given collection of data $\{x_1, \ldots, x_n\}$ is defined by

$$A_{ij} = \begin{cases} exp(-\|x_i - x_j\|^2/2\sigma^2), & \text{if } i \neq j, \\ 0 & \text{if } i = j. \end{cases}$$

The Laplacian Matrix

Laplacian Matrix: The Laplacian matrix with respect to the similarity matrix A is defined by

$$L = D - A,$$

where D is a diagonal matrix defined by

$$D_{ii} = \sum_j A_{ij}.$$

An example: Consider three data points $x_1 = (0.54, 1.83), x_2 = (-2.26, 0.86), x_3 = (0.32, -1.31)$ and $\sigma = 1$. We can form

$$A = \begin{pmatrix} 0 & 0.013 & 0.007 \\ 0.013 & 0 & 0.003 \\ 0.007 & 0.003 & 0 \end{pmatrix} \qquad L = \begin{pmatrix} 0.02 & -0.013 & -0.007 \\ -0.013 & 0.016 & -0.003 \\ -0.007 & -0.003 & 0.01 \end{pmatrix}$$

Normalized Laplacian Matrix

Normalized Laplacian Matrix: The normalized Laplacian matrix with respect to the similarity matrix A is defined by

$$L^{norm} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}.$$

An example: Consider three data points $x_1 = (0.54, 1.83), x_2 = (-2.26, 0.86), x_3 = (0.32, -1.31)$ and $\sigma = 1$. We can form

$$A = \begin{pmatrix} 0 & 0.013 & 0.007 \\ 0.013 & 0 & 0.003 \\ 0.007 & 0.003 & 0 \end{pmatrix} \qquad L^{\text{norm}} = \begin{pmatrix} 1 & -0.727 & -0.495 \\ -0.727 & 1 & -0.237 \\ -0.495 & -0.237 & 1 \end{pmatrix}$$

Spectral Clustering algorithm

Input: Similarity matrix A, number of clusters K.

Step 1: Compute the noralized Laplacian matrix L^{norm}.

Step 2: Calculate the eigenvectors corresponding to the smallest K eigenvalues of L^{norm}. Form it as $V = [v_1, v_2, \dots, v_K]^T$.

Step 3: Run K-means algorithm on the columns of V.

Agenda

1. Cut of a graph
2. Another interpretation of the Laplacian Matrix
3. Theoretical Understanding of Spectral Clustering
4. Applications in Community Detection

Weighted Graph and Cut

Weighted Graph

Weighted Graph: In practice, we can assign a weight for each edge of the graph, which formed a (undirected) weighted graph. In addition to the vertices V and edge set E, we will introduce a weight matrix W to represent the weight information. In other words, we will represent a weighted graph as G = (V, E, W).

The weight matrix W is defined as

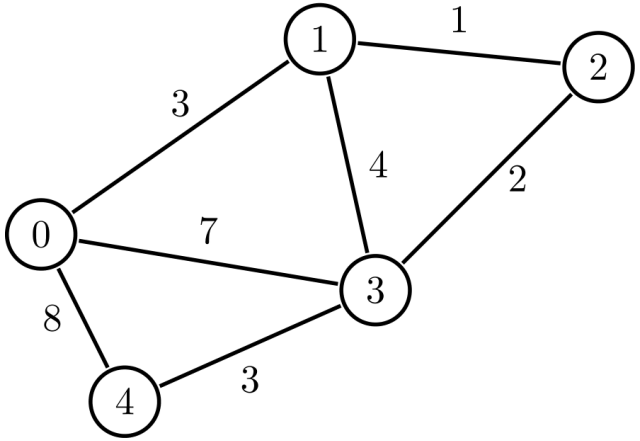
$$W_{ij} = \begin{cases} w_{ij} & \text{if } (i,j) \in E, \\ 0 & \text{otherwise.} \end{cases}$$

Remark: 1. For a undirected graph, W is a symmetric matrix. In other words, we have

$$W_{ij} = W_{ji}$$

2. For an unweighted graph, W is just the adjacency matrix A.

Example



$$W = \begin{pmatrix} 0 & 3 & 0 & 7 & 8 \\ 3 & 0 & 1 & 4 & 0 \\ 0 & 1 & 0 & 2 & 0 \\ 7 & 4 & 2 & 0 & 3 \\ 8 & 0 & 0 & 3 & 0 \end{pmatrix}$$

Cut of a Graph

Cut of a Graph: Let G = (V, E, W) be a given graph and let (S, S^c) be the partition of the vertices of the graph. Then, the cut of the partition (S, S^c) is defined as

$$Cut(S) = \sum_{i \in S} \sum_{j \in S^c} W_{ij}$$

The cut of a graph is a good measure for a clustering (two clusters) of the vertices of the graph G. In particular, we want to find a partition (S, S^c) of the vertices V which keeps as many of the edges, or connections, within the clusters and has as few edges as possible across clusters.

Compute the cut via Laplacian Matrix

Let G = (V, E, W) be a given graph with n vertices and let (S, S^c) be the partition of the vertices of the graph. Define a vector $y \in R^n$ as

$$y_i = \begin{cases} 1 & \text{if } i \in S, \\ -1 & \text{if } i \in S^c. \end{cases}$$

Then, we have

$$Cut(S) = \frac{1}{4} \sum_{i<j} W_{ij} \left(y_i - y_j\right)^2$$

Another interpretation of the Laplacian Matrix

Compute the cut via Laplacian Matrix

Let G = (V, E, W) be a given graph with n vertices, the Laplacian matrix of G is defined as

$$L_G = D - W,$$

where D is a diagonal matrix with

$$D_{ii} = \sum_{j=1}^{n} W_{ij}.$$

Theorem Let G = (V, E, W) be a given graph with n vertices and L_G is its Laplacian matrix. Let $x \in R^n$ be any given vector, then

$$x^T L_G x = \sum_{i<j} W_{ij}(x_i - x_j)^2$$

Therefore,

$$Cut(S) = \frac{1}{4}\, \mathrm{y}^{\mathrm{T}} \mathrm{L_G y}$$

Model Two Clusters Clustering Problem as a Balanced Minimum Cut

Let G = (V, E, W) be a given graph with n vertices, we can model the clustering of the vertices V of the graph as a balanced minimum cut problem

$$\min_{S \subseteq V, |S| = |S^c|} Cut(S) = \min_{y \in \{-1,1\}^n, 1^T y = 0} \frac{1}{4} y^T L_G y$$

Relaxed balanced minimum cut problem:

$$\min_{||y|| = \sqrt{n}, 1^T y = 0} \frac{1}{4} y^T L_G y$$

If we take $z = y/\sqrt{n}$, we can write the above problem as

$$\min_{||z|| = 1, 1^T z = 0} \frac{n}{4} z^T L_G z$$

Understanding Spectral Clustering

THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學

Property of the Laplacian Matrix

Theorem Let G = (V, E, W) be a given graph with n vertices, and let L_G be its Laplacian matrix, then

1. All the eigenvalues of L_G are real and nonnegative.
2. 0 is the smallest eigenvalue of L_G.

Remark:

$$L_G = D - W = \sum_{i<j} W_{ij}(e_i - e_j)(e_i - e_j)^T$$

Solution to the Balanced Minimum Cut

Relaxed balanced minimum cut problem:

$$\min_{||y||=\sqrt{n}, 1^T y=0} \frac{1}{4} y^T L_G y$$

If we take $z = y/\sqrt{n}$, we can write the above problem as

$$\min_{||z||=1, 1^T z=0} \frac{n}{4} z^T L_G z$$

Theorem The solution z for the relaxed balanced minimum cut problem is the eigenvector corresponding to the second smallest eigenvalue.

Spectral Clustering Algorithm for Two Clusters

Input: A graph G = (V, E, W).

Step 1: Compute the Laplacian matrix L_G.

Step 2: Calculate the eigenvector v_2 corresponding to the second smallest eigenvalues of L_G. Form $V = [1, v_2]^T$.

Step 3: Run K-means algorithm on the columns of V.

Spectral Clustering for K Clusters

Spectral Clustering Algorithm

Input: A graph G = (V, E, W), number of clusters K.

Step 1: Compute the noralized Laplacian matrix L_G.

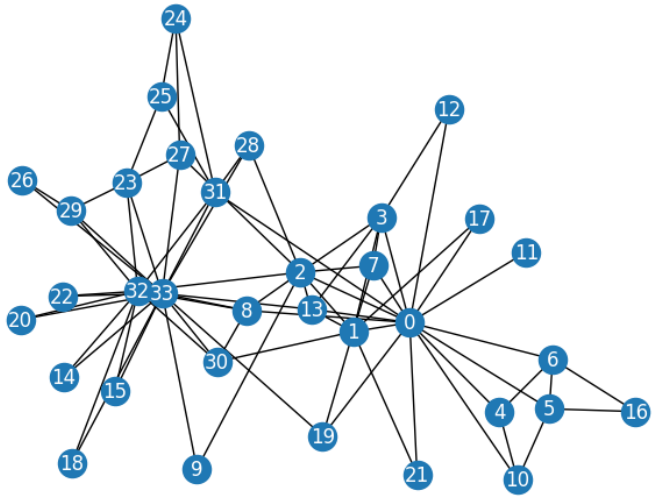
Step 2: Calculate the eigenvectors corresponding to the smallest K eigenvalues of L_G. Form it as $V = [v_1, v_2, \dots, v_K]^T$.

Step 3: Run K-means algorithm on the columns of V.

Applications in Community Detection

Model Two Clusters Clustering Problem as a Balanced Minimum Cut

Community detection in a network identifies and groups the more densely interconnected nodes in a given graph. This graph can take the form of a social network graph, a biological network, or a representation of a local network of computers。



Spectral Clustering for Community Detection

Spectral Clustering Algorithm for Community Detection

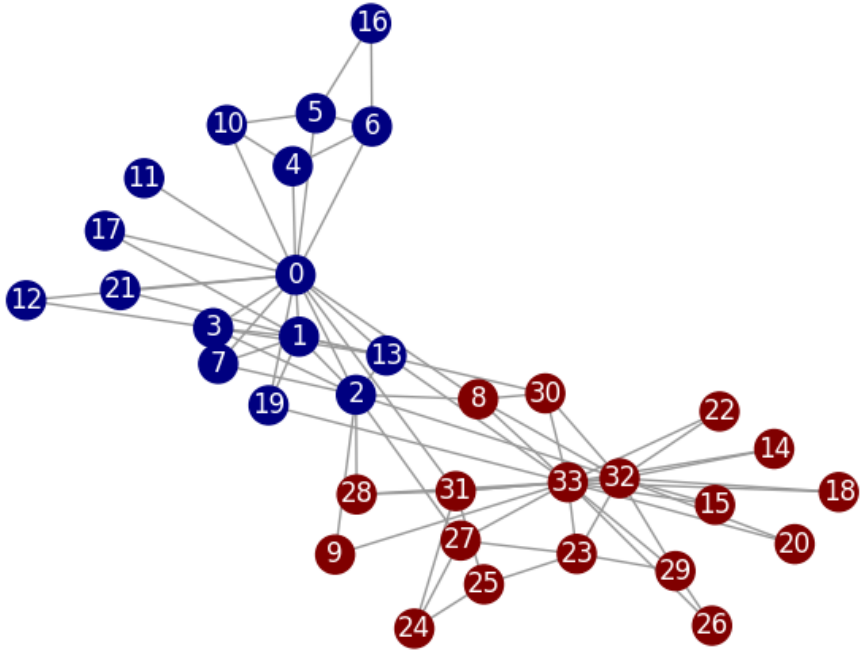
Input: A graph G = (V, E, W), number of clusters K.

Step 1: Compute the noralized Laplacian matrix L_G.

Step 2: Calculate the eigenvectors corresponding to the smallest K eigenvalues of L_G. Form it as $V = [v_1, v_2, \dots, v_K]^T$.

Step 3: Run K-means algorithm on the columns of V.

Application in Karate Club Dataset




Thank You!