Clustering
Lecture 6: Spectral Methods

Jing Gao
SUNY Buffalo
Outline

• Basics
  – Motivation, definition, evaluation

• Methods
  – Partitional
  – Hierarchical
  – Density-based
  – Mixture model
  – Spectral methods

• Advanced topics
  – Clustering ensemble
  – Clustering in MapReduce
  – Semi-supervised clustering, subspace clustering, co-clustering, etc.
Motivation

• **Complex cluster shapes**
  – K-means performs poorly because it can only find spherical clusters
  – Density-based approaches are sensitive to parameters

• **Spectral approach**
  – Use similarity graphs to encode local neighborhood information
  – Data points are vertices of the graph
  – Connect points which are “close”
Similarity Graph

• Represent dataset as a weighted graph \( G(V,E) \)
• All vertices which can be reached from each other by a path form a connected component
• Only one connected component in the graph—The graph is fully connected

\( V=\{x_i\} \)  Set of \( n \) vertices representing data points

\( E=\{W_{ij}\} \)  Set of weighted edges indicating pair-wise similarity between points
Graph Construction

• **ε-neighborhood graph**
  – Identify a threshold value, ε, and include edges if the affinity between two points is greater than ε

• **k-nearest neighbors**
  – Insert edges between a node and its $k$-nearest neighbors
  – Each node will be connected to (at least) $k$ nodes

• **Fully connected**
  – Insert an edge between every pair of nodes
  – Weight of the edge represents similarity
  – Gaussian kernel:

$$w_{ij} = \exp\left(-\|x_i - x_j\|^2 / \sigma^2\right)$$
ε-neighborhood Graph

• ε-neighborhood
  – Compute pairwise distance between any two objects
  – Connect each point to all other points which have distance smaller than a threshold ε

• Weighted or unweighted
  – Unweighted—There is an edge if one point belongs to the ε–neighborhood of another point
  – Weighted—Transform distance to similarity and use similarity as edge weights
kNN Graph

• Directed graph
  – Connect each point to its $k$ nearest neighbors

• kNN graph
  – Undirected graph
  – An edge between $x_i$ and $x_j$: There’s an edge from $x_i$ to $x_j$ OR from $x_j$ to $x_i$ in the directed graph

• Mutual kNN graph
  – Undirected graph
  – Edge set is a subset of that in the kNN graph
  – An edge between $x_i$ and $x_j$: There’s an edge from $x_i$ to $x_j$ AND from $x_j$ to $x_i$ in the directed graph
Clustering Objective

Traditional definition of a “good” clustering

• Points assigned to same cluster should be highly similar
• Points assigned to different clusters should be highly dissimilar

Apply this objective to our graph representation

Minimize weight of between-group connections
Graph Cuts

• Express clustering objective as a function of the edge cut of the partition

• Cut: Sum of weights of edges with only one vertex in each group

• We wants to find the minimal cut between groups

\[
cut(C_1, C_2) = \sum_{i \in C_1, j \in C_2} w_{ij}
\]

\[
cut(C_1, C_2) = 0.3
\]
Bi-partitional Cuts

• Minimum (bi-partitional) cut

$$\min \text{Cut}(C_1, C_2) = \sum_{i \in C_1} \sum_{j \in C_2} w_{ij}$$
Example

- Minimum Cut

\[ \text{Cut}(BCDE, A) = 0.17 \]
Normalized Cuts

- Minimal (bipartitional) normalized cut

$$\min \frac{\text{Cut}(C_1, C_2)}{\text{Vol}(C_1)} + \frac{\text{Cut}(C_1, C_2)}{\text{Vol}(C_2)} = \min \left( \frac{1}{\text{Vol}(C_1)} + \frac{1}{\text{Vol}(C_2)} \right) \text{Cut}(C_1, C_2)$$

$$\text{Vol}(C) = \sum_{i \in C, j \in V} w_{ij}$$
Example

• Normalized Minimum Cut

\[ \text{NormCut}(C_1, C_2) = \frac{\text{Cut}(C_1, C_2)}{\text{Vol}(C_1)} + \frac{\text{Cut}(C_1, C_2)}{\text{Vol}(C_2)} \]

\[ \text{NormCut}(BCDE, A) = 1.067 \]
Example

• Normalized Minimum Cut

\[ \text{NormCut}(C_1, C_2) = \frac{\text{Cut}(C_1, C_2)}{\text{Vol}(C_1)} + \frac{\text{Cut}(C_1, C_2)}{\text{Vol}(C_2)} \]

\[ \text{NormCut}(BCDE, A) = 1.067 \]
\[ \text{NormCut}(ABC, DE) = 1.038 \]
Problem

- Identifying a minimum cut is NP-hard
- There are efficient approximations using linear algebra
- Based on the Laplacian Matrix, or graph Laplacian
Matrix Representations

• **Similarity matrix** \( (W) \)
  
  – \( n \times n \) matrix
  
  – \( W = [w_{ij}] : \) edge weight between vertex \( x_i \) and \( x_j \)

<table>
<thead>
<tr>
<th></th>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( x_3 )</th>
<th>( x_4 )</th>
<th>( x_5 )</th>
<th>( x_6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>0</td>
<td>0.8</td>
<td>0.6</td>
<td>0</td>
<td>0.1</td>
<td>0</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>0.8</td>
<td>0</td>
<td>0.8</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( x_3 )</td>
<td>0.6</td>
<td>0.8</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( x_4 )</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
<td>0.8</td>
<td>0.7</td>
</tr>
<tr>
<td>( x_5 )</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
<td>0.8</td>
<td>0</td>
<td>0.8</td>
</tr>
<tr>
<td>( x_6 )</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>0.8</td>
<td>0</td>
</tr>
</tbody>
</table>

• **Important properties**
  
  – Symmetric matrix
Matrix Representations

- **Degree matrix** ($D$)
  - $n \times n$ diagonal matrix
  - $D(i,i) = \sum_j w_{ij}$: total weight of edges incident to vertex $x_i$

- **Used to**
  - Normalize adjacency matrix
Matrix Representations

- **Laplacian matrix** $(L)$
  - $n \times n$ symmetric matrix

  $$L = D - W$$

  - **Important properties**
    - Eigenvalues are non-negative real numbers
    - Eigenvectors are real and orthogonal
    - Eigenvalues and eigenvectors provide an insight into the connectivity of the graph...
Find An Optimal Min-Cut (Hall’70, Fiedler’73)

• Express a bi-partition \((C_1, C_2)\) as a vector

\[
f_i = \begin{cases} 
1 & \text{if } x_i \in C_1 \\
-1 & \text{if } x_i \in C_2 
\end{cases}
\]

• We can minimise the cut of the partition by finding a non-trivial vector \(f\) that minimizes the function

\[
g(f) = \sum_{i,j \in V} w_{ij} (f_i - f_j)^2 = 2 f^T L f
\]

Laplacian matrix
Why does this work?

\[ L = D - W \]

\[ f^T L f = f^T D f - f^T W f \]

\[ = \sum_i d_i f_i^2 - \sum_{ij} f_i f_j w_{ij} \]

\[ = \frac{1}{2} \left( \sum_i \left( \sum_j w_{ij} \right) f_i^2 - 2 \sum_{ij} f_i f_j w_{ij} + \sum_j \left( \sum_i w_{ij} \right) f_j^2 \right) \]

\[ = \frac{1}{2} \sum_{ij} w_{ij} (f_i - f_j)^2 \]

--Clustering objective function

- \( L f = \lambda f \)  \( f^T L f = f^T \lambda f = \lambda f^T f = \lambda \)

- If we let \( f \) be eigen vectors of \( L \), then the corresponding eigen value is the value of the clustering objective function
Optimal Min-Cut

• The Laplacian matrix $L$ is semi-positive definite
• The Rayleigh Theorem shows:
  – The minimum value for $g(f)$ is given by the 2nd smallest eigenvalue of the Laplacian $L$
  – The optimal solution for $f$ is given by the corresponding eigenvector $\lambda_2$, referred as the Fiedler Vector
Spectral Bi-partitioning Algorithm

1. Pre-processing
   - Build Laplacian matrix $L$ of the graph

2. Decomposition
   - Find eigenvectors $X$ and eigenvalues $\Lambda$ of the matrix $L$
   - Map vertices to corresponding components of $\lambda_2$
**Spectral Bi-partitioning Algorithm**

The matrix which represents the eigenvector of the Laplacian (the eigenvector matched to the corresponded eigenvalues with increasing order)

<table>
<thead>
<tr>
<th></th>
<th>0.41</th>
<th>-0.41</th>
<th>0.65-</th>
<th>0.31-</th>
<th>0.38-</th>
<th>0.11</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.41</td>
<td>-0.44</td>
<td>0.01</td>
<td>0.30</td>
<td>0.71</td>
<td>0.22</td>
</tr>
<tr>
<td></td>
<td>0.41</td>
<td>-0.37</td>
<td>0.64</td>
<td>0.04</td>
<td>0.39-</td>
<td>0.37-</td>
</tr>
<tr>
<td></td>
<td>0.41</td>
<td>0.37</td>
<td>0.34</td>
<td>0.45-</td>
<td>0.00</td>
<td>0.61</td>
</tr>
<tr>
<td></td>
<td>0.41</td>
<td>0.41</td>
<td>0.17-</td>
<td>0.30-</td>
<td>0.35</td>
<td>0.65-</td>
</tr>
<tr>
<td></td>
<td>0.41</td>
<td>0.45</td>
<td>0.18-</td>
<td>0.72</td>
<td>0.29-</td>
<td>0.09</td>
</tr>
</tbody>
</table>

![Bar chart](image)
**Spectral Bi-partitioning**

- **Grouping**
  - Sort components of reduced 1-dimensional vector
  - Identify clusters by splitting the sorted vector in two (above zero, below zero)

<table>
<thead>
<tr>
<th>X</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>X_1</td>
<td>0.2</td>
</tr>
<tr>
<td>X_2</td>
<td>0.2</td>
</tr>
<tr>
<td>X_3</td>
<td>0.2</td>
</tr>
<tr>
<td>X_4</td>
<td>-0.4</td>
</tr>
<tr>
<td>X_5</td>
<td>-0.7</td>
</tr>
<tr>
<td>X_6</td>
<td>-0.7</td>
</tr>
</tbody>
</table>

Split at 0

- Cluster C₁:
  - Positive points
- Cluster C₂:
  - Negative points

<table>
<thead>
<tr>
<th>X</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>X_1</td>
<td>0.2</td>
</tr>
<tr>
<td>X_2</td>
<td>0.2</td>
</tr>
<tr>
<td>X_3</td>
<td>0.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>X_4</td>
<td>-0.4</td>
</tr>
<tr>
<td>X_5</td>
<td>-0.7</td>
</tr>
<tr>
<td>X_6</td>
<td>-0.7</td>
</tr>
</tbody>
</table>
**Normalized Laplacian**

- **Laplacian matrix** ($L$)

\[
L = D^{-1}(D - W)
\]

\[
L = D^{-0.5}(D - W)D^{-0.5}
\]

<table>
<thead>
<tr>
<th></th>
<th>1.00</th>
<th>-0.52</th>
<th>-0.39</th>
<th>0.00</th>
<th>-0.06</th>
<th>0.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.52</td>
<td>1.00</td>
<td>-0.50</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>-0.39</td>
<td>-0.50</td>
<td>1.00</td>
<td>-0.12</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>-0.12</td>
<td>1.00</td>
<td>0.47-</td>
<td>0.44-</td>
<td></td>
</tr>
<tr>
<td>-0.06</td>
<td>0.00</td>
<td>0.00</td>
<td>-0.47</td>
<td>1.00</td>
<td>0.50-</td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.44-</td>
<td>0.50-</td>
<td>1.00</td>
<td></td>
</tr>
</tbody>
</table>
**K-Way Spectral Clustering**

- How do we partition a graph into $k$ clusters?

  1. **Recursive bi-partitioning** (Hagen et al.,’91)
     - Recursively apply bi-partitioning algorithm in a hierarchical divisive manner.
     - Disadvantages: Inefficient, unstable

  2. **Cluster multiple eigenvectors** (Shi & Malik,’00)
     - Build a reduced space from multiple eigenvectors.
     - Commonly used in recent papers
     - A preferable approach
Eigenvectors & Eigenvalues

Histogram of the sample

Eigenvalues

Eigenvector 1

Eigenvector 2

Eigenvector 3

Eigenvector 4

Eigenvector 5
**K-way Spectral Clustering Algorithm**

- **Pre-processing**
  - Compute Laplacian matrix \( L \)

- **Decomposition**
  - Find the eigenvalues and eigenvectors of \( L \)
  - Build embedded space from the eigenvectors corresponding to the \( k \) smallest eigenvalues

- **Clustering**
  - Apply \( k \)-means to the reduced \( n \times k \) space to produce \( k \) clusters
How to select $k$?

- *Eigengap*: the difference between two consecutive eigenvalues
- Most stable clustering is generally given by the value $k$ that maximizes the expression

$$\Delta_k = \left| \lambda_k - \lambda_{k-1} \right|$$

$$\max \Delta_k = \left| \lambda_2 - \lambda_1 \right|$$

$\Rightarrow$ Choose $k=2$
Take-away Message

• Clustering formulated as graph cut problem
• How min-cut can be solved by eigen decomposition of Laplacian matrix
• Bipartition and multi-partition spectral clustering procedure