Lecture 23:

- Spectral clustering
- Hierarchical clustering
- What is a good clustering?
Administrative

• **Project presentations** will be held on Thursday, January 5, 2017, 11:00am-12:30pm

• **Final exam** is on January 9, 2017, 09:30-11:30am at D8.

• **Final project reports** are due on January 6, 2017, 11:59:59pm
Last time... K-Means

- **An iterative clustering algorithm**
  - **Initialize:** Pick $K$ random points as cluster centers (means)
  - **Alternate:**
    - Assign data instances to closest mean
    - Assign each mean to the average of its assigned points
  - **Stop when no points’ assignments change**
Today

• Spectral clustering
• Hierarchical clustering
• What is a good clustering?
Spectral clustering
Graph-Theoretic Clustering

**Goal:** Given data points \( x_1, \ldots, x_n \) and similarities \( W(x_i, x_j) \), partition the data into groups so that points in a group are similar and points in different groups are dissimilar.

**Similarity Graph:** \( G(V,E,W) \)

- \( V \) – Vertices (Data points)
- \( E \) – Edge if similarity > 0
- \( W \) - Edge weights (similarities)

Partition the graph so that edges within a group have large weights and edges across groups have small weights.
Graphs Representations

Adjacency Matrix

\[
\begin{bmatrix}
a & b & c & d & e \\
a & 0 & 1 & 0 & 0 & 1 \\
b & 1 & 0 & 0 & 0 & 0 \\
c & 0 & 0 & 0 & 0 & 1 \\
d & 0 & 0 & 0 & 0 & 1 \\
e & 1 & 0 & 1 & 1 & 0 \\
\end{bmatrix}
\]
A Weighted Graph and its Representation

Affinity Matrix

\[
\begin{bmatrix}
1 & .1 & .3 & 0 & 0 \\
.1 & 1 & .4 & 0 & .2 \\
.3 & .4 & 1 & .6 & .7 \\
0 & 0 & .6 & 1 & 1 \\
0 & .2 & .7 & 1 & 1 \\
\end{bmatrix}
\]

\(W_{ij}\): probability that \(i\) & \(j\) belong to the same cluster
Similarity graph construction

- **Similarity Graphs**: Model local neighborhood relations between data points

- E.g. epsilon-NN
  \[ W_{ij} = \begin{cases} 
  1 & \|x_i - x_j\| \leq \epsilon \\
  0 & \text{otherwise}
\end{cases} \]

  or mutual k-NN graph (\( W_{ij} = 1 \) if \( x_i \) or \( x_j \) is k nearest neighbor of the other)

Controls size of neighborhood

\[ G = \{V,E\} \]
Similarity graph construction

• Similarity Graphs: Model local neighborhood relations between data points

• E.g. Gaussian kernel similarity function

\[ W_{ij} = e^{-\frac{||x_i - x_j||^2}{2\sigma^2}} \]

Controls size of neighborhood
Scale affects affinity

- **Small $\sigma$:** group only nearby points
- **Large $\sigma$:** group far-away points
Feature grouping by “relocalisation” of eigenvectors of the proximity matrix

\[ W_{ij} = \exp(-\| z_i - z_j \|^2 / s^2) \]

With an appropriate \( s \)

The eigenvectors of \( W \) are:

\[
\begin{array}{ccc}
A & B & C \\
A & 1.00 & 0.63 & 0.03 \\
B & 0.63 & 1.00 & 0.0 \\
C & 0.03 & 0.0 & 1.00 \\
\end{array}
\]

The first 2 eigenvectors group the points as desired…
Example eigenvector

![Affinity matrix](image1.png)

![Eigenvector points](image2.png)
Example eigenvector

Affinity matrix

points

eigenvector
Graph cut

- Set of edges whose removal makes a graph disconnected
- Cost of a cut: sum of weights of cut edges
- A graph cut gives us a partition (clustering)
  - What is a “good” graph cut and how do we find one?
Minimum cut

- A cut of a graph $G$ is the set of edges $S$ such that removal of $S$ from $G$ disconnects $G$.

**Cut:** sum of the weight of the cut edges:

$$\text{cut}(A,B) = \sum_{u \in A, \ v \in B} W(u, v),$$

with $A \cap B = \emptyset$
Minimum cut

- We can do segmentation by finding the minimum cut in a graph
- Efficient algorithms exist for doing this

Minimum cut example
Minimum cut

• We can do segmentation by finding the minimum cut in a graph
  - Efficient algorithms exist for doing this

Minimum cut example
Drawbacks of Minimum cut

• Weight of cut is directly proportional to the number of edges in the cut.

* Slide from Khurram Hassan-Shafique CAP5415 Computer Vision 2003
Normalized cuts

Write graph as $V$, one cluster as $A$ and the other as $B$

$\text{Ncut}(A,B) = \frac{\text{cut}(A,B)}{\text{assoc}(A,V)} + \frac{\text{cut}(A,B)}{\text{assoc}(B,V)}$

$\text{cut}(A,B)$ is sum of weights with one end in $A$ and one end in $B$

$\text{cut}(A,B) = \sum_{u \in A, v \in B} W(u,v)$,

with $A \cap B = \emptyset$

$\text{assoc}(A,V)$ is sum of all edges with one end in $A$.

$\text{assoc}(A,B) = \sum_{u \in A, v \in B} W(u,v)$

$A$ and $B$ not necessarily disjoint

Normalized cut

Let $W$ be the adjacency matrix of the graph
Let $D$ be the diagonal matrix with diagonal entries $D(i, i) = \sum_j W(i, j)$
Then the normalized cut cost can be written as

$$\frac{y^T (D - W) y}{y^T D y}$$

where $y$ is an indicator vector whose value should be 1 in the $i$-th position if the $i$-th feature point belongs to $A$ and a negative constant otherwise

J. Shi and J. Malik. Normalized cuts and image segmentation. PAMI 2000
Normalized cut

• Finding the exact minimum of the normalized cut cost is NP-complete, but if we relax $y$ to take on arbitrary values, then we can minimize the relaxed cost by solving the generalized eigenvalue problem

$$(D - W)y = \lambda Dy$$

• The solution $y$ is given by the generalized eigenvector corresponding to the second smallest eigenvalue

• Intuitively, the $i$-th entry of $y$ can be viewed as a “soft” indication of the component membership of the $i$-th feature

  - Can use 0 or median value of the entries as the splitting point (threshold), or find threshold that minimizes the Ncut cost

J. Shi and J. Malik. Normalized cuts and image segmentation, PAMI 2000
Normalized cut algorithm

1. Given an image or image sequence, set up a weighted graph \( G = (V, E) \), and set the weight on the edge connecting two nodes being a measure of the similarity between the two nodes.

2. Solve \( (D - W)x = \lambda Dx \) for eigenvectors with the smallest eigenvalues.

3. Use the eigenvector with second smallest eigenvalue to bipartition the graph.

4. Decide if the current partition should be sub-divided, and recursively repartition the segmented parts if necessary.

J. Shi and J. Malik. [Normalized cuts and image segmentation](https://doi.org/10.1109/34.868688), PAMI 2000
K-Means vs. Spectral Clustering

- Applying k-means to Laplacian eigenvectors allows us to find clusters with non-convex boundaries.

Both perform same

Spectral clustering is superior
K-Means vs. Spectral Clustering

- Applying k-means to Laplacian eigenvectors allows us to find clusters with non-convex boundaries.
K-Means vs. Spectral Clustering

- Applying k-means to Laplacian eigenvectors allows us to find cluster with non-convex boundaries.
Examples

squiggles, 4 clusters

nips, 8 clusters

Ng et al., 2001
Some Issues

• Choice of number of clusters $k$
  - Most stable clustering is usually given by the value of $k$ that maximizes the eigengap (difference between consecutive eigenvalues)

$$\Delta_k = |\lambda - \lambda_{k-1}|$$

![](histogram.png) ![](eigenvalues.png)
Some Issues

- Choice of number of clusters $k$
- Choice of similarity
  - Choice of kernel
    for Gaussian kernels, choice of $\sigma$

Good similarity measure

Poor similarity measure
Some Issues

• Choice of number of clusters $k$

• Choice of similarity
  - Choice of kernel
    for Gaussian kernels, choice of $\sigma$

• Choice of clustering method
  - $k$-way vs. recursive 2-way
Hierarchical clustering
Hierarchical Clustering

- **Bottom-Up (agglomerative):** Starting with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

- The number of dendrograms with $n$ leafs is given by $n$ leafs = $(2n -3)!/[(2(n-2))(n-2)!]$

<table>
<thead>
<tr>
<th>Number of leafs</th>
<th>Number of possible Dendrograms</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>105</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>10</td>
<td>34,459,425</td>
</tr>
</tbody>
</table>
We begin with a distance matrix which contains the distances between every pair of objects in our dataset.

\[
\begin{array}{ccccc}
0 & 8 & 8 & 7 & 7 \\
0 & 2 & 4 & 4 & \\
0 & 3 & 3 & \\
0 & 1 & \\
0 & \\
\end{array}
\]

\[
\begin{align*}
D(, ) &= 8 \\
D(, ) &= 1
\end{align*}
\]
Bottom-Up (agglomerative):
Start with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.
Bottom-Up (agglomerative):
Start with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.
Bottom-Up (agglomerative):
Start with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

Consider all possible merges…

Choose the best
Bottom-Up (agglomerative):
Start with each item in its own cluster, find the best pair to merge into a new cluster. Repeat until all clusters are fused together.

But how do we compute distances between clusters rather than objects?
Computing distance between clusters: **Single Link**

- Cluster distance = distance of two closest members in each class

  ![Diagram of clusters and distances](image)

  - Potentially long and skinny clusters
Computing distance between clusters: Complete Link

- Cluster distance = distance of two **farthest** members in each class

- Tight clusters
Computing distance between clusters: **Average Link**

- Cluster distance = average distance of all pairs

- The most widely used measure
- Robust against noise
Agglomerative Clustering

Good
• Simple to implement, widespread application
• Clusters have adaptive shapes
• Provides a hierarchy of clusters

Bad
• May have imbalanced clusters
• Still have to choose number of clusters or threshold
• Need to use an “ultrametric” to get a meaningful hierarchy
What is a good clustering?
What is a good clustering?

- **Internal criterion**: A good clustering will produce high quality clusters in which:
  - the intra-class (that is, intra-cluster) similarity is high
  - the inter-class similarity is low
  - The measured quality of a clustering depends on both the obj. representation and the similarity measure used

- **External criteria** for clustering quality
  - Quality measured by its ability to discover some or all of the hidden patterns or latent classes in gold standard data
  - Assesses a clustering with respect to ground truth
  - Example:
    - Purity
    - Entropy of classes in clusters (or Mutual Information between classes and clusters)
External Evaluation of Cluster Quality

- Simple measure: purity, the ratio between the dominant class in the cluster and the size of cluster
  - Assume documents with C gold standard classes, while our clustering algorithms produce K clusters, $\omega_1, \omega_2, \ldots, \omega_K$ with $n_i$ members.

$$Purity(w_i) = \frac{1}{n_i} \max_{j} (n_{ij}) \quad j \in C$$

- Example:

Cluster I: Purity = $\frac{1}{6} \max(5, 1, 0) = \frac{5}{6}$
Cluster II: Purity = $\frac{1}{6} \max(1, 4, 1) = \frac{4}{6}$
Cluster III: Purity = $\frac{1}{5} \max(2, 0, 3) = \frac{3}{5}$
External Evaluation of Cluster Quality

- Let:
  \[ TC = TC_1 \cup TC_2 \cup ... \cup TC_n \]
  \[ CC = CC_1 \cup CC_2 \cup ... \cup CC_m \]
  be the target and computed clusterings, respectively.

- \( TC = CC \) = original set of data

- Define the following:
  - \( a \): number of pairs of items that belong to the same cluster in both \( CC \) and \( TC \)
  - \( b \): number of pairs of items that belong to different clusters in both \( CC \) and \( TC \)
  - \( c \): number of pairs of items that belong to the same cluster in \( CC \) but different clusters in \( TC \)
  - \( d \): number of pairs of items that belong to the same cluster in \( TC \) but different clusters in \( CC \)
External Evaluation of Cluster Quality

Measure of clustering agreement: how similar are these two ways of partitioning the data?

\[ P = \frac{a}{a + c} \]
\[ R = \frac{a}{a + d} \]
\[ F = \frac{2 \times P \times R}{P + R} \]

Rand Index

\[ \frac{a + b}{a + b + c + d} \]

F-measure

Measure of clustering agreement: how similar are these two ways of partitioning the data?
External Evaluation of Cluster Quality

**Rand Index**

\[
\frac{a + b}{a + b + c + d}
\]

**Adjusted Rand Index**

\[
\frac{2(ab - cd)}{(a + c)(c + b) + (a + d)(d + b)}
\]

Extension of the Rand index that attempts to account for items that may have been clustered by chance.
External Evaluation of Cluster Quality

$$Entropy(CC_i) = \sum_{TC_j \in TC} -p(TC_j | CC_i) \log p(TC_j | CC_i)$$

$$AvgEntropy(CC) = \sum_{i=1}^{m} \frac{|CC_i|}{|CC|} Entropy(CC_i)$$

**Average Entropy**

Measure of purity with respect to the target clustering