Outline

Part 1: Basic Concepts of data clustering
- Non-Supervised Learning and Clustering
  - Problem formulation – cluster analysis
  - Taxonomies of Clustering Techniques
  - Data types and Proximity Measures
  - Difficulties and open problems

Part 2: Clustering Algorithms
- Hierarchical methods
  - Single-link
  - Complete-link
  - Clustering Based on Dissimilarity Increments Criteria

Hierarchical Clustering

Use proximity matrix: nxn
- \( D(i,j) \): proximity (similarity or distance) between patterns \( i \) and \( j \)

Diagram:
- Step 0
- Step 1
- Step 2
- Step 3
- Step 4

Agglomerative

Step 4 → Step 3 → Step 2 → Step 1 → Step 0

Divisive
Hierarchical Clustering: Agglomerative Methods

1. Start with \( n \) clusters containing one object
2. Find the most similar pair of clusters \( C_i \) and \( C_j \) from the proximity matrix and merge them into a single cluster
3. Update the proximity matrix (reduce its order by one, by replacing the individual clusters with the merged cluster)
4. Repeat steps (2) e (3) until a single cluster is obtained (i.e. \( N - 1 \) times)

Similarity measures between clusters:

Well known similarity measures can be written using the Lance-Williams formula, expressing the distance between cluster \( k \) and cluster \( i+j \), obtained by the merging of clusters \( i \) and \( j \):

\[
d(i+j,k) = a_d(i,k) + a_d(j,k) + b_d(i,j) + c_d(i,k) - d(j,k)
\]

<table>
<thead>
<tr>
<th>Method</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-link</td>
<td>( a = a_0 = 0.5 ); ( b = 0 ); ( c = -0.5 ) ( d(i+j,k) = \min {d(i,k),d(j,k)} )</td>
</tr>
<tr>
<td>Complete-link</td>
<td>( a = a_0 = 0.5 ); ( b = 0 ); ( c = 0.5 ) ( d(i+j,k) = \max {d(i,k),d(j,k)} )</td>
</tr>
<tr>
<td>Centroid</td>
<td>( a = \frac{n_i}{n_i+n_j} ); ( a = \frac{n_j}{n_i+n_j} ); ( b = \frac{n_i n_j}{n_i+n_j} ); ( c = 0 ) ( d(i+j,k) = d(\mu_i, \mu_j) )</td>
</tr>
<tr>
<td>Median</td>
<td>( a = a_0 = 0.5 ); ( b = -0.25 ); ( c = 0 )</td>
</tr>
<tr>
<td>(Average link)</td>
<td>( a = \frac{n_i}{n_i+n_j} ); ( a = \frac{n_j}{n_i+n_j} ); ( b = c = 0 ) ( d(C_i,C_j) = \frac{1}{n_{C_i} n_{C_j}} \sum d(a,b) )</td>
</tr>
<tr>
<td>Ward’s Method</td>
<td>(minimum variance) ( a = \frac{n_i}{n_i+n_j} ); ( a = \frac{n_j}{n_i+n_j} ); ( b = -\frac{n_i}{n_i+n_j} ); ( c = 0 )</td>
</tr>
</tbody>
</table>
Hierarchical Clustering: Agglomerative Methods

**Single Link:** Distance between two clusters is the distance between the closest points. Also called “neighbor joining.”

\[
\begin{align*}
\text{Single-link} & : a_i = a_j = 0.5 \land b = 0 \land c = -0.5 \land d(i, j, k) = \min\{d(i, k), d(j, k)\} \\
\text{Complete-link} & : a_i = a_j = 0.5 \land b = 0 \land c = 0 \land d(i, j, k) = \max\{d(i, k), d(j, k)\} \\
\text{Centroid} & : a_i = \frac{n_i}{n_i + n_j} \land a_j = \frac{n_j}{n_i + n_j} \land b = -\frac{n_{ij}}{n_i + n_j} \land c = 0 \land d(i, j, k) = d(i, k) + d(j, k) \\
\text{Median} & : a_i = a_j = 0.5 \land b = -0.25 \land c = 0 \\
\text{(Average link)} & : a_i = \frac{n_i}{n_i + n_j} \land a_j = \frac{n_j}{n_i + n_j} \land b = c = 0 \land d(C_i, C_j) = \frac{1}{n_{ij}} \sum_{a,b} d(a, b) \\
\text{Ward’s Method (minimum variance)} & : a_i = \frac{n_i + n_j}{n_i + n_j + n_k} \land a_j = \frac{n_j + n_k}{n_i + n_j + n_k} \land a_k = \frac{n_k + n_i}{n_i + n_j + n_k} \land b = -\frac{n_i}{n_i + n_j + n_k} \land c = 0
\end{align*}
\]

**Complete Link:** Distance between clusters is distance between farthest pair of points.

\[
\begin{align*}
\text{Single-link} & : a_i = a_j = 0.5 \land b = 0 \land c = -0.5 \land d(i, j, k) = \min\{d(i, k), d(j, k)\} \\
\text{Complete-link} & : a_i = a_j = 0.5 \land b = 0 \land c = 0 \land d(i, j, k) = \max\{d(i, k), d(j, k)\} \\
\text{Centroid} & : a_i = \frac{n_i}{n_i + n_j} \land a_j = \frac{n_j}{n_i + n_j} \land b = -\frac{n_{ij}}{n_i + n_j} \land c = 0 \land d(i, j, k) = d(i, k) + d(j, k) \\
\text{Median} & : a_i = a_j = 0.5 \land b = -0.25 \land c = 0 \\
\text{(Average link)} & : a_i = \frac{n_i}{n_i + n_j} \land a_j = \frac{n_j}{n_i + n_j} \land b = c = 0 \land d(C_i, C_j) = \frac{1}{n_{ij}} \sum_{a,b} d(a, b) \\
\text{Ward’s Method (minimum variance)} & : a_i = \frac{n_i + n_j}{n_i + n_j + n_k} \land a_j = \frac{n_j + n_k}{n_i + n_j + n_k} \land a_k = \frac{n_k + n_i}{n_i + n_j + n_k} \land b = -\frac{n_i}{n_i + n_j + n_k} \land c = 0
\end{align*}
\]
### Hierarchical Clustering: Agglomerative Methods

#### Centroid: Distance between clusters is distance between centroids.

<table>
<thead>
<tr>
<th>Method</th>
<th>Formula</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-link</td>
<td>$a_i = a_j = 0.5$ ; $b = 0$ ; $c = 0$</td>
<td>$d(i, j, k) = \min[d(i, k), d(j, k)]$</td>
</tr>
<tr>
<td>Complete-link</td>
<td>$a_i = a_j = 0.5$ ; $b = 0$ ; $c = 0.5$</td>
<td>$d(i, j, k) = \max[d(i, k), d(j, k)]$</td>
</tr>
<tr>
<td>Centeroid</td>
<td>$a_i = n_i / n_i + n_j$ ; $a_j = n_j / n_i + n_j$ ; $b = \frac{n_i n_j}{n_i + n_j}$</td>
<td>$c = 0$ $d(i, j, k) = d(\mu_i, \mu_j)$</td>
</tr>
<tr>
<td>Median</td>
<td>$a_i = a_j = 0.5$ ; $b = -0.25$ ; $c = 0$</td>
<td></td>
</tr>
<tr>
<td>(Average link)</td>
<td>$a_i = n_i / n_i + n_j$ ; $a_j = n_j / n_i + n_j$ ; $b = \frac{n_i n_j}{n_i + n_j}$</td>
<td>$c = 0$ $d(C_i, C_j) = \frac{1}{n_i n_j} \sum d(a, b)$</td>
</tr>
<tr>
<td>Ward’s Method (minimum variance)</td>
<td>$a_i = a_j = 0.5$ ; $b = 0$ ; $c = 0$</td>
<td></td>
</tr>
</tbody>
</table>

#### Average Link: Distance between clusters is average distance between the cluster points.

<table>
<thead>
<tr>
<th>Method</th>
<th>Formula</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-link</td>
<td>$a_i = a_j = 0.5$ ; $b = 0$ ; $c = 0$</td>
<td>$d(i, j, k) = \min[d(i, k), d(j, k)]$</td>
</tr>
<tr>
<td>Complete-link</td>
<td>$a_i = a_j = 0.5$ ; $b = 0$ ; $c = 0.5$</td>
<td>$d(i, j, k) = \max[d(i, k), d(j, k)]$</td>
</tr>
<tr>
<td>Centeroid</td>
<td>$a_i = n_i / n_i + n_j$ ; $a_j = n_j / n_i + n_j$ ; $b = \frac{n_i n_j}{n_i + n_j}$</td>
<td>$c = 0$ $d(i, j, k) = d(\mu_i, \mu_j)$</td>
</tr>
<tr>
<td>Median</td>
<td>$a_i = a_j = 0.5$ ; $b = -0.25$ ; $c = 0$</td>
<td></td>
</tr>
<tr>
<td>(Average link)</td>
<td>$a_i = n_i / n_i + n_j$ ; $a_j = n_j / n_i + n_j$ ; $b = \frac{n_i n_j}{n_i + n_j}$</td>
<td>$c = 0$ $d(C_i, C_j) = \frac{1}{n_i n_j} \sum d(a, b)$</td>
</tr>
<tr>
<td>Ward’s Method (minimum variance)</td>
<td>$a_i = a_j = 0.5$ ; $b = 0$ ; $c = 0$</td>
<td></td>
</tr>
</tbody>
</table>
Unsupervised Learning

Clustering Algorithms

Hierarchical Clustering: Agglomerative Methods

Ward's Link: Minimizes the sum-of-squares criterion (measure of heterogeneity)

$$ESS = \sum_{k=1}^{K} \sum_{j=1} d \left( x_{i,j} - \bar{x}_{kj} \right)^2$$

<table>
<thead>
<tr>
<th>Method</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-link</td>
<td>$$a_i = a_j = 0.5; \quad b = 0; \quad c = -0.5 \quad d(i+j,k) = \min {d(i,k),d(j,k)}$$</td>
</tr>
<tr>
<td>Complete-link</td>
<td>$$a_i = a_j = 0.5; \quad b = 0; \quad c = 0.5 \quad d(i+j,k) = \max {d(i,k),d(j,k)}$$</td>
</tr>
<tr>
<td>Centroid</td>
<td>$$a_i = \frac{n_i}{n_i + n_j}; \quad a_j = \frac{n_j}{n_i + n_j}; \quad b = -\frac{n_{ij}}{n_i + n_j}; \quad c = 0 \quad d(i+j,k) = d(c_i,c_j)$$</td>
</tr>
<tr>
<td>Median</td>
<td>$$a_i = a_j = 0.5; \quad b = -0.25 \quad c = 0$$</td>
</tr>
<tr>
<td>(Average link)</td>
<td>$$a_i = \frac{n_i}{n_i + n_j}; \quad a_j = \frac{n_j}{n_i + n_j}; \quad b = c = 0 \quad d(i,j,k) = d(c_i,c_j) = \frac{1}{n_{ij}} \sum_{b=1}^{n_{ij}} d(a,b)$$</td>
</tr>
<tr>
<td>Ward's Method (minimum variance)</td>
<td>$$a_i = \frac{n_i}{n_i + n_j}; \quad a_j = \frac{n_j}{n_i + n_j}; \quad b = -\frac{n_{ij}}{n_i + n_j}; \quad c = 0 \quad d(i,j,k) = d(c_i,c_j)$$</td>
</tr>
</tbody>
</table>

From Single Clustering to Ensemble Methods

April 2009

Unsupervised Learning – Ana Fred
Unsupervised Learning

Clustering Algorithms

Single Linkage:

\[ d(C_i, C_j) = \min_{a \in C_i, b \in C_j} \{d(a, b)\} \]

Complete-Link:

\[ d(C_i, C_j) = \max_{a \in C_i, b \in C_j} \{d(a, b)\} \]
Complete-Link: \[ d(C_i, C_j) = \max_{a \in C_i, b \in C_j} \{d(a, b)\} \]

Dendrogram

Single-link and Complete-Link

A clustering of the data objects is obtained by cutting the dendrogram at the desired level, then each connected component forms a cluster.

Favours connectedness

Favours compactness
**Single-link and Complete-Link**

- **SL algorithm:**
  - Favors connectedness
  - Equivalent to building a MST and cutting at weak links

*Single-link method, th=0.49*
Single-link and Complete-Link

**SL algorithm:**
- Favors connectedness
- Detects arbitrary-shaped clusters with even densities
- Cannot handle distinct density clusters
- Is sensitive to in-between patterns

**CL algorithm:**
- Favors compactness
Single-link and Cor

- **SL algorithm:**
  - Favors connectedness
  - Detects arbitrary-shaped clusters with even densities
  - Cannot handle distinct density clusters
  - Is sensitive to in-between patterns
  - Needs criteria to set the final number of clusters

- **CL algorithm:**
  - Favors compactness
  - Imposes spherical-shaped clusters on data

---

Single-link and Complete-Link

- **SL algorithm:**
  - Favors connectedness
  - Detects arbitrary-shaped clusters with even densities
  - Cannot handle distinct density clusters
  - Is sensitive to in-between patterns
  - Needs criteria to set the final number of clusters

- **CL algorithm:**
  - Favors compactness
  - Imposes spherical-shaped clusters on data
  - Needs criteria to set the final number of clusters
Hierarchical Clustering

- Weakness
  - do not scale well: time complexity of $O(n^2)$, where $n$ is the number of total objects
  - can never undo what was done previously

- Integration of hierarchical with distance-based clustering
  - BIRCH (Zhang, Ramakrishnan, Livny, 1996): uses a ClusteringFeature-tree and incrementally adjusts the quality of sub-clusters
  - CURE (Guha, Rastogi & Shim, 1998): selects well-scattered points from the cluster and then shrinks them towards the center of the cluster by a specified fraction
  - CHAMELEON (G. Karypis, E.H. Han and V. Kumar, 1999): hierarchical clustering using dynamic modeling
    1. Use a graph partitioning algorithm: cluster objects into a large number of relatively small sub-clusters
    2. Use an agglomerative hierarchical clustering algorithm: find the genuine clusters by repeatedly combining these sub-clusters

Clustering Based on Dissimilarity Increments Criteria

- Smoothness Hypothesis:
  - A cluster is a set of patterns sharing important characteristics in a given context
  - A dissimilarity measure encapsulates the notion of pattern resemblance
  - Higher resemblance patterns are more likely to belong to the same cluster and should be associated first
  - Dissimilarity between neighboring patterns within a cluster should not occur with abrupt changes
  - The merging of well separated clusters results in abrupt changes in dissimilarity values

Clustering Based on Dissimilarity Increments Criteria

Dissimilarity Increments:

\( (x_i, x_j, x_k) \) - nearest neighbors

\[ x_j : j = \arg \min_i \{d(x_i, x_j), l \neq i \} \]

\[ x_k : k = \arg \min_i \{d(x_i, x_j), l \neq i \} \]

Dissimilarity increment:

\[ d_{inc}(x_i, x_j, x_k) = |d(x_i, x_j) - d(x_j, x_k)| \]

Distribution of Dissimilarity Increments:

- Uniformly distributed data
Clustering Based on Dissimilarity Increments Criteria

Distribution of Dissimilarity Increments:

- 2D Gaussian data

- Ring-shaped data
Clustering Based on Dissimilarity Increments Criteria

Distribution of Dissimilarity Increments:

- Directional expanding data

Distribution of Dissimilarity Increments: Exponential distribution: 

\[ p(x) = \beta \exp(-\beta x) \]
Clustering Based on Dissimilarity Increments Criteria

- **Exponential distribution:**
  - Higher density patterns $\Rightarrow$ higher $\beta$
  - Well separated clusters $\Rightarrow d_{\text{inc}}$ on the tail of $p(x)$

\[
\text{Gap between clusters: } \text{gap}_i = d_i(C_i, C_j) - d_i(C_i)
\]

**Diagram:**
- Blue line: 700 samples, Gaussian
- Green line: 200 samples, Gaussian
- Red line: 700 samples, uniform
- Black line: 200 samples, uniform
Clustering Based on Dissimilarity Increments Criteria

$\text{Gap between clusters: } \text{gap}_i = d(C_i, C_j) - d_i(C_i)$

Cluster Isolation criterion:

Let $C_i, C_k$ be two clusters which are candidates for merging, and let $\mu_i, \mu_k$ be the respective mean values of the dissimilarity increments in each cluster. Compute the increments for each cluster, $\text{gap}_i$ and $\text{gap}_k$. If $\text{gap}_i \geq \alpha \mu_i$ (or $\text{gap}_k \geq \alpha \mu_k$), isolate cluster $C_i$ ($C_k$) and proceed the clustering strategy with the remaining patterns. If neither cluster exceeds the gap limit, merge them.
Clustering Based on Dissimilarity Increments Criteria

**Setting the Isolation Criterion Parameter** $\alpha$:

- Result: the crossings of the tangential line, at points which are multiple of the distribution mean value, $\alpha/\beta$, with the $x$ axis, is given by $(\alpha + 1)/\beta$.

\[ \alpha \in [3, 5] \text{ cover the significant part of the distribution} \]

**Hierarchical Clustering Algorithm:**

- A statistic of the dissimilarity increments within a cluster is maintained and updated during cluster merging.

- Clusters are obtained by comparing dissimilarity increments with a dynamic threshold, $\alpha \mu_i$, based on cluster statistics.
Clustering Based on Dissimilarity Increments Criteria

Results:

- Ring-Shaped Clusters

Single-link method, \( th=0.49 \)
Clustering Based on Dissimilarity Increments Criteria

Results:
- Ring-Shaped Clusters

Dissimilarity Increments-base method:

- \( d_1 \)
- \( d_2 \)
- \( g_1 \)
- \( g_2 \)

From Single Clustering to Ensemble Methods - April 2009
Clustering Based on Dissimilarity Increments Criteria

Results:

2-D Patterns with Complex Structure

<table>
<thead>
<tr>
<th>Dissimilarity Increments-base method</th>
<th>Single-link</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Diagram 1" /></td>
<td><img src="image2.png" alt="Diagram 2" /></td>
</tr>
</tbody>
</table>

Clustering of Contour Images

- The data set is composed by 634 contour images of 15 types of hardware tools: 11 to 115.
- When counting each pose as a distinct sub-class in the object type, we obtain a total of 24 classes.
Clustering of Contour Images

Contour extraction

- the object boundary is sampled at 50 equally spaced points
- the angle between consecutive segments is quantized in 8 levels.

<table>
<thead>
<tr>
<th>Class</th>
<th>Tool</th>
<th>Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><img src="image1.png" alt="Image" /></td>
<td>61076210626767676706626307610760676706320262 sequentially</td>
</tr>
<tr>
<td></td>
<td></td>
<td>71176266262616270616706260761073617651635767262 sequentially</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7117626626262610026267376706756622026262620182 sequentially</td>
</tr>
<tr>
<td>2</td>
<td><img src="image2.png" alt="Image" /></td>
<td>000800008600000017108800060681000000075088000003 sequentially</td>
</tr>
<tr>
<td></td>
<td></td>
<td>00000000850000000086000081000000008000000000 sequentially</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0000000089000017108800060681000000075088000003 sequentially</td>
</tr>
<tr>
<td>3</td>
<td><img src="image3.png" alt="Image" /></td>
<td>1080000096880000070780000808500008012 sequentially</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10800000968800000610700070780000008500000012 sequentially</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10800000968800000710700070780000008500000012 sequentially</td>
</tr>
</tbody>
</table>
Clustering of Contour Images

Single-link vs. Dissimilarity Increments-based method

(string-edit distance)

Clustering Based on Dissimilarity Increments Criteria

Description:
- Hierarchical agglomerative algorithm adopting a cluster isolation criterion based on dissimilarity increments

Strength:
- Method is not conditioned by a particular dissimilarity measure (examples used Euclidean and string-edit distances)
- Ability to identify arbitrary shaped and sized clusters
- The number of clusters is intrinsically found

Weakness:
- Sensitive to in-between points connecting touching clusters
Unsupervised Learning
Clustering Algorithms

Outline

- **Partitional Methods**
  - K-Means
  - Spectral Clustering
  - EM-based Gaussian Mixture Decomposition

Part 3.: Validation of clustering solutions
- Cluster Validity Measures

Part 4.: Ensemble Methods
- Evidence Accumulation Clustering

Partitional Methods

- **K-Means**
  - Minimizes the cost function:
    \[ H_{KM} = \sum_{i=1}^{n} ||x_i - y_{c_i}||^2 \]
  - Algorithm:
    1. Input: \( k \), the number of clusters; data set
    2. Randomly select \( k \) seed points from the data set, and take them as initial centroids.
    3. Partition the data into \( k \) clusters by assigning each object to the cluster with the nearest centroid.
      \[ c_i = \arg\min_{\nu \in \{1,...,k\}} ||x_i - y_\nu||^2 \]
    4. Compute centroids of the clusters of the current partition. The centroid is the center (mean point) of the cluster.
      \[ y_\nu = \frac{1}{n_\nu} \sum_{i \in \nu} x_i \text{ with } n_\nu = |\{i : c_i = \nu\}| \]
    5. Go back to step 2 or stop when no more new assignment.
Partitional Methods: K-Means

- Favors compactness

Strength:
- Fast algorithm \(O(tkn)\) – \(t\) is the number
- Scalability
- Often terminates at a local optimum

Weakness:
- Imposes spherical-shaped clusters

Strength:
- Fast algorithm \(O(tkn)\) – \(t\) is the number
- Scalability
- Often terminates at a local optimum

Weakness:
- Imposes spherical-shaped clusters
Partitional Methods: K-Means

- Favors compactness

- Strength
  - Fast algorithm ($O(ktn)$ – $t$ is the number of iterations; normally, $k, t << n$)
  - Scalability
  - Often terminates at a local optimum.

- Weakness
  - Imposes spherical-shaped clusters
  - Is sensitive to the number of objects in clusters
  - Dependence on initialization
Unsupervised Learning

Clustering Algorithms

Unsupervised Learning

Ana Fred

Partitional Methods: K-Means

- Favors compactness

- Strength
  - Fast algorithm \( O(tkn) \) – \( t \) is the number of iterations; normally, \( k, t \ll n \).
  - Scalability
  - Often terminates at a local optimum.

- Weakness
  - Imposes spherical-shaped clusters
  - Is sensitive to the number of objects in clusters
  - Dependence on initialization
  - Needs criteria to set the final number of clusters
  - Applicable only when mean is defined (what about categorical data?)
Variations of K-Means Method

- K-Means (MacQueen’67): each cluster is represented by the center of the cluster
- A few variants of the k-means which differ in:
  - Selection of the initial k means
  - Dissimilarity calculations (Mahalanobis distance -> elliptic clusters)
  - Strategies to calculate cluster means
    - Medoid - each cluster is represented by one of the objects in the cluster
    - Fuzzy version: Fuzzy K-Means

- Handling categorical data: k-modes (Huang’98)
  - Replacing means of clusters with modes
  - Using new dissimilarity measures to deal with categorical objects
  - Using a frequency-based method to update modes of clusters

Spectral Clustering

- For a given data set, $X$, Spectral Clustering finds a set of data clusters on the basis of spectral analysis of a similarity graph.
- The clustering problem is defined in terms of a complete graph, $G$, with vertices $V=\{1, \ldots, N\}$, corresponding to the data points in the data set, and each edge between two vertices is weighted by the similarity between them.
- The weight matrix is also called the affinity matrix or the similarity matrix.

$$A_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right)$$

Gaussian Kernel
Spectral Clustering

- Cutting edges of $G$ we obtain disjoint subgraphs of $G$ as the clusters of $X$.
- The goal of clustering is to organize the dataset into disjoint subsets with high intra-cluster similarity and low inter-cluster similarity.
  - The resulting clusters should be as compact and isolated as possible.

The graph partitioning for data clustering can be interpreted as a minimization problem of an objective function, in which the compactness and isolation are quantified by the subset sums of edge weights.

**Common objective functions**
- Ratio cut ($R_{cut}$)
- Normalised cut ($N_{cut}$)
- Min-max cut ($M_{cut}$)

$$R_{cut}(C_1, \ldots, C_k) = \sum_{i=1}^{k-1} \frac{\text{cut}(C_i, X \setminus C_i)}{\text{card} C_i}$$
$$N_{cut}(C_1, \ldots, C_k) = \sum_{i=1}^{k-1} \frac{\text{cut}(C_i, X \setminus C_i)}{\text{cut}(C_i, X)}$$
$$M_{cut}(C_1, \ldots, C_k) = \sum_{i=1}^{k-1} \frac{\text{cut}(C_i, X \setminus C_i)}{\text{cut}(C_i, C_j)}$$

- $\text{cut}(A, B)$ is the sum of the edge weights between $v \in A$ and $v \in B$.
- $X \setminus C_i$ is the complement of $C_i$ in $X$.
- $\text{card} C_i$ denotes the number of points in $C_i$.
Spectral Clustering

The solution of the minimization problem of any of the previous objective functions is obtained from the matrix of the first $k$ eigenvectors of a matrix derived from the affinity matrix (Laplacian matrix):

- The eigenvectors for $Ncut$ and $Mcut$ are identical, and obtained from the symmetrical Laplacian

$L = I_{sym} - D^{-1/2}AD^{-1/2}$

- $D$ is a diagonal matrix whose $i$-th entry is the sum of the $i$-th row of $A$

Another common choice is $L = I_{sym} - D^2(D-A)$

Distinct algorithms differ on the way of producing and using the eigenvectors and how to derive clusters from them:

- Some use each eigenvector one at a time
- Other, use top $k$ eigenvectors simultaneously

Closely related with spectral graph partitioning, in which the second eigenvector of a graph’s Laplacian is used to define a semi-optimal cut; the second eigenvector solves a relaxation of an NP-hard discrete graph partitioning problem, giving an approximation to the optimal cut.

---

Spectral Clustering (Ng et al, 2001)

Maps the feature space into a new space, $Y$, based on the eigenvectors of a matrix derived from an affinity matrix associated with the data set.

The data partition is obtained by applying the K-means algorithm on the new space.

$(NG. \text{ and Al.} 2001)$

$A_{ij} = e^{-\frac{(Y_i - Y_j)^2}{2\sigma^2}}$

Original feature space $\rightarrow$ Eigen vector space

A. Y. Ng and M. I. Jordan and Y. Weiss, On Spectral Clustering: Analysis and an algorithm, NIPS 2001
Spectral Clustering (Ng et al, 2001)

Algorithm:

Given a set of points $S = \{s_1, \ldots, s_n\}$ in $\mathbb{R}^d$ that we want to cluster into $k$ subsets:

1. Form the affinity matrix $A \in \mathbb{R}^{n \times n}$ defined by $A_{ij} = \exp(-\|s_i - s_j\|^2/2\sigma^2)$ if $i \neq j$, and $A_{ii} = 0$.
2. Define $D$ to be the diagonal matrix whose $(i, i)$-element is the sum of $A$'s $i$-th row, and construct the matrix $L = D^{-1/2}AD^{-1/2}$.
3. Find $x_1, x_2, \ldots, x_k$, the $k$ largest eigenvectors of $L$ (chosen to be orthogonal to each other in the case of repeated eigenvalues), and form the matrix $X = [x_1, x_2, \ldots, x_k] \in \mathbb{R}^{n \times k}$ by stacking the eigenvectors in columns.
4. Form the matrix $Y$ from $X$ by renormalizing each of $X$’s rows to have unit length (i.e., $Y_{ij} = X_{ij}/(\sum_j X_{ij}^2)^{1/2}$).
5. Treating each row of $Y$ as a point in $\mathbb{R}^k$, cluster them into $k$ clusters via $K$-means or any other algorithm (that attempts to minimize distortion).
6. Finally, assign the original point $s_i$ to cluster $j$ if and only if row $i$ of the matrix $Y$ was assigned to cluster $j$. 

---

(a) NIPS 2001
(b) Original clusters
(c) Spectral clusters
(d) Original projection
(e) Spectral projection
(f) Spectral clusters projection
Spectral Clustering (Ng et al, 2001)

Results strongly depend on parameter values: $k$ e $\sigma$

$K=2$, $\sigma=0.1$

$K=2$, $\sigma=0.4$
Spectral Clustering (Ng et al, 2001)

Results strongly depend on parameter values: $k$ e $\sigma$

- $K=3$, $\sigma=0.1$
- $K=3$, $\sigma=0.45$
**Spectral Clustering**

Selection of parameter values:

- **MSE:**

\[
MSE_{y} = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{n} \sqrt{(y_i' - m_j)^2}
\]

- **Eigengap**

\[
\delta(A) = 1 - \frac{\lambda_2}{\lambda_1}
\]

- **Rcut**

\[
R_{cut}^{K} = \frac{\sum_{k=1}^{K} \sum_{d=1, d \neq k}^{K} \sum_{j \in S_d} \sum_{j \in S_k} A_{jk}}{\sum_{d=1}^{K} \sum_{j \in S_d} A_{jj}}
\]

**Selection of Parameters: Global Results on selecting \( \sigma \) and \( K \)**

None of the studied methods is suitable for the automatic selection of the spectral clustering parameters.

A majority voting decision did not significantly improve the results.

Percentage of correct classification:

- \( \sigma \)
Spectral Clustering

**Strength**
- Detects arbitrary-shaped clusters.
- By using an adequate similarity measure between patterns, can be applied to all types of data.

**Weakness**
- Computationally heavy
- Needs criteria to set the final number of clusters and scaling factor

Model-Based Clustering: Finite Mixtures

- *k* random sources, with probability density functions $f_i(x)$, $i=1,...,k$

Choose at random

$X \rightarrow$ random variable

Conditional: $f(x|source\ i) = f_i(x)$

Joint: $f(x$ and source $i) = f_i(x) \alpha_i$

Unconditional: $f(x) = \sum_{all\ sources} f(x$ and source $i) = \sum_{i=1}^{k} \alpha_i f_i(x)$
Model-Based Clustering: Finite Mixtures

- Each component models one cluster
- Clustering = mixture fitting

\[ f(x|\Theta) = \sum_{i=1}^{k} \alpha_i f(x|\theta_i) \]

Gaussian Mixture Decomposition

- Mixture Model:

\[ f(x|\Theta) = \sum_{i=1}^{k} \alpha_i f(x|\theta_i) \]

Component densities

- Mixing probabilities: \( \alpha_i \geq 0 \) and \( \sum_{i=1}^{k} \alpha_i = 1 \)

- Gaussian

  - Arbitrary covariances: \( f(x|\theta_i) = N(x|\mu_i, C_i) \)

    \[ \Theta = \{ \mu_1, \mu_2, \ldots, \mu_k, C_1, C_2, \ldots, C_k, \alpha_1, \alpha_2, \ldots, \alpha_{k-1} \} \]

  - Common covariance: \( f(x|\theta_i) = N(x|\mu_i, C) \)

    \[ \Theta = \{ \mu_1, \mu_2, \ldots, \mu_k, C, \alpha_1, \alpha_2, \ldots, \alpha_{k-1} \} \]
Mixture Model Fitting

- n independent observations \( x = \{ x^{(1)}, x^{(2)}, \ldots, x^{(n)} \} \)
- Mixture density model: \( f(x|\Theta) = \sum_{i=1}^{k} \alpha_i f(x|\theta_i) \)
- Estimate \( \Theta \) that maximizes (log)likelihood (ML estimate of \( \Theta \)):
  \[ \hat{\Theta} = \arg \max_{\Theta} L(x, \Theta) \]

\[
L(x, \Theta) = \log \prod_{j=1}^{n} f(x^{(j)} | \Theta) = \sum_{j=1}^{n} \log \sum_{i=1}^{k} \alpha_i f(x^{(j)} | \theta_i)
\]

Gaussian Mixture Model Fitting

- Problem: the likelihood function is unbounded as \( \det(C_i) \to 0 \)
  - There is no global maximum
  - Unusual goal: a “good” local maximum
- Example: a 2-component Gaussian mixture
  \[
f(x | \mu_1, \mu_2, \sigma^2, \alpha) = \frac{\alpha}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu_1)^2}{2\sigma^2}} + \frac{1-\alpha}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu_2)^2}{2\sigma^2}}
\]

some data points \( \{ x_1, x_2, \ldots, x_n \} \)

\[
\mu_1 = x_1
\]

\[
L(x, \Theta) = \log \left( \frac{\alpha}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_1-\mu_1)^2}{2\sigma^2}} + \frac{1-\alpha}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x_1-\mu_2)^2}{2\sigma^2}} \right) + \sum_{j=2}^{n} \log(...)
\]

\( \to \infty \), as \( \sigma^2 \to 0 \)
Mixture Model Fitting

- ML estimate has no closed-form solution
- Standard alternative: expectation-maximization (EM) algorithm:
  - Missing data problem:
    - Observed data: \( x = \{ x^{(1)}, x^{(2)}, ..., x^{(n)} \} \)

Missing labels ("colors")

\[ z^{(1)} = \begin{bmatrix} z_i^{(1)} & z_i^{(2)} & ... & z_i^{(K)} \end{bmatrix}^T = [0 \ ... \ 1 \ 0 \ ... \ 0]^T \]

"1" at position \( i \) \( \iff \) \( x_i \) generated by component \( i \)

- Complete log-likelihood function:

\[ L_c(x, z, \Theta) = \sum_{j=1}^{K} \sum_{i=1}^{n} z_i^{(j)} \log (\alpha_j f_j(x^i | \Theta)) \]
The EM Algorithm

- The E-step: compute the expected value of $L_c(x, z, \Theta)$

$$E[L_c(x, z, \Theta) | x, \hat{\Theta}^{(t)}] = Q(\Theta, \hat{\Theta}^{(t)})$$

- The M-step: update parameter estimates

$$\hat{\Theta}^{(t+1)} = \arg \max_{\Theta} Q(\Theta, \hat{\Theta}^{(t)})$$

EM-Algorithm for Mixture of Gaussian

- Iterative procedure: $\hat{\Theta}^{(0)}, \hat{\Theta}^{(1)}, ..., \hat{\Theta}^{(t)}, \hat{\Theta}^{(t+1)}, ...$

- The E-step: $w_{ij}^{(t)} = \frac{\hat{\alpha}_i f(x^{(j)} | \hat{\mu}_i^{(t)})}{ \sum_{i=1}^{n} \hat{\alpha}_i f(x^{(j)} | \hat{\mu}_i^{(t)})}$

$w_{ij}^{(t)}$ Estimate, at iteration $t$, of the probability that $x^{(j)}$ was produced by component $i$

- The M-step:

$$\hat{\alpha}_i^{(t+1)} = \frac{1}{n} \sum_{j=1}^{n} w_{ij}^{(t+1)}$$

$$\hat{\mu}_i^{(t+1)} = \frac{\sum_{j=1}^{n} w_{ij}^{(t+1)} x^{(j)}}{\sum_{j=1}^{n} w_{ij}^{(t+1)}}$$

$$C_i^{(t+1)} = \frac{\sum_{j=1}^{n} w_{ij}^{(t+1)} (x^{(j)} - \hat{\mu}_i^{(t+1)}) (x^{(j)} - \hat{\mu}_i^{(t+1)})^T}{\sum_{j=1}^{n} w_{ij}^{(t+1)}}$$
Mixture Gaussian Decomposition: Model Selection

How many components?

- The maximized likelihood never decreases when $k$ increases
- $\hat{k} = \arg \min \{ C(\hat{\Theta}(k)) \}$
- Usually:
  $C(\hat{\Theta}(k)) = -L(x, \hat{\Theta}(k)) + P(\hat{\Theta}(k))$

Criteria in this category:
- Akaike’s information criterion (AIC), Whindham and Cutler, 1992.
- Schwarz’s Bayesian inference criterion (BIC), Fraley and Raftery, 1998.

Resampling-based techniques

Parameter code-length

Given $\Theta(k)$, shortest code-length for $x$ (Shannon’s):

$$L(x | \Theta(k)) = -\log f(x | \Theta(k))$$

MDL criterion: parameter code length

Total code-length (two part code):

$$L(x, \Theta(k)) = -\log f(x | \Theta(k)) + L(\Theta(k))$$

MDL criterion:

$$\hat{\Theta}(k) = \arg \min_{\Theta(k)} \{ -\log f(x | \Theta(k)) + L(\Theta(k)) \}$$

$$L(\text{each component of } \Theta(k)) = \frac{1}{2} \log(n^*)$$

$n^* \rightarrow$ Amount of data from which the parameter is estimated
Mixture Gaussian Decomposition: Model Selection

Classical MDL: \( n' = n \)

\[
\hat{\Theta}_{(k)} = \arg \min_{\Theta_{(k)}} \left\{ -\log f(x|\Theta_{(k)}) + \frac{k}{2} \log(n) \right\}
\]

Mixtures MDL (MMDL) (Figueiredo, 2002)

\[
\hat{\Theta}_{(k)} = \arg \min_{\Theta_{(k)}} \left\{ -\log f(x|\Theta_{(k)}) + \frac{k(N_p + 1)}{2} \log(n) + \frac{N_p}{2} \sum_{m=1}^{k} \log(\alpha_m) \right\}
\]

Using EM and redefining the M-Step

\[
\beta_i = \left( \sum_{j=1}^{n} w_{ij}(j+1) - \frac{N_p}{2} \right)
\]

\[
G_{i}^{(t+1)} = \frac{\beta_i}{\sum \beta_m}
\]

This M-step may annihilate components

Np is the number of parameters of each component:

- Gaussian, arbitrary covariances: \( \text{Np} = d + d(d+1)/2 \)
- Gaussian, common covariance: \( \text{Np} = d \)


Gaussian Mixture Decomposition – EM MMDL

Examples

From Single Clustering to Ensemble Methods – April 2009
Gaussian Mixture Decomposition – EM MMDL

Examples

Strength:
- Model-based approach
- Good for Gaussian data
- Handles touching clusters

Weakness:
- Unable to detect arbitrary shaped clusters
- Dependence on initialization

Unsupervised Learning – Ana Fred

Gaussian Mixture Decomposition

Strength:
- Model-based approach
- Good for Gaussian data
- Handles touching clusters

Weakness:
- Unable to detect arbitrary shaped clusters
- Dependence on initialization
Gaussian Mixture Decomposition

It is a local (greedy) algorithm (likelihood never decreases)

=> Initialization dependent

References


