Cluster Analysis: Advanced Concepts and Algorithms

More Clustering Methods

- Prototype-based clustering
- Density-based clustering
- Graph-based clustering
- Scalable Clustering
Prototype-based Clustering

- Fuzzy c-means: objects are allowed to belong to more than one cluster
- EM (Expectation-Maximization): a cluster is modeled as a statistical distribution
- SOM (Self-Organizing Maps): clusters are constrained to have fixed relationships

Fuzzy c-means

- A collection of fuzzy clusters is a subset of all possible fuzzy subsets of the set of data points.
  - Each point is assigned a weight for each cluster. The sum of the weight for each point is 1.
  - Each cluster contains at least one point but not all points.
- Difference from K-means: instead of updating centroid and reassign points, c-means updates the centroid and the weight.
Clustering using Mixture Models

- Mixture model: using statistical distributions to model data, with each distribution corresponds to a cluster
- Maximum Likelihood principle:
  - parameters for distribution are fixed but unknown
  - best parameters are obtained by maximizing the probability of obtaining the samples observed
Likelihood Function:

likelihood function for a data set that contains $n$ points:

$$P(D \mid \theta) = \prod_{k=1}^{k=n} P(x_k \mid \theta)$$

$P(D \mid \theta)$ is called the likelihood of $\theta$ w.r.t. the set of samples $D$.

log-likelihood function: 

$$l(\theta) = \log P(D \mid \theta)$$

Distribution parameters $\hat{\theta} = \arg \max_{\theta} l(\theta)$ that maximize the log-likelihood is given by:

$$\nabla_{\theta} l = \sum_{k=1}^{k=n} \nabla_{\theta} \log P(x_k \mid \theta) = 0$$

---

Figure 9.3. 200 points from a Gaussian distribution and their log probability for different parameter values.
The EM Algorithm

Select an initial set of model parameters
Repeat
   **Expectation** Step: for each object, calculate the probability that each object belongs to each distribution
   **Maximization** Step: Given the probabilities from the expectation step, find the new estimates of the parameters that maximize the expected likelihood
Until the parameters do not change

*K*-means is a special case of EM for Gaussian distributions with equal covariance but different means

**Figure 9.5.** EM clustering of a two-dimensional point set with two clusters of differing density.

Left cluster corresponds to Gaussian distribution with center at (-4,1) and std=2
Right cluster corresponds to Gaussian distribution with center at (0,0), and std=0.5
SOM

- Centroids have a pre-determined topographic ordering relationship
- For each data point, its closest centroid and centroids in the neighborhood are updated
- Output is a set of centroids with clusters implicitly defined.

Figure 9.8. Visualization of the relationships between SOM cluster for Los Angeles Times document data set.
Density-based Clustering

- Grid-based clustering:
  - define a set of grid cells
  - assign data points to the appropriate cells and compute the density of each cell
  - eliminate cells with low density
  - merge adjacent cells to form clusters
- CLIQUE: cluster data points using a subset of attributes
- DENCLUE: use kernel density function

Figure 9.11. Example figures for subspace clustering.
**CLIQUE**

- Partition each dimension into the same number of equal width interval and partition an $m$-dimensional data space into non-overlapping rectangular units
- Repeatedly generate $k$-dimensional subspace cells from $k$-1-dimensional subspace cells and eliminate non-dense cells
- Merge adjacent cells to form clusters

**DENCLUE**

- Influence or kernel function: each data point has an influence that extends over a range
- Density function: sum of all kernel function associated with each data point

1. Square Wave Influence Function
   \[ f_{Square}(x, y) = \begin{cases} 
   0 & \text{if } d(x, y) > \sigma \\
   1 & \text{otherwise } 
   \end{cases} \]

2. Gaussian Influence Function
   \[ f_{Gauss}(x, y) = e^{-\frac{(x \cdot y)^2}{2\sigma^2}} \]
DENCLUE

- Find density attractor $x^*$ which is the local maximum of the density function
- Associate points to a density attractor $x^*$ that maximize the increase in density
- Each density attractor and data points associated with it represents a cluster
- Discard a cluster if its density attractor has a density less than $\xi$
- Combine clusters that are connected
- Points that are attracted to smaller maxima are considered outliers

Graph-based Clustering

- Graph-Based clustering uses the proximity graph
  - Start with the proximity matrix
  - Consider each data point as a node in a graph
  - Each edge between two nodes has a weight which is the proximity between the two data points
  - In the simplest case, clusters are connected components in the graph
- Graph-based clustering algorithms:
  - Chameleon
  - Shared Nearest Neighbor and SNN density
Chameleon

- Adapt to the characteristics of the data set to find the natural clusters
- Use a dynamic model to measure the similarity between clusters
  - Main property is the relative closeness and relative interconnectivity of the cluster
  - Two clusters are combined if the resulting cluster shares certain properties with the constituent clusters
  - The merging scheme preserves self-similarity
- One of the areas of application is spatial data

Characteristics of Spatial Data Sets

- Clusters are defined as densely populated regions of the space
- Clusters have arbitrary shapes, orientation, and non-uniform sizes
- Difference in densities across clusters and variation in density within clusters
- Existence of special artifacts (streaks) and noise

The clustering algorithm must address the above characteristics and also require minimal supervision.
Chameleon

- **Preprocessing Step:** Represent the Data by a Graph
  - Given a set of points, construct the k-nearest-neighbor (k-NN) graph to capture the relationship between a point and its k nearest neighbors
  - Concept of neighborhood is captured dynamically (even if region is sparse)

- **Phase 1:** Use a multilevel graph partitioning algorithm on the graph to find a large number of clusters of well-connected vertices
  - Each cluster should contain mostly points from one “true” cluster, i.e., is a sub-cluster of a “real” cluster

Chameleon

- **Phase 2:** Use Hierarchical Agglomerative Clustering to merge sub-clusters
  - Two clusters are combined if the resulting cluster shares certain properties with the constituent clusters

  - Two key properties used to model cluster similarity:
    - **Relative Interconnectivity:** Absolute interconnectivity of two clusters normalized by the internal connectivity of the clusters
    - **Relative Closeness:** Absolute closeness of two clusters normalized by the internal closeness of the clusters
**Shared Near Neighbor**

**SNN graph:** the weight of an edge is the number of shared neighbors between vertices given that the vertices are connected

![SNN graph diagram]

**SNN Density Clustering**

1. **Compute the similarity matrix**
   - This corresponds to a similarity graph with data points for nodes and edges whose weights are the similarities between data points.

2. **Sparsify the similarity matrix by keeping only the $k$ most similar neighbors**
   - This corresponds to only keeping the $k$ strongest links of the similarity graph.

3. **Construct the shared nearest neighbor graph from the sparsified similarity matrix.**
   - At this point, we could apply a similarity threshold and find the connected components to obtain the clusters (Jarvis-Patrick algorithm).

4. **Find the SNN density of each Point.**
   - Using a user specified parameters, $Eps$, find the number points that have an SNN similarity of $Eps$ or greater to each point. This is the SNN density of the point.
### SNN Density Clustering

5. **Find the core points**  
   Using a user specified parameter, $MinPts$, find the core points, i.e., all points that have an SNN density greater than $MinPts$.

6. **Form clusters from the core points**  
   If two core points are within a radius, $Eps$, of each other, they are placed in the same cluster.

7. **Discard all noise points**  
   All non-core points that are not within a radius of $Eps$ of a core point are discarded.

8. **Assign all non-noise, non-core points to clusters**  
   This can be done by assigning such points to the nearest core point.

(Note that steps 4-8 are DBSCAN)

### Scalable Clustering Algorithms

- Handles large amount of data
  - BIRCH
  - CURE
BIRCH

- Balanced Iterative Reducing and Clustering using Hierarchies
- Minimized I/O cost (1 or 2 scan)

- Clustering Feature: \( CF = (N, \overrightarrow{LS}, SS) \)
  - \( N \) – number of points in a cluster
  - \( LS \) – linear sum of points in a cluster
  - \( SS \) – square sum of points in a cluster

CF-tree

- Similar to B+-tree
- Each node corresponds to one page
- Parameters:
  - \( B \) – branching factor
  - \( T \) – threshold
- Non-leaf node – contains at most \( B \) CF entries of its child
- Leaf node – each entry’s diameter (or radius) has to be less than \( T \), number of entries per node is determined by the page size and entry size.
  - A leaf node represents a cluster made up of all the subclusters represented by its entries.
CF-tree

![CF-tree Diagram]

BIRCH Algorithm

Data

Phase 1: Load into memory by building a CF tree

Initial CF tree

Phase 2 (optional): Condense into desirable range by building a smaller CF tree

smaller CF tree

Phase 3: Global Clustering

Good Clusters

Phase 4: (optional and offline) Cluster Refining

Better Clusters
BIRCH Algorithm

- **Phase 1:**
  - Start with initial threshold and insert points into the tree
  - If run out of memory, increase threshold value, and rebuild a smaller tree by reinserting values from older tree and then other values
  - Remove outliers while rebuilding tree

- **Phase 2:**
  - Optional – prepare the tree for Phase 3
  - Removes outliers, and grouping clusters

- **Phase 3:**
  - Cluster all leaf nodes on the CF values according to an existing algorithm (e.g. agglomerative hierarchical clustering)

- **Phase 4:**
  - Optional – refine clusters
  - Perform additional passes over the dataset and reassign data points to the closest centroid
  - Recalculate the centroids and redistributing the items.
An Example of the CF-tree

Initially, the data points in one cluster.

The data arrives, and a check is made whether the size of the cluster does not exceed $T$. 
An Example of the CF-tree

If the cluster size grows too big, the cluster is split into two clusters, and the points are redistributed.

An Example of the CF-tree

At each node of the tree, the CF tree keeps information about the mean of the cluster, and the mean of the sum of squares to compute the size of the clusters efficiently.
CURE

- Uses a number of points to represent a cluster

- Representative points are found by selecting a constant number of points from a cluster and then “shrinking” them toward the center of the cluster

- Cluster similarity is the similarity of the closest pair of representative points from different clusters

CURE

- Shrinking representative points toward the center helps avoid problems with noise and outliers

- CURE is better able to handle clusters of arbitrary shapes and sizes