ST-DBSCAN: An Algorithm for Clustering Spatial-Temporal Data

Di Qin

Department of Statistics and Operations Research
Introduction

- Clustering: the process of grouping large data sets according to their similarity.

*Partitional, Hierarchical, Grid-based, Model-based and Density-based*

- Density-based clustering
  - Objects form a dense region should be grouped together
  - Search for regions of high density in a feature space that are separated by regions of lower density
Spatial- Temporal Data

- Data stored as temporal slices of the spatial dataset
- The clustering algorithms have to consider the spatial and temporal neighbors of objects.
- The algorithm can be used in geographic information systems, medical image, and weather forecasting
DBSCAN

- Designed to discover arbitrary-shaped clusters
- Eps: radius value based on a user defined distance measure
- MinPts: number of min points that should occur within Eps radius
- Core object: a point that its neighborhood of a Eps has to contain at least MinPts of other points
Problems of Existing Approaches

- Clustering spatial-temporal data
  - Distance measure to determine similarity
  - Only one distance parameter Eps
Problems of Existing Approaches

- Clustering spatial-temporal data
  - Two distance metrics
  - $A(x_1, y_1, t_1, t_2), B(x_2, y_2, t_3, t_4)$
  - Spatial value & Non-spatial value
    - $Eps1 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
    - Closeness of two points geographically
    - $Eps2 = \sqrt{(t_1 - t_3)^2 + (t_2 - t_4)^2}$
    - Similarities of temperature values
      - First retaining only the temporal neighbors and their corresponding spatial values.
Problems of Existing Approaches

- Identifying noise objects
  - $\text{noise} = \{p \in D | \forall i: p \notin C_i\}$  
    - $C_i$ is a cluster of database $D$.
  - $O_1$ and $O_2$ are noise
  - Densities are different
  - Cannot determine an appropriate $\text{Eps}$
Problems of Existing Approaches

- Identifying noise objects

(i) $\text{density\_distance\_max}(p) = \max\{\text{dist}(p,q)\mid q \in D \land \text{dist}(p,q) \leq \text{Eps}\}$,
(ii) $\text{density\_distance\_min}(p) = \min\{\text{dist}(p,q)\mid q \in D \land \text{dist}(p,q) \leq \text{Eps}\}$.

- The density distance of $p$ is defined as: $\frac{\text{density\_distance\_max}(p)}{\text{density\_distance\_min}(p)}$

\[
\text{density\_factor}(C) = 1 / \left[ \sum_{p \in C} \frac{\text{density\_distance}(p)}{|C|} \right]
\]
Problems of Existing Approaches

- Identifying adjacent clusters
- Existing approaches are adequate if clusters are distant
- The value of border objects may be very different than the value of other border objects in opposite side.
Problems of Existing Approaches

- Identifying adjacent clusters
- Comparing the average value of a cluster with new coming value.
- If the absolute difference between Cluster_Avg() and Object_Value is bigger than $\Delta \varepsilon$
- The object is not appended to the cluster
The Description of the Algorithm

- **Inputs:**
  - $D = \{O_1, O_2, \ldots, O_n\}$ Set of objects
  - $\text{Eps1}$: Max geographical coordinate (spatial) distance value
  - $\text{Eps2}$: Max non-spatial distance value
  - $\text{MinPts}$: Min number of points within $\text{Eps1}$ and $\text{Eps2}$ distance (may be $\ln(n)$, $n$ is the size of the dataset)
  - $\Delta \varepsilon$: Threshold value to be included in a cluster.

- **Outputs:**
  - $C = \{C_1, C_2, \ldots, C_k\}$ Set of clusters
The Description of the Algorithm

- Start from the first point, whether it is a core object
- Select a next point
- If the selected point does not belong to any clusters:
  - Retrieve_Neighbors(object, Eps1, Eps2)
    - The objects have a distance less than Eps1, Eps2.
      - Noise if the total number is less than MinPts
The Description of the Algorithm

- If the selected point has enough neighbors within $Eps1$ and $Eps2$:  
  - A new cluster is constructed.
- If an object is not a noise or not in a cluster, and the difference between average value of the cluster and the new coming value is smaller than $\Delta \varepsilon$, it is placed into the current cluster.
- The Retrieve_Neighbors function considers both the spatial and non-spatial values.
Application

- The task is to discover the region that have similar seawater characteristics.
- The data contains information about four seas: the Black Sea, the Marmara Sea, the Aegean Sea, and the east of the Mediterranean Sea.
The database contains weekly daytime and nighttime temperature records measured at 5340 stations between 2001 and 2004.
Application

- $\varepsilon_1 = 3$, $\varepsilon_2 = 0.5$  $\text{MinPts} = 15$
- C1 is the coldest area bordered by Ukraine and Russia
- C2 is the second coldest area
- C4 covers north of Aegean Sea
- C5 and C7 can be one cluster in winter
- C6 becomes a little small in summer
- C7 is the hottest region
Thanks