# Contents

**1 Introduction** **1**

**2 Model for trajectory clustering** **3**

**3 Trajectory Distance** **5**

3.1 Warping based distance . . . . . 5

3.1.1 DTW . . . . . 6

3.1.2 LCSS . . . . . 7

3.1.3 Pros and Cons . . . . . 8

3.2 Shape based distance . . . . . 9

3.2.1 Fréchet distance . . . . . 9

3.2.2 Hausdorff distance . . . . . 10

3.2.3 Pros and Cons . . . . . 11

3.3 HC-SIM . . . . . 11

**4 Clustering** **13**

4.1 Methods . . . . . 14

4.1.1 K-means . . . . . 14

4.1.2 DBSCAN . . . . . 15

4.1.3 Hierarchical Clustering . . . . . 17

4.2 Quality Criteria . . . . . 21

4.2.1 Silhouette Coefficient . . . . . 22

4.2.2 Pair Sets Index . . . . . 23

**5 Experiments** **25**

5.1 Ground Truth Data . . . . . 25

5.1.1 Data . . . . . 25

5.1.2 Trajectory Similarity . . . . . 28

5.1.3 Analysis of the distances . . . . . 29

5.1.4 Clusters Validation . . . . . 34

5.2 Unknown Data . . . . . 37

5.2.1 Data . . . . . 37

5.2.2 Analysis of the distances . . . . . 38

**6 Conclustions** **45**

# List of Tables


1 Internal Validation Indexes . . . . . 22
2 External Validation Indexes . . . . . 23
3 Mopsi Data structure . . . . . 28


# List of Figures


1 Trajectories with varying lengths . . . . . 2
2 Clustering Methods . . . . . 2
3 A GPS trajectory with 7076 points (Qian & Lu, 2017). . . . . 4
4 Euclidean distance . . . . . 5
5 DTW distance . . . . . 7
6 LCSS . . . . . 8
7 Fréchet distance . . . . . 9
8 Hausdorff distance . . . . . 10
9 Distances calculated by Hausdorff and Fréchet (Besse et al., 2015) 11
10 C-SIM . . . . . 12
11 DBSCAN (Su et al., 2020) . . . . . 15
12 The distance threshold $r$ defines the neighborhood of a point. (Ertöz et al., 2003) . . . . . 16
13 Hierarchical clustering models (Bian et al., 2019) . . . . . 17
14 HAC clustering (Sultana, 2020) . . . . . 19
15 Ward Linkage (Ignacio Gonzalez et al., 2017) . . . . . 20
16 Pairing by Hungarian . . . . . 24
17 Mopsi Trajectories . . . . . 26
18 Mopsi Clusters . . . . . 27
19 Overview of our process . . . . . 28
20 DTW Trajectories Clustering . . . . . 29
21 LCSS Trajectories Clustering . . . . . 30
22 Hausdorff Trajectories Clustering . . . . . 31
23 Fréchet Trajectories Clustering . . . . . 32
24 HCSIM Trajectories Clustering . . . . . 33
25 Silhouette Score depending on cluster . . . . . 34
26 Pair Sets Index based on distance . . . . . 35
27 Compare Trajectories Clusters . . . . . 36

28 Caltrain Trajectories Dataset . . . . . 37
29 Silhouette Score . . . . . 38
30 Caltrain Clusters . . . . . 39
31 Caltrain DTW Clusters . . . . . 40
32 Caltrain LCSS Clusters . . . . . 41
33 Caltrain Hausdorff Clusters . . . . . 42
34 Caltrain Frechet Clusters . . . . . 43
35 Caltrain HCSIM Clusters . . . . . 44

# List of Equations

1 Euclidean distance . . . . . 6
2 DTW distance . . . . . 6
3 LCSS Similarity . . . . . 7
4 LCSS Distance . . . . . 8
5 Fréchet distance . . . . . 9
6 Hausdorff distance . . . . . 10
7 C-SIM . . . . . 12
8 HC-SIM . . . . . 12
9 HC-SIM distance . . . . . 13
10 Single Linkage . . . . . 18
11 Complete Linkage . . . . . 18
12 Average Linkage . . . . . 19
13 Ward Linkage . . . . . 19
14 Silhouette Coefficient . . . . . 23

**Abstract**

In the past few years, there has been a significant advancement in the development of location-based positioning devices, and an increasing number of moving objects and their trajectories are being captured. Thus, it follows that the subject of moving object trajectory clustering is certain to be of prime importance to researchers working on data mining on moving objects. To give a context, we look at how development and the current trend in moving object clustering is in and then review common cluster techniques presented in the past few years. In this thesis, we start off by summarizing the basic characteristics of trajectory. Second, we examine the metrics for determining the similarity/dissimilarity of two trajectories. Thirdly, we investigate the methods and implementation processes of conventional moving object clustering methods. Finally, the validation criteria used to assess the efficacy and efficiency of clustering algorithms are explored.

# 1 Introduction

The increasing usage of integrated mobile devices integrate with GPS, WIFI and data storage hardware allows us to gather a massive quantity of data. Due to the intricacy of the collected data, extract useful information is a challenging problem to solve. Knowing principal routes that people or vehicles follow during the day can provide valuable information for analysis of mobility. For instance the presence of important routes not adequately covered by the public transport service could be highlighted by a set of trajectories, which provide information on how to improve it. Trajectory clustering is an appropriate way of analyzing trajectory data, and has been applied to road network extraction, detecting taxi fraud, data compression (M. Chen et al., 2012) etc. Additionally, trajectory clustering is used to gather temporal spatial information in the trajectory data and is widespread used in many application areas, such as motion prediction (Z. Chen et al., 2010) and traffic monitoring (Atev et al., 2006)

Trajectory data is stored in a variety of forms based on the kind of device, the velocity of the object, and even the function. For instance, a GPS device generates a trajectory, which is a locations sequence in geographical space identified by a combination of coordinates and a time stamp. Additional object-related attributes such as direction, velocity, or geographic information may be added in certain cases (Ying et al., 2011, 2010). Multidimensional data is used in many different areas, for example, to better understand animal migratory patterns by examining animal trajectories, to help forecast the weather by analyzing hurricane data, to help athletes achieve higher levels of performance, and much more.

Measurement of how similar two trajectories are is a typical approach in a wide range of applications. As a result, the measurement of distances is necessary in many tasks and applications of trajectory analysis since it enables us to judge the similarity of two trajectories. The spacing between the trajectories must be properly determined. Because trajectories are high-dimensional data (a sequence of positions on a map/a time-series of positions on a time-line) that has both spatial and temporal qualities, they must be taken into consideration when doing calculations like distance measurements. Therefore, there are a number of distance metrics used for trajectory data. E.g., the term "measure the sequence-only distance between trajectories" may include concepts such as Euclidean distance

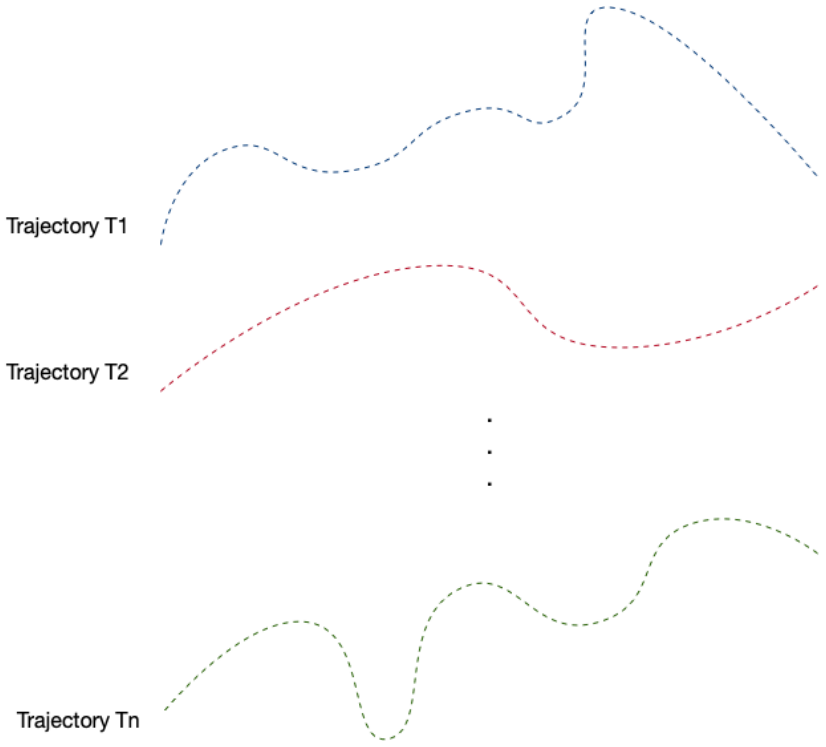


Figure 1: Trajectories with varying lengths

and Dynamic Time Wrapping Distance (DTW). Various approaches, including as clustering and classification, are often employed to identify valuable patterns from large volumes of trajectory data. Clustering is an unsupervised learning method that organizes data based on their distance (Han et al., 2011; R. Xu & Wunsch, 2005). Trajectories that occur in each cluster tend to be rather similar and distinct from those in other clusters (Berkhin, 2006; Besse et al., 2015; Yuan et al., 2017).

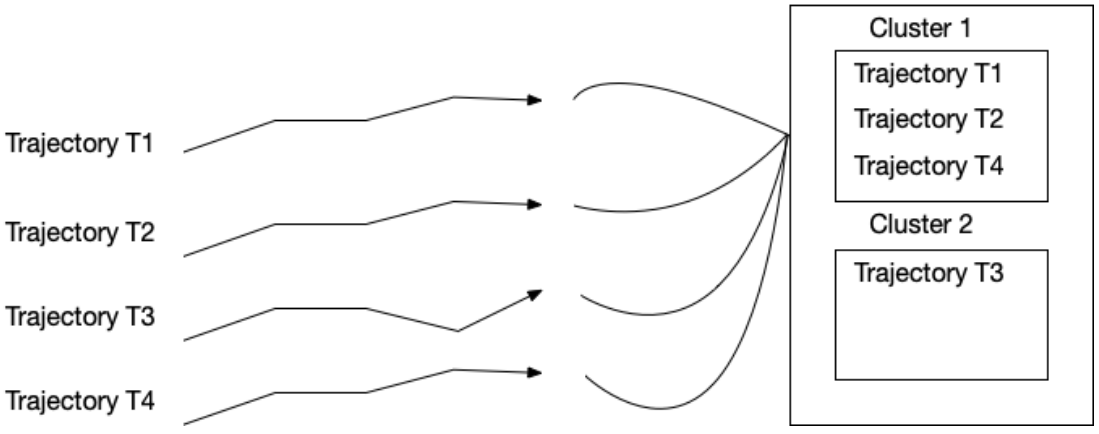


Figure 2: Clustering Methods

There are two primary methodologies used for data clustering like trajectories. Specify trajectory clustering algorithms first, then use generic clustering methods compare trajectory distance/dissimilarity. Density-based clustering (Kriegel et al., 2011) is one of the most ideal clustering methods for trajectories, since it can extract clusters of any form and is also tolerant to outliers (Ester et al., 1996). Density-Based Spatial Clustering of Applications with Noise (DBSCAN), is one of the most extensively used approaches in this group, which is commonly used (L. Zhao & Shi, 2019; Cheng et al., 2018; J. Chen et al., 2011; Lee et al., 2007). Fundamental to the clustering challenge is measuring similarity. In order to measure similarity (inverse distance), it should be done prior to grouping. The definition of distance in spatial trajectories is significantly complicated. Trajectories are non-linear sequences of points with vary length in multiple dimensions. Thus, in order to compare two trajectories a comprehensive approach is required to fully determine the distance between them. Different distance metrics are presented according to the analysis's goal and the data type. Due to the domain-specific nature of the concept of similarity, distinct distance metrics are defined to address various aspects of similarity, those are temporal, spatio-temporal, and spatial. Euclidean, Fréchet, Hausdorff, DTW, and LCSS distances are the fundamental metrics in measuring similarity from which many additional metrics are derived (Abbaspour et al., 2017; Aghabozorgi et al., 2015; Wang et al., 2013).

# 2 Model for trajectory clustering

A trajectory is a series of time-stamped data that captures the path of moving things, such as humans, cars, animals, and natural features. For example, Global Position System (GPS) tracking devices generating a trajectory as $Trajectory = (A, B, \cdots, T_n)$, which is consecutive spatial space sequence of points, and $T_i$ indicates a combination of coordinates and timestamps such as $T_i = (x_i, y_i, t_i)$.

Clustering methods are designed to aggregate similar trajectories into distinct groups. This is a challenging task due to the complicated of the trajectory, as objects within a particular region can take numerous paths. Clustering trajectories may be done in a variety of ways. We will focus on trajectory clustering based on distance, but have also investigated alternative techniques. In most situations,

trajectory data is represented as a multidimensional (2D or 3D) time series as in Figure 3; Gaffney & Smyth (1999), Vasquez & Fraichard (2004). M. Chen et al. (2012) used GPS trajectory clustering for improving data compression. The continuous definition of trajectory also applies by Gariel et al. (2011) to re-sample trajectories to achieve equal-length time series. These new trajectories are then analyzed for the main components to obtain and finally cluster the primary components. The purpose is to group trajectories that follow similar paths. As clustering is focused on trajectories, and it necessitates a new distance measurement to define the similarity between them.

There has been considerable effort put into developing different trajectory distances. Numerous approaches have been employed to cluster trajectories from a data collection. Clustering techniques based on Euclidean distance provide erroneous results due to the fact that trajectories include a variable number of points. As a result, numerous warping distance-based techniques have been established. Typical examples are distance measurements based on Dynamic Time Warping (DTW), and Longest Common Subsequence (LCSS), all of which reorder trajectories' time index in order to obtain a perfect match. Another possibility is to concentrate on the shape of the trajectories. Hausdorff and Fréchet, for example, may be adapted to trajectories. In next section, we will study and compare several distances on trajectory.

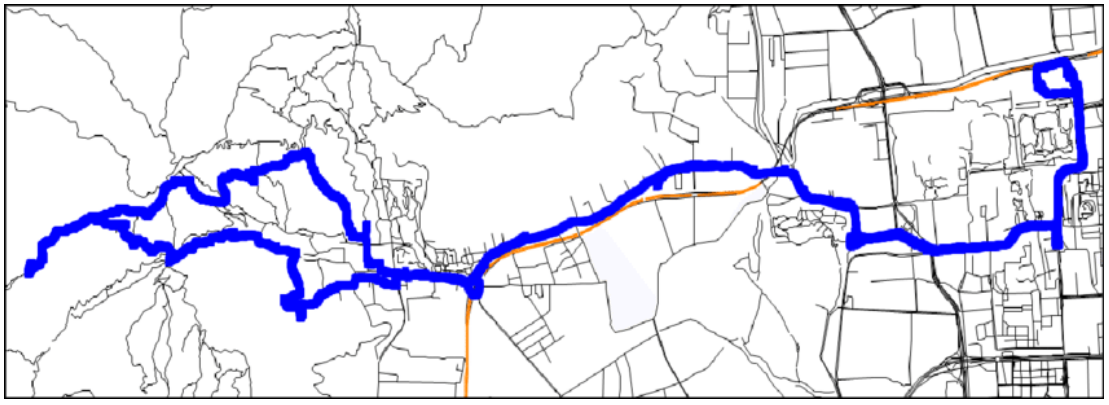

Figure 3: A GPS trajectory with 7076 points (Qian & Lu, 2017).

# 3 Trajectory Distance

A trajectory distance generalization that consideres moving objects and it measures on average how close two objects were during some time period. $d(A, B)$ represents the distance of two trajectories $A\,and\,B$. The greater the value, the less similarity between the two trajectories. Apart from the concept of mathematical distance, several measurements may be utilized to characterize this dissimilarity. Distances may be classified into two types: those based on the geometry of the trajectory (Shape-based distances) and those that consider temporal dimension (Warping based distance).

## 3.1 Warping based distance

In the 1960s, Euclidean distance was suggested to calculate distance between time series, and now, during the last several decades, has been recognized to be one of the most commonly used distance functions (Keogh & Pazzani, 2000; Faloutsos et al., 1994; Pfeifer & Deutrch, 1980; Priestley, 1980).

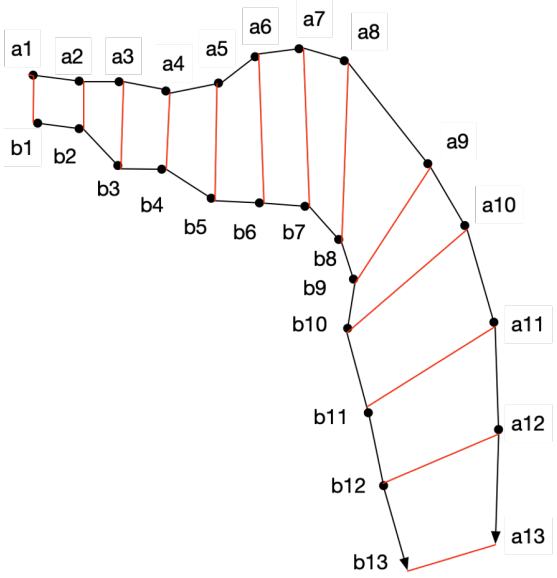


Figure 4: Euclidean distance

The Euclidean distance $d_{Euclidean}(A, B)$ between trajectories $A\,and\,B$ with the

same size $n$ is define as follows:

$$d_{Euclidean}(A, B) = \frac{\sum_{i=1}^{n} d(a_i, b_i)}{n} \tag{1}$$

Where $a_i$ and $b_i$ are the ith sample point of two trajectories $A$ and $B$. The time complexity of Euclidean distance is $O(n)$.

The Euclidean distance measure is simple to understand; however, it is only accurate if the compared trajectories are of equal size, which is not likely to be the case in practical applications. The primary objective in warping distance is to resolve this issue. The dynamic time warping algorithm (DTW) (Kruskal, 1983) and the longest common subsequence algorithm (LCSS) (Kearney & Hansen, 1990) were introduced to increase the accuracy. While both distances are specified identically, they use distinct cost functions.

### 3.1.1 DTW

*Dynamic time warping* (DTW) is similarity method for comparing two sequences. In the beginning, Myers et al. (1980) developed DTW to calculate time-series distance. In 1980s, DTW was introduced to measure trajectory distance (Kruskal, 1983) and has become one of the most extensively method for measuring trajectory distance. DTW explores all point alignments between two trajectories in search of the distance which is minimum.

Specifically, the DTW distance $d_{DTW}(A, B)$ between two trajectories $A$ and $B$ is defined as:

$$d_{DTW}(A, B) = \begin{cases} 0, if\ n\ =\ 0\ and\ m\ =\ 0 \\ \infty, if\ n\ =\ 0\ or\ m\ =\ 0 \\ d(Head(A), Head(B)) + min\{d_{DTW}(A, Rest(B)), \\ d_{DTW}(Rest(A), B), d_{DTW}(Rest(A), Rest(B)\}\ otherwise \end{cases} \tag{2}$$

With $Head(A)$ denotes $a_1$, $Rest(A)$ denotes $\{a_2, \ldots, a_n\}$ in case trajectory $A$ is consists of GPS points $\{a_1, \ldots, a_n\}$. $n$, $m$ are trajectories $A$ and $B$ lengths. DTW's time complexity is $O(mn)$.

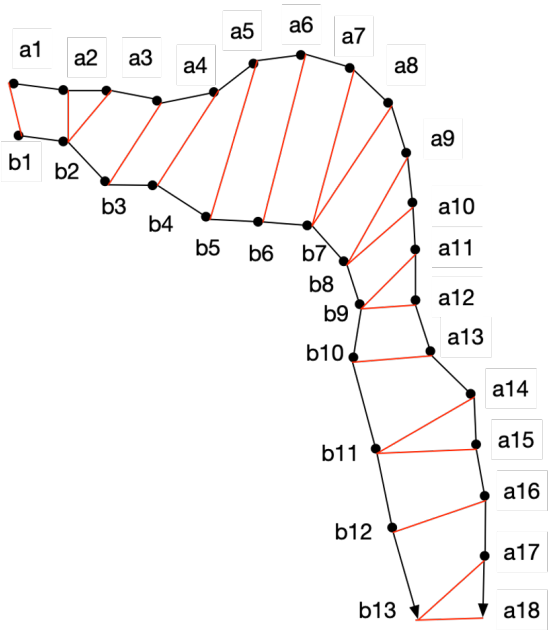


Figure 5: DTW distance

### 3.1.2 LCSS

*Longest common subsequence* (LCSS) is well-known method for measuring string similarity that searches the longest common sub-sequence between two strings. For example, for two sequences 'ABCD' and 'ACBAD', their longest sub-sequences are 'ABD' and 'ACD'. Because a trajectory may be considered as a series of sample points, LCSS was used to compare the similarity of trajectories. However, it is difficult to identify two sample points with the identical geographical information. When comparing trajectories $A\,and\,B$, LCSS considers $a_i\,(a_i \in A)\,and\,b_j\,(b_j \in B)$ to be the same if the their distance is less than a threshold $\epsilon$. Therefore, we will ignore some sample points of $A\,and\,B$ which are too far apart to contribute.

In contrast to Euclidean distance, LCSS is robust to noise. LCSS similarity is defined as:

$$\mathrm{s}_{LCSS}(A,B) = \begin{cases} 0, \text{ if } n = 0 \text{ or } m = 0 \\ 1 + \mathrm{LCSS}(\mathrm{Rest}(A), \mathrm{Rest}(B)), \text{ if } d(\mathrm{Head}(A), \mathrm{Head}(B)) \leq \varepsilon \\ \max(\mathrm{LCSS}(\mathrm{Rest}(A), B), \mathrm{LCSS}(A, \mathrm{Rest}(B))), \text{ otherwise} \end{cases} \tag{3}$$

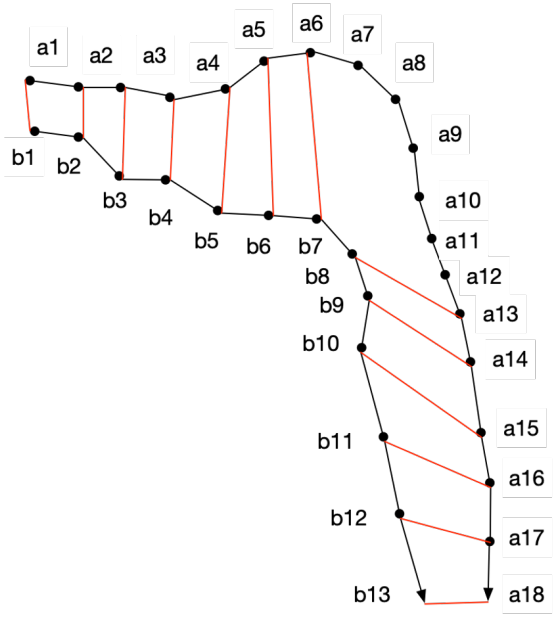


Figure 6: LCSS

LCSS is similarity so we will calculate distance:

$$d_{LCSS}(A, B) = 1 - s_{LCSS}(A, B) \tag{4}$$

### 3.1.3 Pros and Cons

DTW and LCSS enable to compare different length trajectories.

However, there are two major drawbacks to using warping-based distance:

- In general, warping methods compare sequences one-to-one so choosing a series as a reference for other sequences is necessary. Two sequences must be well-balanced in order to detect changes in time series such as when there was acceleration and deceleration. Therefore, the sequence to be referenced must be carefully selected.
- Due to the inherent noise in traffic data, traditional warping-based methods cannot obtain accurate results when examining time series.

## 3.2 Shape based distance

Instead of matching the trajectory sample points, these distances compare shape of trajectories. This indicates that trajectories are clustered regardless of their location. The most well-known distances are the Hausdorff and Fréchet.

### 3.2.1 Fréchet distance

The Fréchet distance (Eiter & Mannila, 1994) is a method measure similarity between curves consider both the location and sequence of points. The Fréchet distance between the two curves is the shortest length of leash necessary for both paths to be traversed.

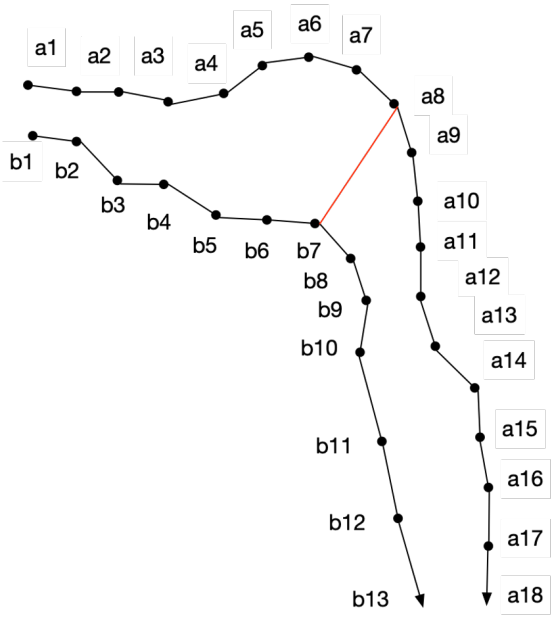


Figure 7: Fréchet distance

The Fréchet distance algorithm is shown below:

$$d_{Frechet}(A, B) = infmax_{tin[t.start,t.end]}\{d(f_a(t), f_b(t))\} \tag{5}$$

$f_a\, and\, f_b$ denote as two continuous functions of two trajectories $A\, and\, B$ over time $t$, $t.start$ and $t.end$ denote starting time and end time of time period $t$ respectively. The Fréchet distance between $A\, and\, B$ is defined as the infimum over

all reparameterizations $f_a(t)$ and $f_b(t)$. Fréchet distance is affected by variations in sampling rate and is susceptible to noise since it is the longest distance between two trajectories at the same time.

Fréchet distance's time complexity is $O(mn)$.

### 3.2.2 Hausdorff distance

The Hausdorff distance measures the distance between two sets of metric spaces. The Hausdorff distance is a metric distance which calculates the distance between two subsets of a metric space. It is the maximum distances from a point in one set to the closest point in the other set.

The Hausdorff distance algorithm is shown as:

$$d_{Hausdorff}(A, B) = max\{\sup_{a \in A} \inf_{b \in B} d(a, b), \sup_{b \in B} \inf_{a \in A} d(a, b)\} \tag{6}$$

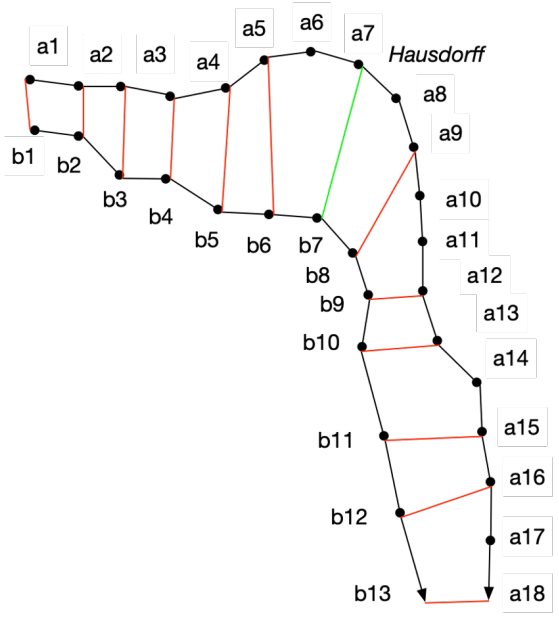


Figure 8: Hausdorff distance

The Hausdorff distance's time complexity is $O(n^2)$.

### 3.2.3 Pros and Cons

Both the Fréchet and Hausdorff distances are metrics, which means they correspond to the triangle inequality. If we want the clustering methods like DBSCAN or K-medoid to be efficient, we need this property of the distance. They've been used in a number of areas where shape comparison is required. However, it may be difficult to do a comprehensive comparison of trajectories. Both distances return the greatest distance between two trajectories at particular moments.

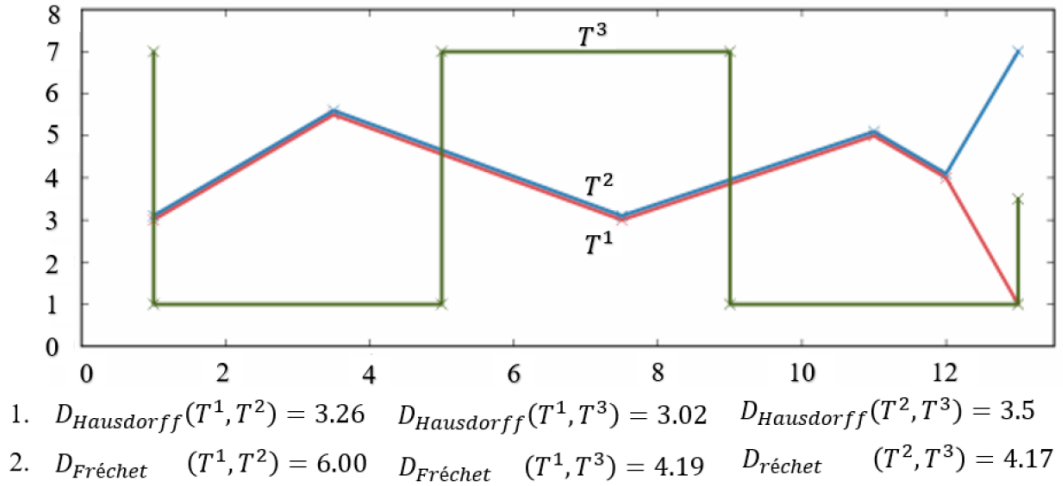


Figure 9: Distances calculated by Hausdorff and Fréchet (Besse et al., 2015)

As in Figure 9, we can see that $T^1$ and $T^2$ are the most comparable of the three trajectories, yet they are the furthest apart in Fréchet due to the maximum distance of trajectory end at 6. And the Hausdorff distances for all trajectories are nearly identical, implying poor precision. These two approaches are effective in applications involving shape comparisons, such as image comparison.

## 3.3 HC-SIM

Fränti & Mariescu-Istodor (2019) proposed a new shape-based distance, HC-SIM (Hierarchical Cell Similarity), an independent variant of C-SIM measure. We will evaluate HC-SIM in this thesis and compare it with other distances. HC-SIM are proposed as it is least impacted by sample rate changes and works well with noise.

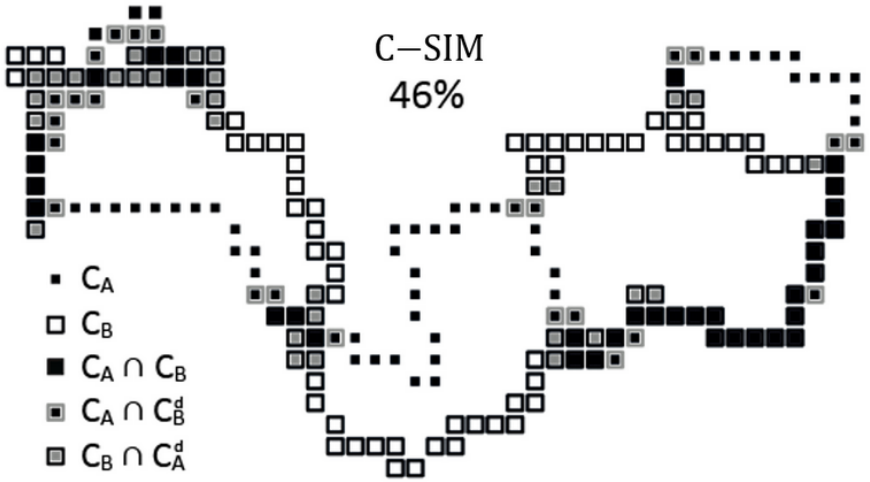


Figure 10: C-SIM

C-SIM algorithm was proposed by Mariescu-Istodor & Fränti (2017) to compute similarity of two trajectories. It computes a cell representation for the two trajectories using a grid, and determines the proportion of cells that are in common with the total cells.

**C-SIM**:

$$S\left(C_{\mathrm{A}}, C_{\mathrm{B}}\right)=\frac{\left|C_{\mathrm{A}} \cap C_{\mathrm{B}}\right|+\left|C_{\mathrm{A}} \cap C_B^d\right|+\left|C_{\mathrm{B}} \cap C_A^d\right|}{\left|C_{\mathrm{A}}\right|+\left|C_{\mathrm{B}}\right|+\left|C_{\mathrm{A}}\right|+\left|C_{\mathrm{A}} \cap C_{\mathrm{B}}\right|} \tag{7}$$

The C-SIM time complexity is $O(N_A + N_B + |C_A| + |C_B|)$ where $N_A$ and $N_B$ are the number of trajectories' points.

HC-SIM implements a grid featuring six layers (0.5%, 1%, 2%, 4%, 8%, and 16%). At each layer, we calculate how many cells shared by two segments to the total number of cells they occupy. In this thesis, we calculate HC-SIM distance through dissimilarity measure:

$$\text{HC-SIM}(A, B)=\frac{1}{L} \sum_{i=1}^{L} \text{C-SIM}(A, B) \tag{8}$$

$$d_{\text{HC-SIM}}(A, B) = 1 - \text{HC-SIM}(A, B) \tag{9}$$

# 4 Clustering

Clustering is the most common unsupervised learning method in pattern recognition. It is basically the task of grouping objects such that similar objects reside in the same group together. Clustering algorithms have gained a lot of attention among researchers and therefore, a large number of clustering algorithms have been evolved. In order to be successful with clustering, first we need to understand the nature of the objects we need to cluster. We need to examine their properties and understand how these objects are inputted to an algorithm. For instance, one may be interested to know whether the data that needs to be clustered is inputted to the algorithm incrementally. This is an interesting problem when data is collected as clusters are evolving. On the other hand, the data set can be present as a whole prior to the execution of the clustering algorithm. Additionally, the properties of the objects that need to be clustered are very important. These properties can give rise to a number of important questions such as if the objects are vector-based or if the objects are metric based. Moreover, one needs to know if objects can be transformed from one form to another so that further analysis can be done on them. One of the most important aspects of clustering is a way of comparing the objects which is widely known as a distance function.

In this section, we will explore clustering algorithms and make an attempt to show which clustering algorithm is reasonable for trajectory clustering. We first start off with some prerequisites that many clustering algorithms require and show that these requirements are met by the trajectory distance function.

## 4.1 Methods

### 4.1.1 K-means

K-means is the well-known unsupervised machine learning technique for partitioning data set into k clusters, where k is pre-defined number of clusters. It classifies objects into number of clusters that objects in the same cluster are similar while objects from other clusters are dissimilar. K-means represents each cluster by its center (also known as the centroid) which match to the mean of data points.

The primary principle underlying k-means clustering is to define clusters that sum of square errors (SSE) or total within-cluster variation is minimized.

The K-means algorithm is summarized as follows:

- Define the number of clusters (k).
- Randomly select k objects from the data set as the initial cluster centroids.
- Allocates each observation to the closest cluster.
- Compute the centroids for the clusters by calculating the mean of the all data points that belong to each cluster.
- Loop through steps 3 and 4 until the cluster assignments stay unchanged.

K-means is a simple algorithm. By using a better initilization method and restarting k-means, it can be significantly enhanced (Fränti & Sieranoja, 2019). Furthermore, it can effectively deal with extremely huge data sets. However, the k-means clustering has several drawbacks. One drawback is that we must predetermine the number of clusters. Another drawback is sensitive to outliers and ordering of the data. Additionally, (Fränti & Sieranoja, 2018) showed that when cluster size unbalance, k-means performs badly. While K-means is an excellent technique for fine-tuning at the local level, it has significant limitations when clusters do not overlap.

### 4.1.2 DBSCAN

Density-based clustering is a well-established subject of research with a straightforward concept: build a structure that properly represents the underlying density with a collection of data points (Kriegel et al., 2011). Based on this idea, Density-based spatial clustering of applications with noise (DBSCAN), is suggested by Ester et al. (1996) has been broadly applied to trajectory clustering. The basic concept is the density of the surrounding neighborhood must be greater than a certain threshold.

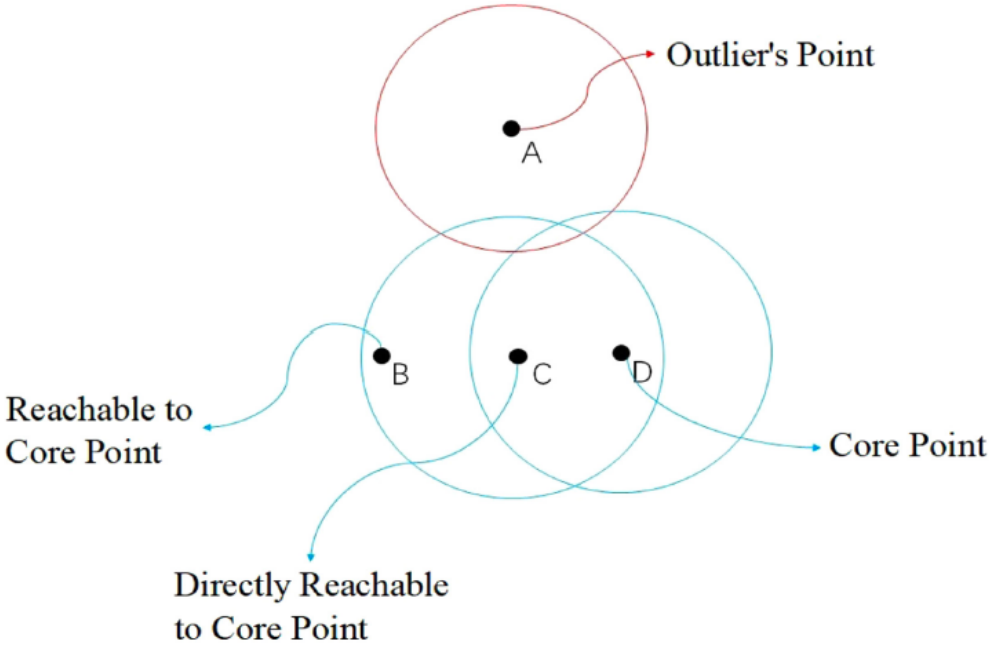


Figure 11: DBSCAN (Su et al., 2020)

The DBSCAN method (Ester et al., 1996; Kriegel et al., 2011) detects all clusters by expanding each cluster's core points to any density-reachable points. It takes an arbitrary point $p$ as its starting point and generates its $\epsilon$-neighborhood. If it is a core point, it will initiate a new cluster, which will be enlarged by include all points in its vicinity. If a neighboring core point is discovered, the search is widened to encompass all points in the vicinity. The cluster is considered full when not discovered further core points in the extended neighborhood, and scan the remaining points to whether start a new cluster. After all points have been processed, those that have not been allocated are considered as noise.

DBSCAN has difficulty when handling data sets that have clusters of varied density. (Ertöz et al., 2003). For exaple in Figure 12. If the threshold is set too low, all of the points in the dense cluster will be considered noise. If the threshold is set too high, all scores will be grouped together. An alternative option would be to repeat the algorithm many times, eliminating previously discovered clusters after each run.

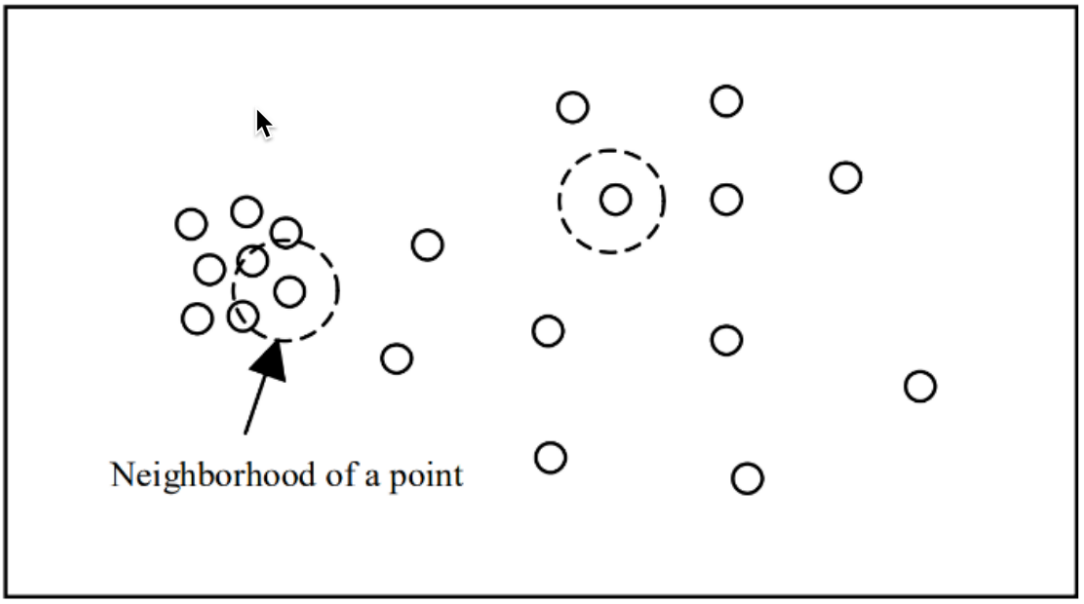


Figure 12: The distance threshold $r$ defines the neighborhood of a point. (Ertöz et al., 2003)

The clustering technique must satisfy the trajectory object's characteristics in order to be chosen. A mean trajectory cannot be easily defined since trajectories differ in length. Neither the k-means nor spectral clustering techniques are applicable to our trajectory collection. A valid metrics are required when implement an efficient algorithms like as partitioning around medoids or the DBSCAN. The majority of the investigated distances, such as LCSS and DTW, have not been classified as metrics so we will not use DBSCAN or medoid partitioning technique in this case. To cluster trajectories, we will use hierarchical cluster analysis (HCA). Actually, in HCA, only the distance/similarity matrix is required, which means objects of varied lengths can be clustered. In this thesis, we will use HCA to cluster trajectories and validate our distances.

### 4.1.3 Hierarchical Clustering

As the name suggests, hierarchical clustering algorithms (Jain et al., 1999) produce a hierarchical representation of the data. Such hierarchical data representations are normally stored in a tree data structure. Each node of the tree is considered to be a cluster. The root of the tree is at the highest level and it contains all the elements. The root of the tree can be viewed as a single cluster. Each internal node of the tree contains children and each child can be interpreted as a single cluster. Assuming that the entire dataset is given in advance, there are two basic strategies:

- Agglomerative (bottom-up)
- Divisive (top-down)

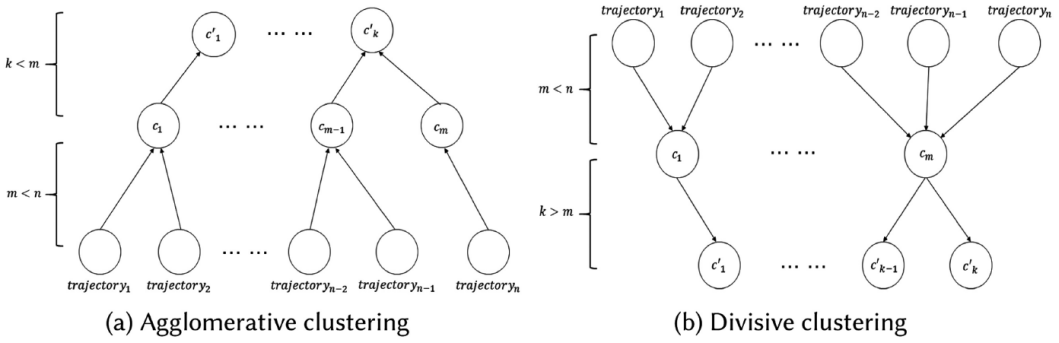


Figure 13: Hierarchical clustering models (Bian et al., 2019)

The agglomerative approach starts at the bottom of the tree. Each object is inserted to a leaf node representing a single cluster containing a single object. The algorithm proceeds by merging similar object by constructing internal nodes containing several nodes from the previous level. This is done recursively until there is a single node which defines the root of the tree.

The divisive approach works is opposite direction. Unlike the agglomerative approach, the divisive approach starts off by creating the root of the tree. The algorithm continues by dividing the entire dataset into two subsets of similar objects. This approach is also recursive. The recursion continues until the division process reaches a subset with a single node.

In this section, we focus on HAC (Hierarchical Agglomerative Clustering) as an example of the hierarchical clustering algorithm. The main idea is to give some insight of how such algorithms are designed. This is also beneficial because the reader will have an idea of what these tree data structures may look like. Assuming that we are given a dataset with $n$ objects to cluster. HAC starts by creating a cluster for each object and then it performs $n - 1$ merges. The two most similar clusters are merged together at each step. These merges form new internal nodes of the tree containing more objects. Notice that when two nodes are being merged together, the algorithm needs to have a dissimilarity measurement between clusters. This means that the distance function is not enough. However, the cluster dissimilarity measure is derived from the provided distance function. HAC clustering may use one of the following ways as shown in Figure 14 to compute the merge cost:

- Single Linkage
- Complete Linkage
- Average Linkage
- Ward Linkage

The dissimilarity measurement is a derivation of distance function. Let $G$ and $H$ be two clusters, the single linkage (SL) measurement is the least distance between any two objects, $g \in G$ and $h \in H$:

$$d_{SL}(G, H) = \min_{\substack{g \in G \\ h \in H}} d(g, h) \tag{10}$$

The complete linkage (CL) is the opposite of single linkage: complete linkage is the maximum distance betweenany two objects, $g \in G$ and $h \in H$:

$$d_{CL}(G, H) = \max_{\substack{g \in G \\ h \in H}} d(g, h) \tag{11}$$

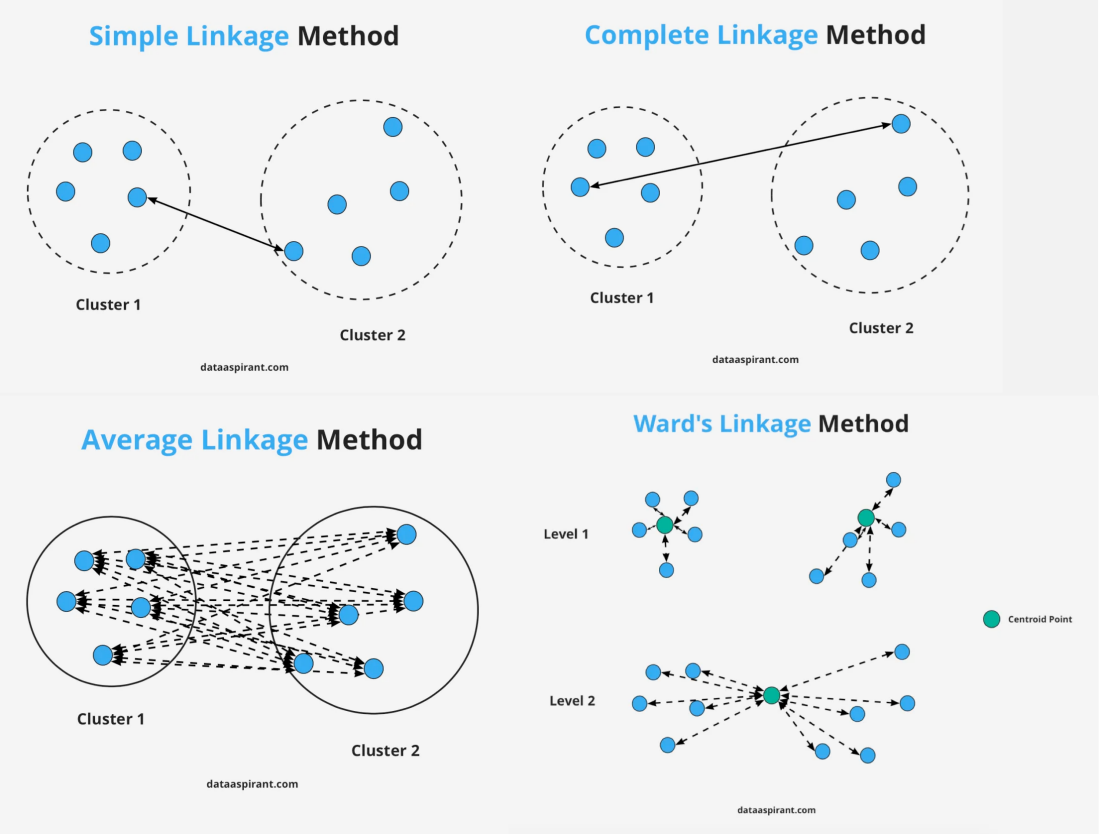


Figure 14: HAC clustering (Sultana, 2020)

The avarage linkage (AL) is the average between groups:

$$d_{AL}(G,H) = \frac{1}{|G| \times |H|} \sum_{g \in G} \sum_{h \in H} d(g,h) \tag{12}$$

Finally, in the ward linkage (WL) an error function is specified for each cluster. This error function is the average distance between each data point in a cluster and the cluster's center of gravity:

$$d_{WL}(G,H) = \frac{n_G n_H}{n_G + n_H} ||\vec{m}_G - \vec{m}_H||^2 \tag{13}$$

Where $\vec{m}_j$ is the center of cluster $j$, and $n_j$ is the number of points in it.

The one we choose to use is called Ward Linkage. In constrast the others, it evaluates cluster variance rather than calculating distance directly. Ward's is

said to be the most suitable method as it minimized sum of square error (SSE).

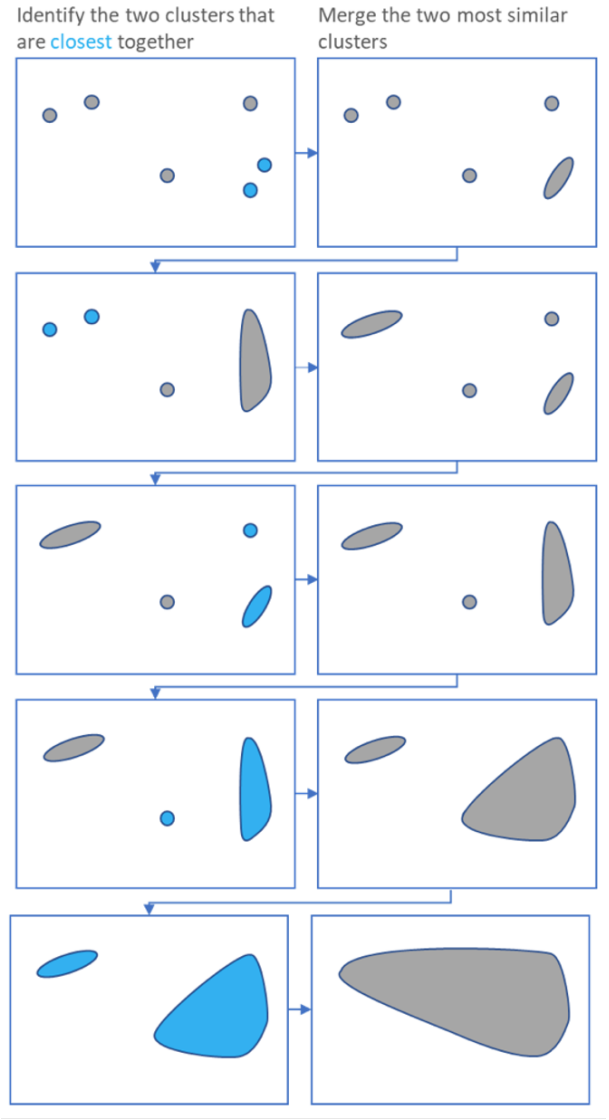


Figure 15: Ward Linkage (Ignacio Gonzalez et al., 2017)

## 4.2 Quality Criteria

Comparing the results of two clustering methods on a data set is a difficult problem in cluster analysis. Different clustering methods with varying cost functions give different solutions, and no particular clustering approach is optimal for all available data sets. Thus, a main task is to determine the optimal clustering for a particular data set. However, the question of which cluster best represents the data set arises. Cluster validity indexes have been provided to handle this problem. They are categorized into internal and external indexes of which the former are validated without external info while the latter are validated against groundtruth.

**Internal validity indexes** evaluate the quality of a clustering method without external information. There are several instances of internal validity indexes based on cluster compactness and cluster separation. Internal validation methods may be used to determine the optimal clustering methodology and cluster number without requiring any extra information. A variety of internal clustering validation metrics for clustering have been studied by many authors. Table 1 shows 13 widely used internal validation indexes.

| **Name** | **Formula** |
|---|---|
| SSW | $SSW_M = \sum_{i=1}^{N} \|x_i - c_{p_i}\|^2$ |
| SSB | $SSB_M = \sum_{i=1}^{M} n_i \left\|c_i - \bar{X}\right\|^2$ |
| Calinski-Harabasz (Caliński & Harabasz, 1974) | $CH = \frac{SSB_M/(M-1)}{SSW_M/(N-M)}$ |
| Ball&Hall (Ball & Hall, 1965) | $BH = SSW_M/M$ |
| Xu-index (L. Xu, 1997) | $Xu = D\log_2\left(\sqrt{SSW_M/\left(DN^2\right)}\right) + \log M$ |
| Krzanowski-Lai (Krzanowski & Lai, 1988) | $diff_M = (M-1)^{2/D} SSW_{M-1} -$ $M^{2/D} SSW_M$ $KL = \left\|diff_M\right\| / diff_{M+1}$ |
| Hartigan (Hartigan, 1975) | $H = \left(\frac{SSW_M}{SSW_{M+1}} - 1\right)(N - M - 1)$ or: $H = \log_2\left(SSB_M/SSW_M\right)$ |
| Dunn's index | $d\left(c_i, c_j\right) = \min_{x \in c_i, x' \in c_j} \|x - x'\|^2$ $\text{diam}\left(c_k\right) = \max_{x, x' \in c_k} \|x - x'\|^2$ $\text{Dunn} = \frac{\min_{i=1}^{M} \min_{j=i+1}^{M} d(c_i, c_j)}{\max_{k=1}^{M} \text{diam}(c_k)}$ |
| Davies & Bouldin | $R_{ij} = \frac{S_i + S_j}{d_{ij}}, i \neq j$ where: $d_{ij} = \|c_i - c_j\|^2$ |

| | |
|---|---|
| | $S_i = \frac{1}{n_i} \sum_{j=1}^{n_i} \|x_j - c_i\|^2$ and, $R_i = \max_{j=1,\ldots,M} R_{ij}, i = 1, \ldots, M$ $DBI = \frac{1}{M} \sum_{i=1}^{M} R_i$ |
| R-Square | $SSW = \sum_{\substack{k=1,\ldots,M \\ d=1,\ldots,D}} \sum_{i=1}^{n_{kd}} \left(x_i - \overline{x^d}\right)^2$ $SST = \sum_{d=1,\ldots,D} \sum_{i=1}^{n_d} \left(x_i - \overline{x^d}\right)^2$ $RS = \frac{SST - SSW}{SST}$ |
| Silhouette Score | $a\left(x_i\right) = \frac{1}{n_m - 1} \sum_{j=1, j \neq i}^{n_m} \|x_i - x_j\|^2_{x_i, x_j \in C_m}$ $b\left(x_i\right) = \min_t \left\{ \frac{1}{n_t} \sum_{j \in C_t} \|x_i - x_j\|^2 \right\}_{x_i \notin C_t}$ $s\left(x_i\right) = \frac{b(x_i) - a(x_i)}{\max(a(x_i), b(x_i))}$ $SC = \frac{1}{N} \sum_{i=1}^{N} s\left(x_i\right)$ |
| Bayesian Information Criterion (BIC) | $BIC = L * N - \frac{1}{2} M(D+1) \sum_{i=1}^{M} \log\left(n_i\right)$ |
| Xie-Beni | $XB = \frac{\sum_{i=1}^{N} \sum_{k=1}^{M} u_{ik}^2 \|x_i - C_k\|^2}{N \min_{t \neq s} \left\{ \|C_t - C_s\|^2 \right\}}$ |

Table 1: Internal Validation Indexes

**External validity indexes** evaluate how closely the clustering match the ground truth (if available) or another clustering. Table 2 shows some investigated external validation measures. They are classified as pair-counting, information-theoretic, and set matching measures. Rand index, the Jaccard coefficient, the Fowlkes-Mallows index are the pair-counting metrics. Clusters have also been compared using information theoretic indices such as entropy, Mutual Information, and variation of information. Mutual information evaluates the information shared between two clusters. F-measure and van Dongen criteria are set matching indices.

In this thesis, we will use Silhouette Coefficient as internal index and an external index introduced by Rezaei & Fränti (2016): Pair Sets Index.

### 4.2.1 Silhouette Coefficient

The Silhouette Coefficient (Rousseeuw, 1987) measures an object's similarity to its own cluster (cohesion) in comparison to other clusters (separation). The Silhouette Coefficient is defined for each sample and is composed of two scores:

Table 2: External Validation Indexes

| Name | Formula |
|---|---|
| Entropy (Y. Zhao & Karypis, 2001) | $E = -\sum_i p_i \left( \sum_j p_{ij}/p_i \log \left( p_{ij}/p_i \right) \right)$ |
| Purity (Y. Zhao & Karypis, 2001) | $P = \sum_i p_i \left( \max_j p_{ij}/p_i \right)$ |
| F-measure (Van Rijsbergen, 1979) | $F = \sum_j p_j \max_i \left[ 2 \frac{p_{ij}}{p_i} \frac{p_{ij}}{p_j} / \left( \frac{p_{ij}}{p_i} + \frac{p_{ij}}{p_j} \right) \right]$ |
| Variation of Information (Meilă, 2003) | $VI = -\sum_i p_i \log p_i - \sum_j p_j \log p_j$ $-2 \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{p_i p_j}$ |
| Rand index | $RI = \frac{\left[ \binom{n}{2} - \sum_i \binom{2^{n_i}}{} - \sum_j \binom{n_j}{2} + 2 \sum_{ij} \binom{n_{ij}}{2} \right]}{\binom{n}{2}}$ |
| Jaccard Index | $J = \frac{\sum_{ij} \binom{n_{ij}}{2}}{\left[ \sum_i \binom{n_i}{2} + \sum_j \binom{n_{ij}}{2} - \sum_{ij} \binom{n_{ij}}{2} \right]}$ |
| Mutual Information (Banerjee et al., 2005) | $MI = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{p_i p_j}$ |
| Fowlkes and Mallows (Fowlkes & Mallows, 1983) | $FM = \sum_{ij} \left( 2_2^{n_{ij}} \right) / \sqrt{\sum_i \binom{n_i}{2} \sum_j \binom{n_{ij}}{2}}$ |
| Minkowski score | $MS = \frac{\sqrt{\sum_i (2^{n_i}) + \sum_j \binom{n_{\cdot j}}{2} - 2 \sum_{ij} \binom{n_{ij}}{2}}}{\sqrt{\sum_j \binom{n_{\cdot j}}{2}}}$ |
| Goodman-Kruskal | $GK = \sum_i p_i \left( 1 - \max_j \frac{p_{ij}}{p_i} \right)$ |

- **Cohension(a):** The mean distance between a sample and all other points in the same class.
- **Separation(b):** The mean distance between a sample and all other points in the *next nearest cluster*.

$$s = \frac{b - a}{max(a, b)} \tag{14}$$

Silhouette coefficients varies between -1 and 1, as a high value indicate well-matched and negative values indicate poorly match.

### 4.2.2 Pair Sets Index

Set-matching measures comprise three questions:

1. How can the similarity of two clusters be measured?
2. How should the clusters be matched?
3. How is overall similarity measured?

Pair Sets Index (PSI) was introduced by (Rezaei & Fränti, 2016) includes optimum cluster pairing using Hungarian algorithm (Kuhn, 1955) as illustrated in Figure 16, a set matching measure based on Braun-Banquet (BB), and a chance correction.

$$PSI = \begin{cases} \frac{S-E(S)}{max(K,K')-E(S)}, if S \geq E(S), max(K,K') > 1 \\ 0, if S < E(S) \\ 1, if K = K' = 1 \end{cases} \quad (15)$$

Where: $S = \sum_{i=1}^{\min(K,K')} \frac{n_{ij}}{\max(n_i,m_j)}$ with $i, j$: paired clusters' indexes

Pair Sets Index (PSI) simply is a metric that normalized in the range of [0, 1]. 1 indicates same number of clusters.

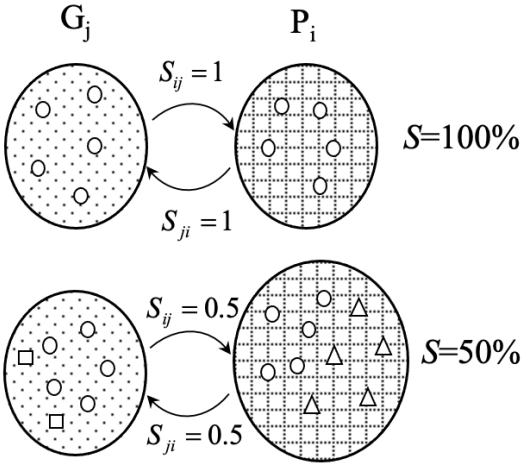


Figure 16: Pairing by Hungarian

# 5 Experiments

In this section, we analyse and evaluate 5 distances DTW, LCSS, Hausdorff, Fréchet and HCSIM. Those distances were evaluated in two dataset: one is known data, one is unknown data then we used Silhouette to detect number of clusters. For the groundtruth dataset we used Pair Sets Index to evaluate. We used Python to implement all distances and *scipy* library for *hierarchical clustering analysis*.

## 5.1 Ground Truth Data

### 5.1.1 Data

The data we used are GPS data collected from Mopsi. MOPSI is a website that enables users in finding out where their friends are, as well as what is surrounding. This website provides photos sharing, navigation, and chatting with friends (Mariescu-Istodor, 2013). Users have recorded their trajectories. Detailed information about the routes, such as speed, distance traveled, and transportation mode used (walking, running, cycling, or skiing) is shown on the map, along with these trajectories by the method in Waga et al. (2012). Mopsi provides many approaches to find trajectories. Additionally, it offers suggestions and tools for data collecting management.

Mopsi data is classified into two types: geo-tagged images and trajectory data. Geo-tagged images include information about the location and time stamp. A trajectory is a collection of GPS points that are recorded at regular intervals. These trajectories are used as data in this thesis. There were 110 trajectories which were recorded by users in Joensuu, Finland grouped into 11 groups extracted from Mopsi data as shown in Figure 17. Green points are the start points and red points are the end points when users stop their routes. Figure 18 illustrated 11 trajectories groups defined by Mopsi users. Table 3 shows trajectory data structure.

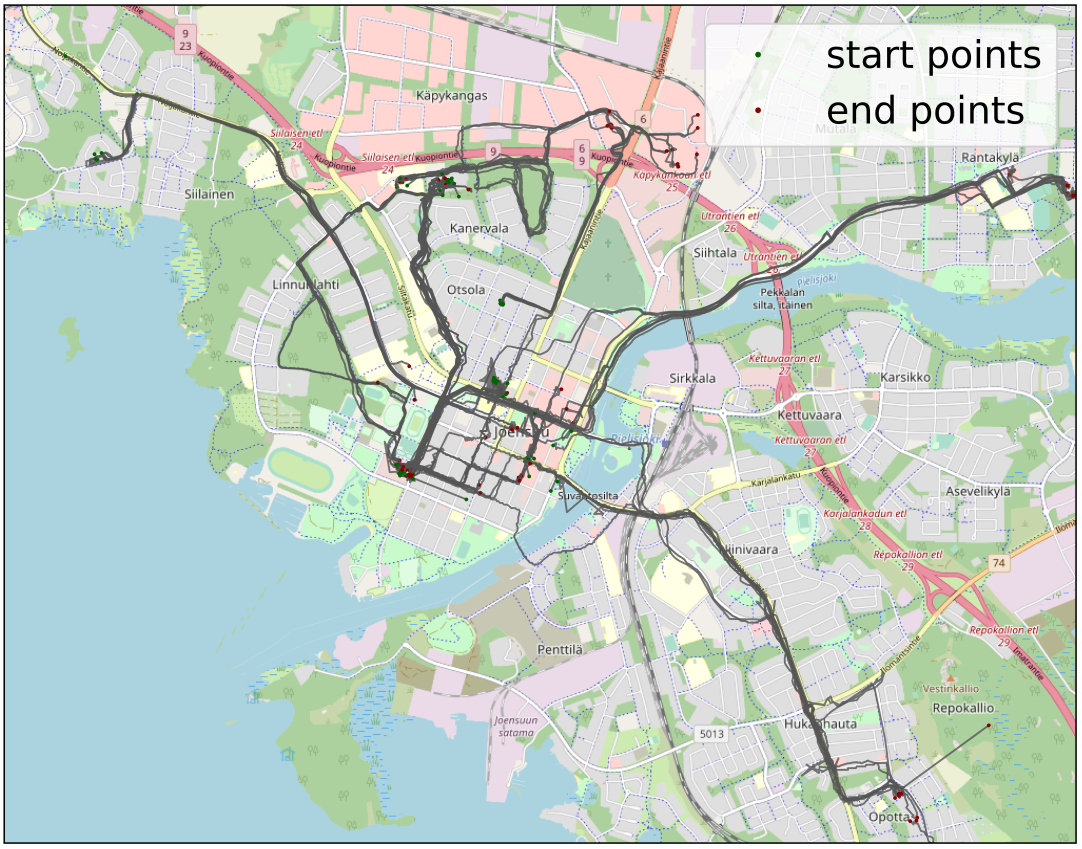


Figure 17: Mopsi Trajectories

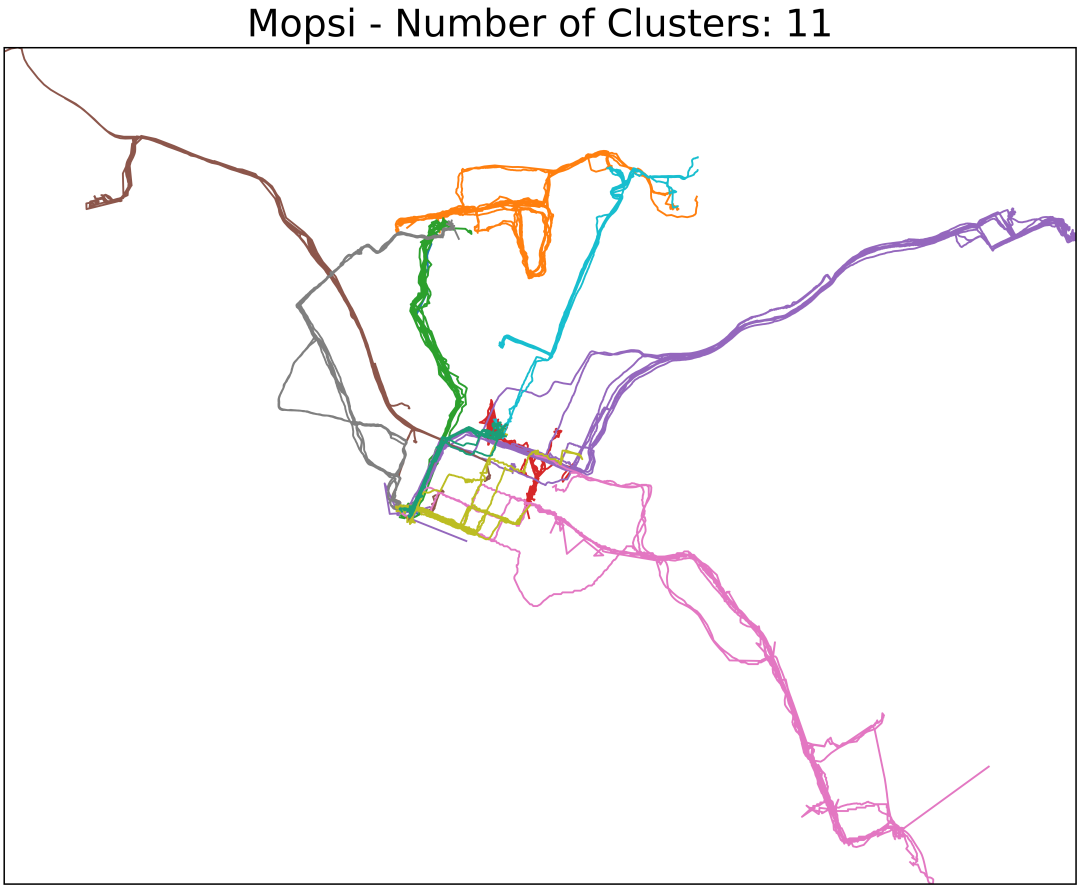


Figure 18: Mopsi Clusters

Table 3: Mopsi Data structure

| Column | Type | Description | Example |
|---|---|---|---|
| Latitude | Double | Point latitude | 62.926880 (62° 55’ 36.7674") |
| Longitude | Double | Point longitude | 23.184691 (23° 11’ 4.8876") |
| Timestamp | String | Point timestamp | 1559983789 seconds |
| Altitude | Double | Point altitude | -1.0 meter |

### 5.1.2 Trajectory Similarity

The quality of a clustering algorithm will depend on the metric used to measure similarity/dissimilarity between two trajectories. Figure 20 illustrated the process computed trajectory similarity.

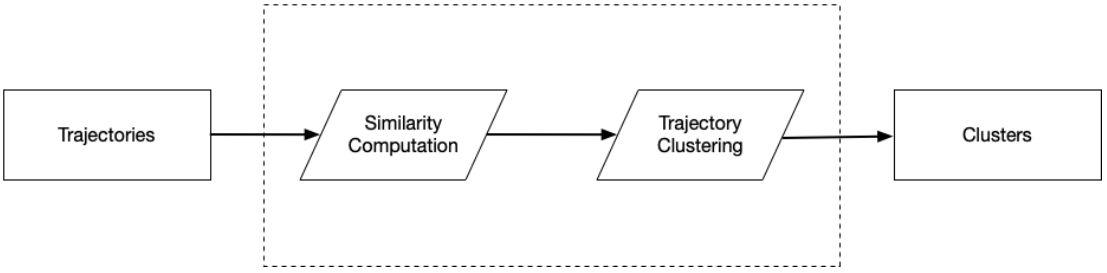


Figure 19: Overview of our process

Figure 19 shows our clustering process that consists of two components. We initially detected pointers among trajectories and then used them to calculate pair-wise distance/similarity between trajectories. With a set of trajectories with pair-wise similarity measures, we labeled trajectories based on the correlation between trajectory pairs. After that we used hierarchical clustering to obtain clusters of trajectories based on the label.

### 5.1.3 Analysis of the distances

As mentioned before, we chose Ward Linkage method on HAC (Hierarchical Agglomerative Clustering) for the clustering. Below figures are Mopsi clustering results using 5 distances LCSS, DTW, Hausdorff, Fréchet and HCSIM with Ward Linkage cluster method. Due to groundtruth data, we used 11 as the number of clusters.

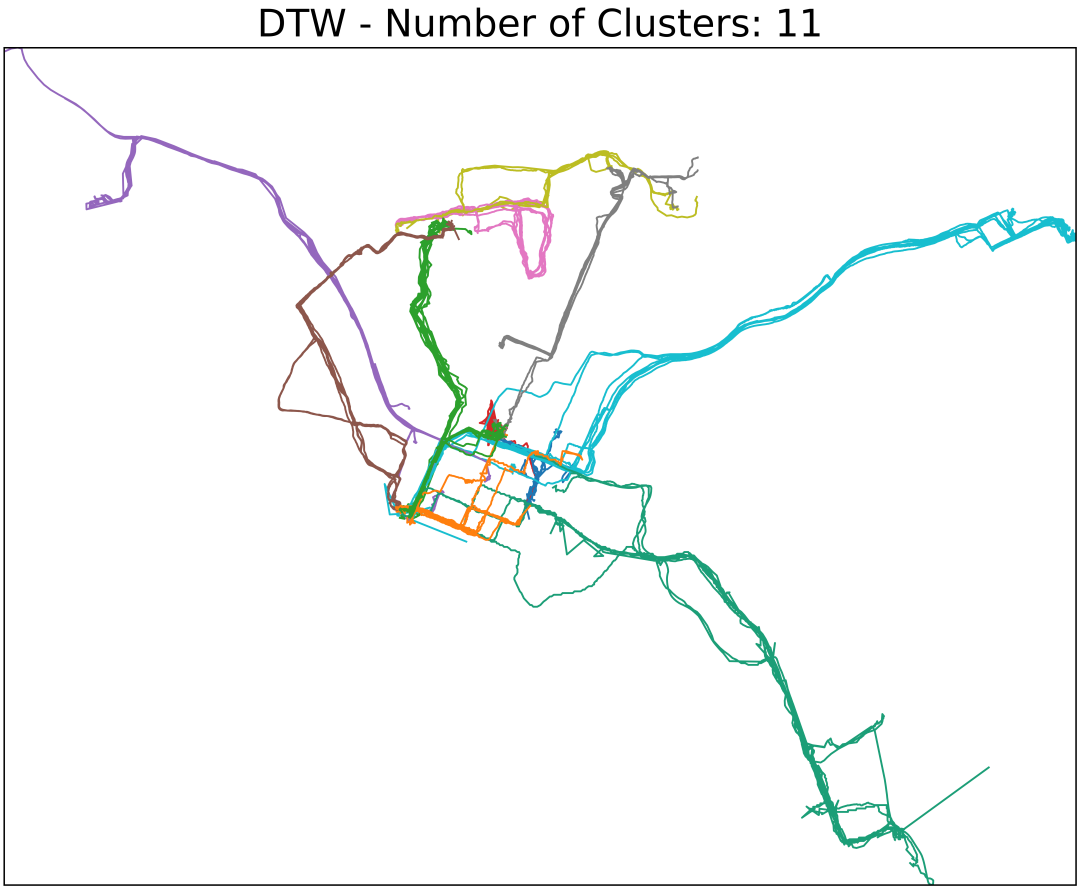


Figure 20: DTW Trajectories Clustering

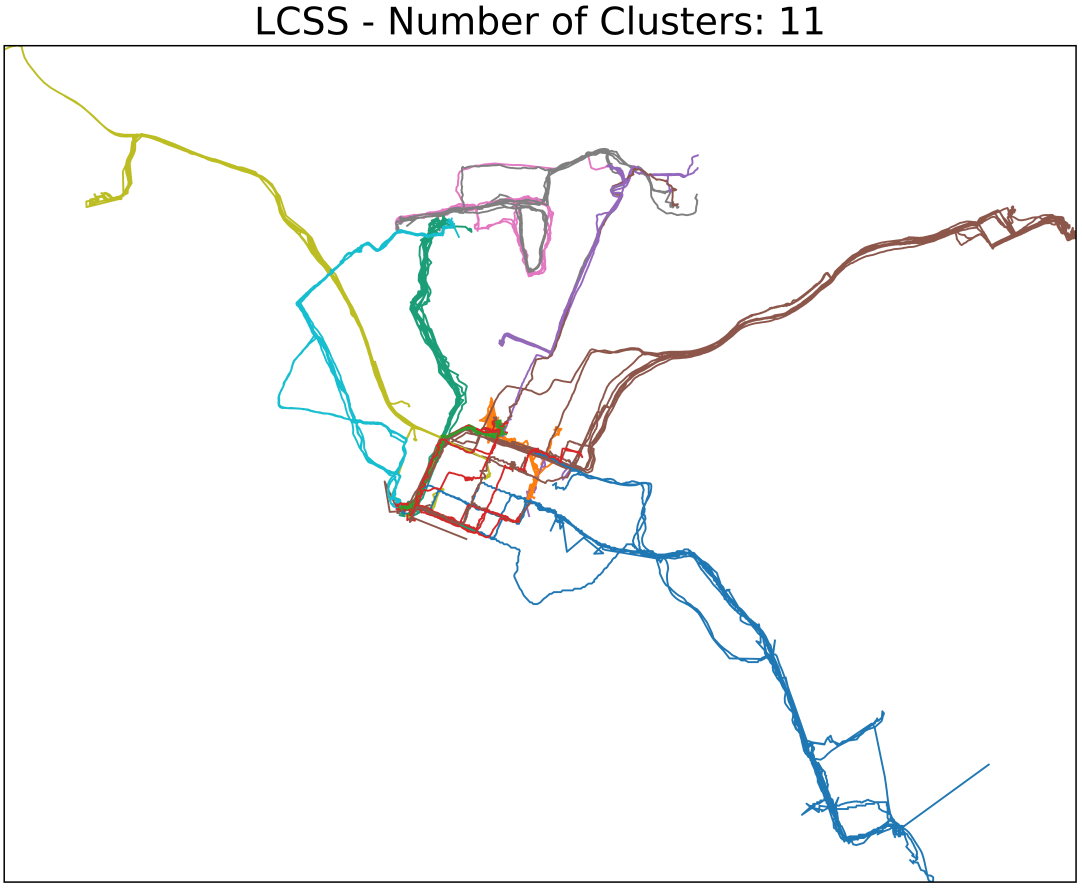


Figure 21: LCSS Trajectories Clustering

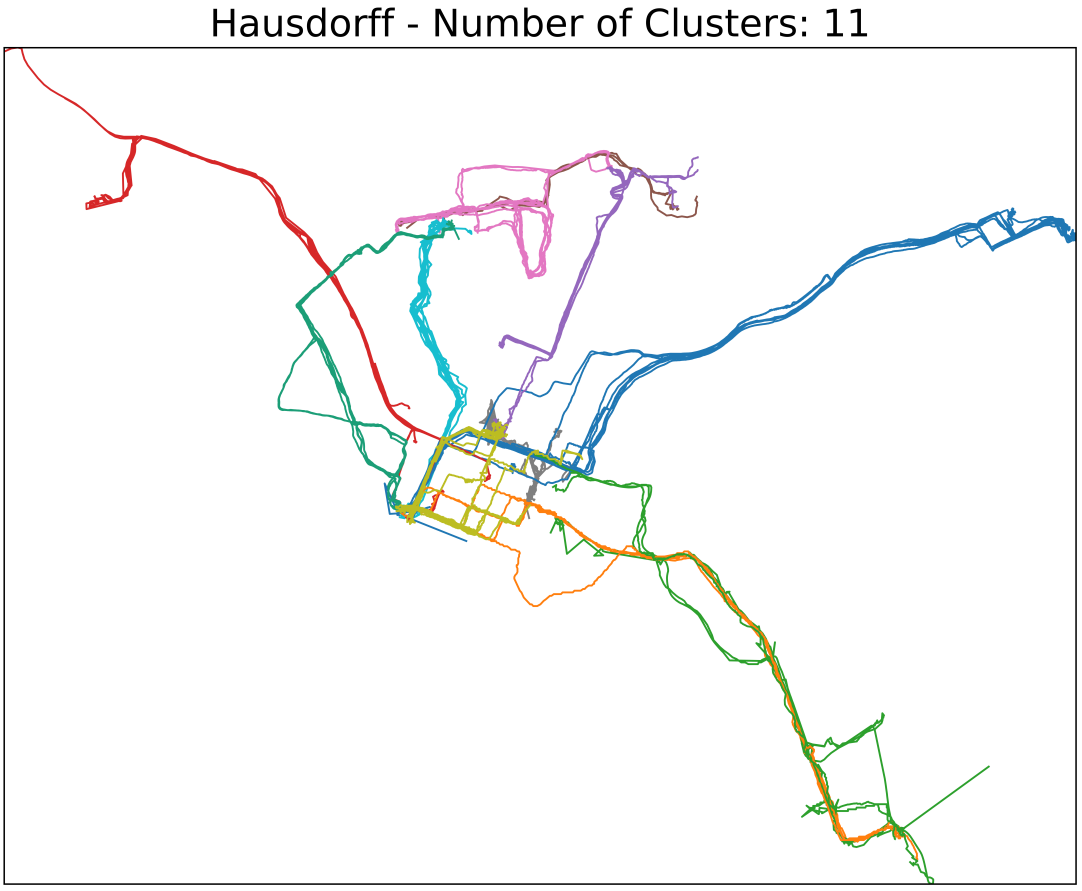


Figure 22: Hausdorff Trajectories Clustering

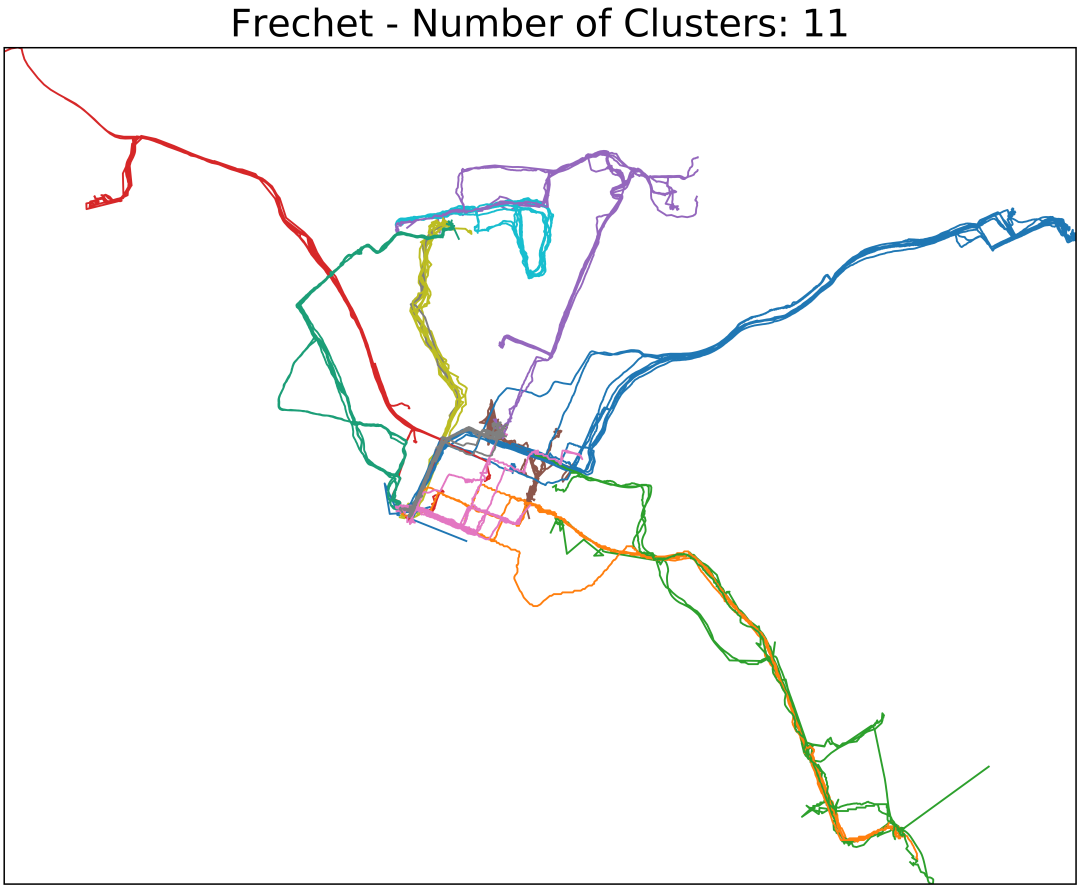


Figure 23: Fréchet Trajectories Clustering

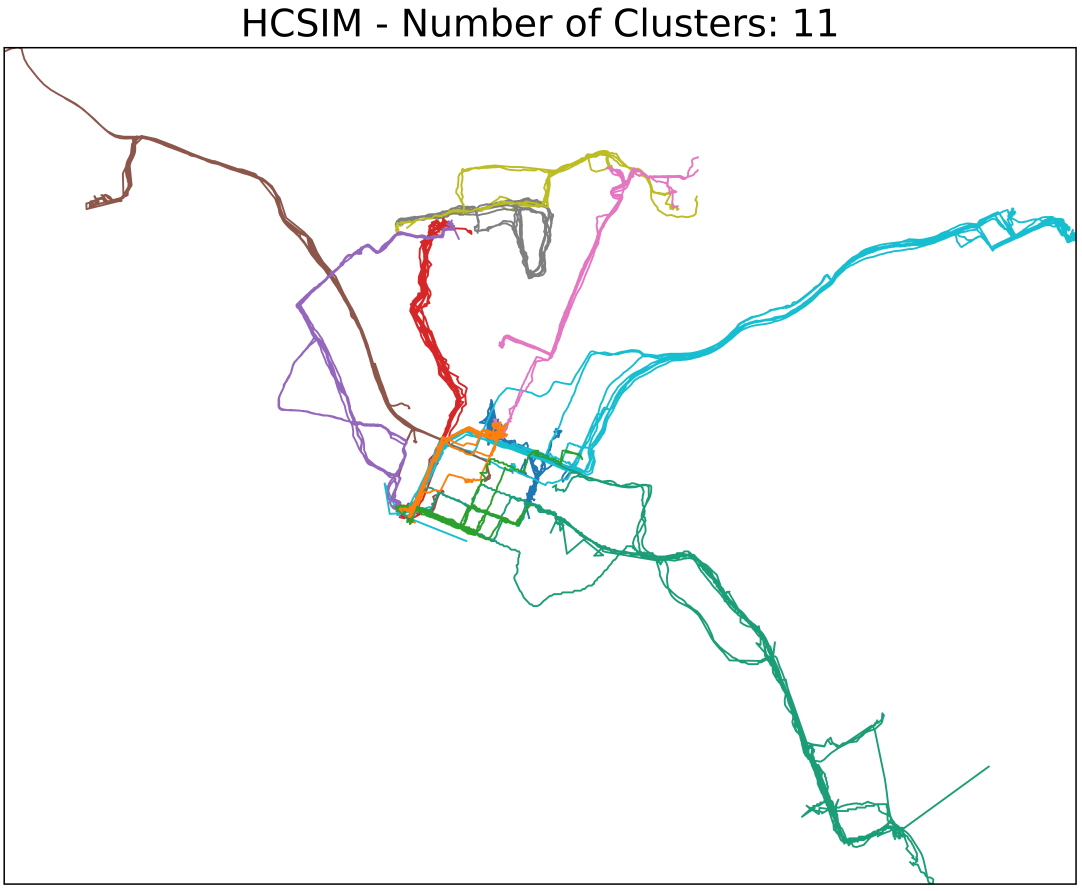


Figure 24: HCSIM Trajectories Clustering

### 5.1.4 Clusters Validation

We used *Silhouette Score* to decided the number of clusters.

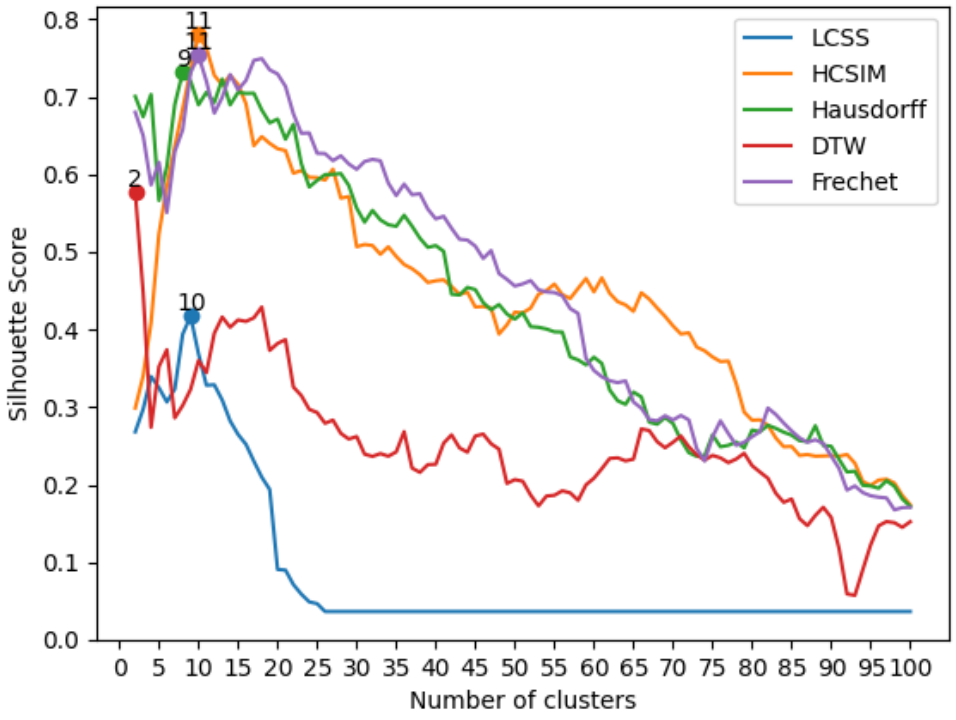


Figure 25: Silhouette Score depending on cluster

Figure 25 illustrated Silhouette Score based on number of clusters using 5 distances. 10 clusters are found with LCSS, 9 with Hausdorff and 11 with Frechet, HCSIM while DTW maximum at 2 clusters. HCSIM and Frechet give the best results as those equal with the groundtruth data.

Additionally, we used Pair Sets Index to evaluate cluster results from those 5 distances. Figure 26 showed that HCSIM (shaped base distance) give the better cluster results than warping-based distances with Pair Sets Index is the highest with red color indicate the index is over 0.8 while green indicate value is less than 0.8.

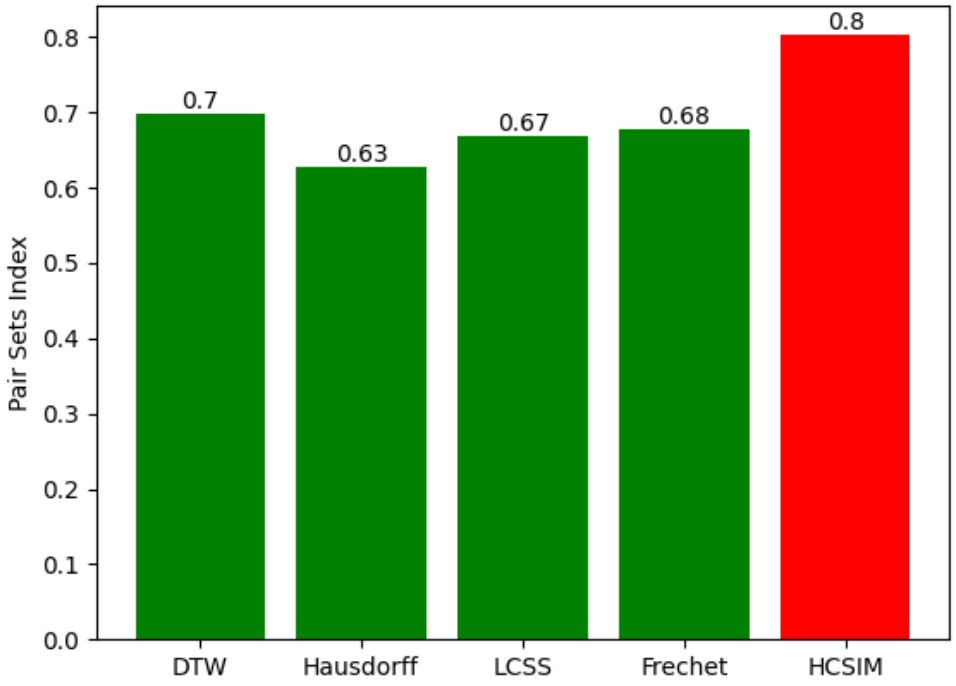


Figure 26: Pair Sets Index based on distance

Indeed, Figure 27 shows that HCSIM clusters nearly match ground truth data while other distances' clusters are not good enough.

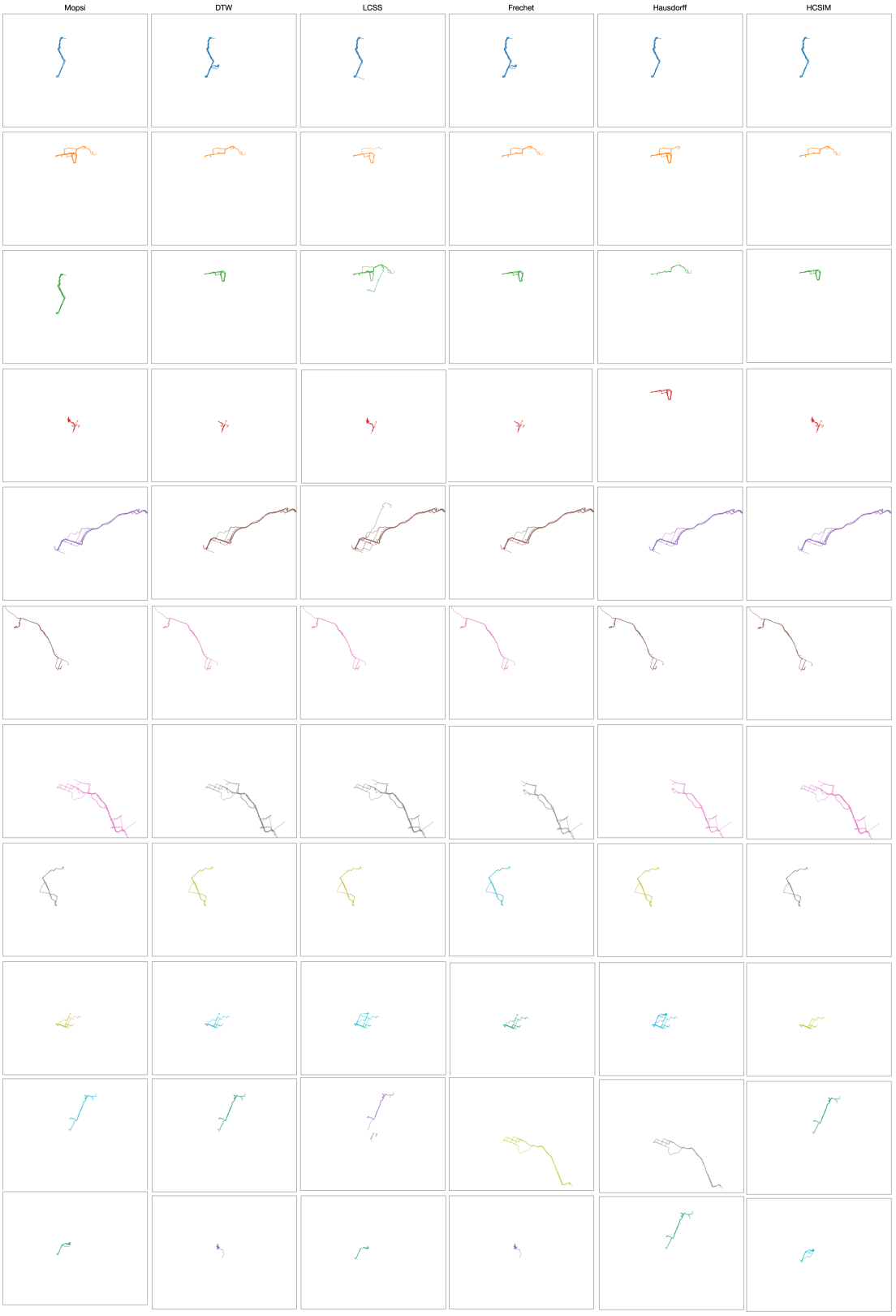


Figure 27: Compare Trajectories Clusters

## 5.2 Unknown Data

### 5.2.1 Data

The data we used are GPS data from 536 San-Francisco taxis over a 24-day period. These data are public and can be found on Piorkowski et al. (2009). We extracted data as shown in Figure 28. We preprocessed data extracting 2574 trajectories with the same start point as black a point in Figure 28 at Caltrain station and red points indicate the end points which are drop-off locations.


Figure 28: Caltrain Trajectories Dataset

### 5.2.2 Analysis of the distances

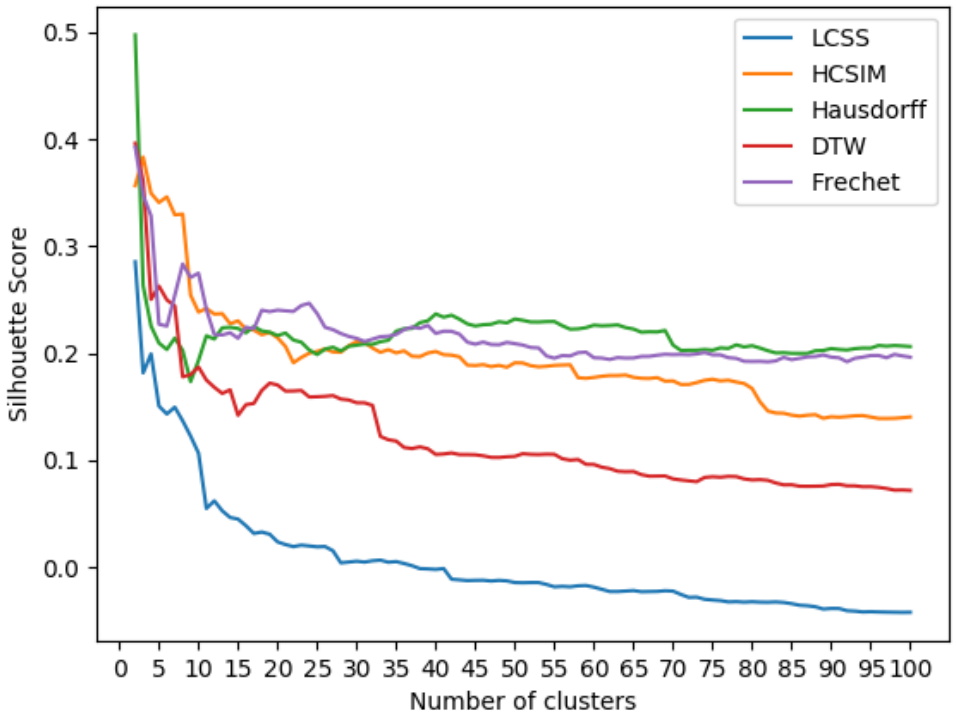


Figure 29: Silhouette Score

Figure 29 hardly determine optimal number of clusters as it ranges from 10 to 15 clusters. The shape-based distances (HC-SIM, Frechet, Hausdorff) are slightly better than warping-based distances (DTW, LCSS) as the Silhouette scores are higher. We chose 15 as the number of clusters for all distances then compare the results.

Figure 30 illustrates visual results for all distances with 15 clusters. Based on Mopsi experiment, HCSIM is expected to be the best. We observe that trajectories are well classified by HCSIM according to their paths in 35. The clusters using HCSIM seem to be consistent as the trajectories in each cluster grouped better than other distances. For example, the DTW cluster 5 in Figure 31, the LCSS cluster 8 in Figure 32, the Hausdorff cluster 3 in Figure 33, and the Frechet cluster 9 in Figure 34 got a lot of noise.

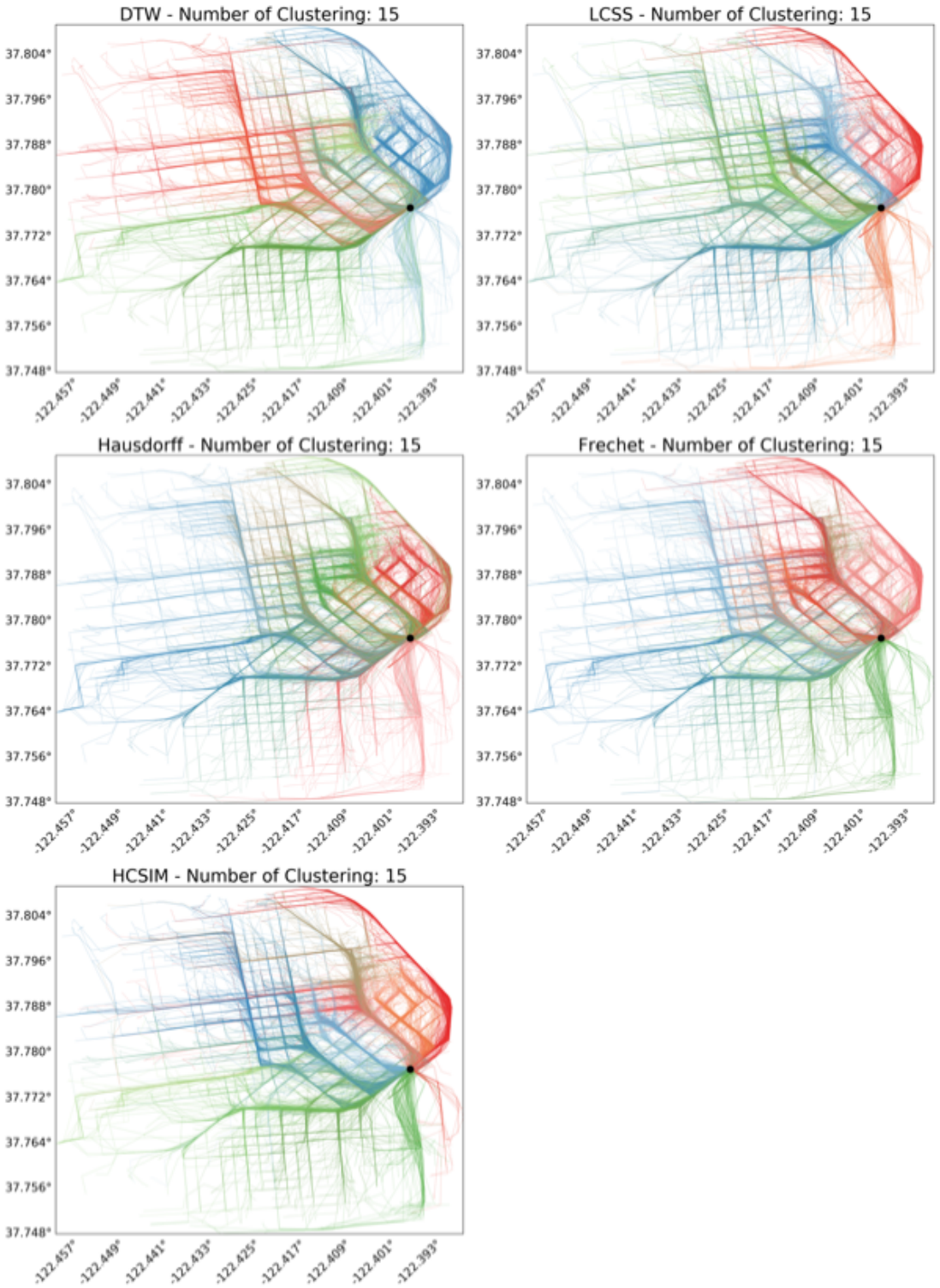


Figure 30: Caltrain Clusters

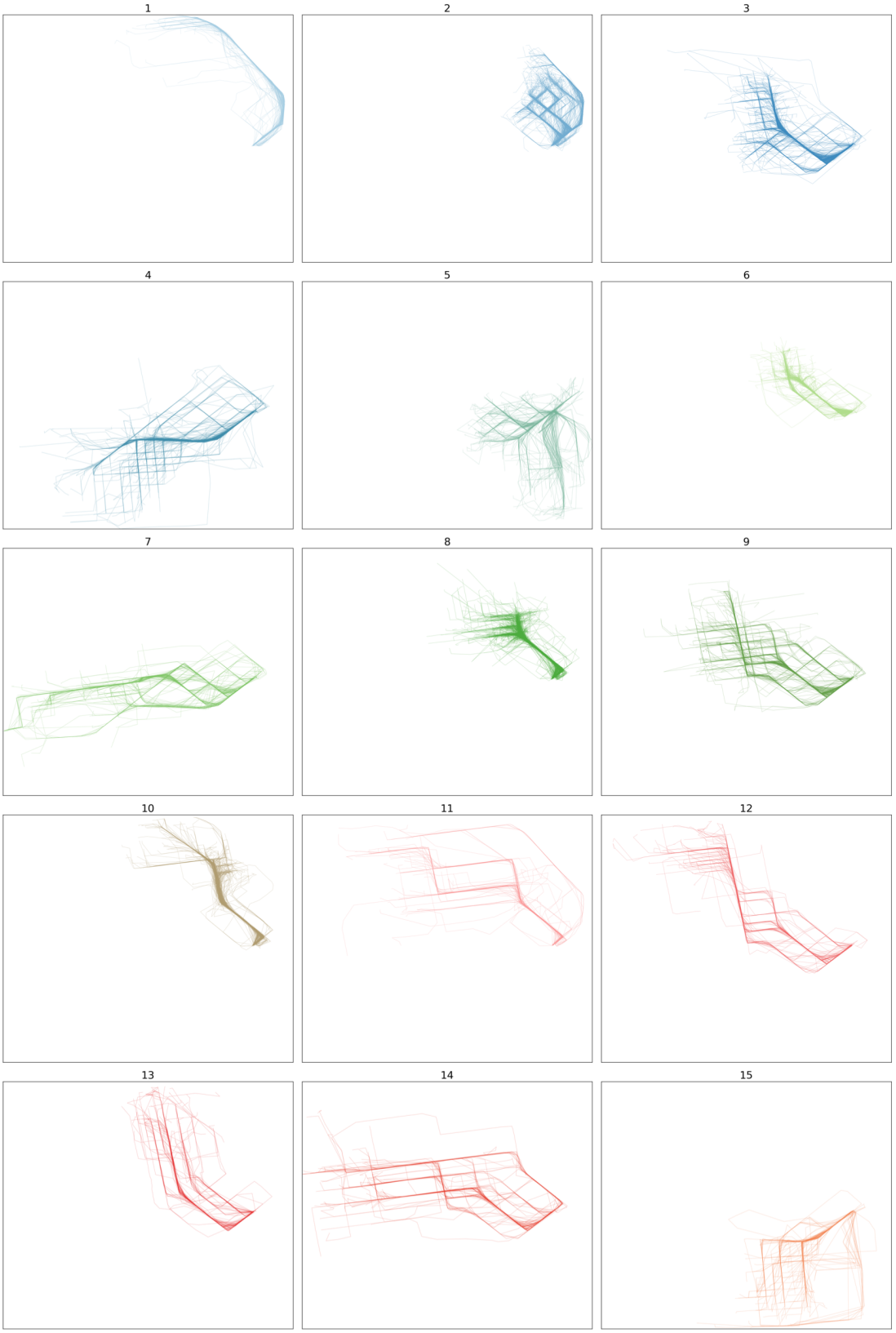


Figure 31: Caltrain DTW Clusters

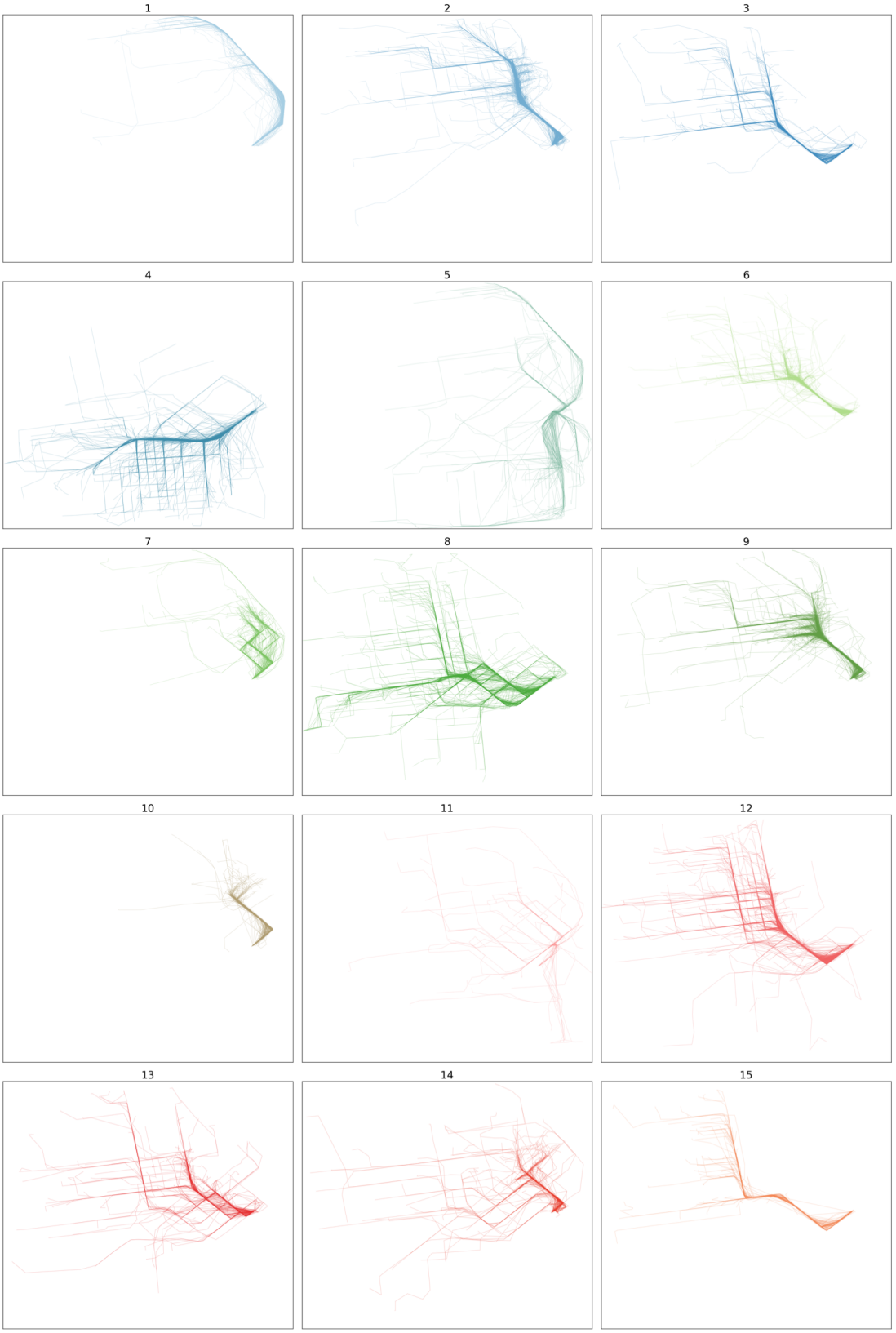


Figure 32: Caltrain LCSS Clusters

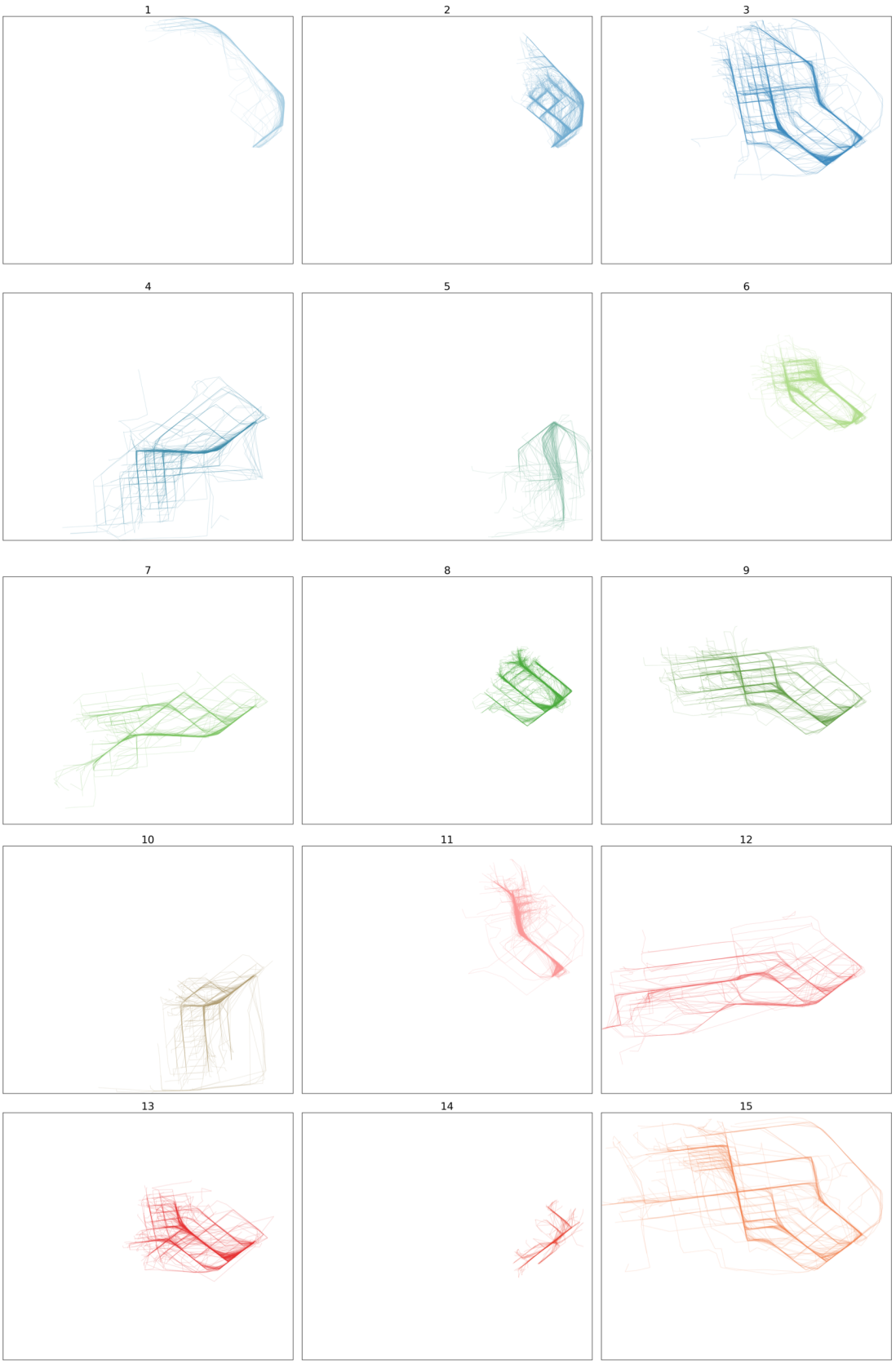


Figure 33: Caltrain Hausdorff Clusters

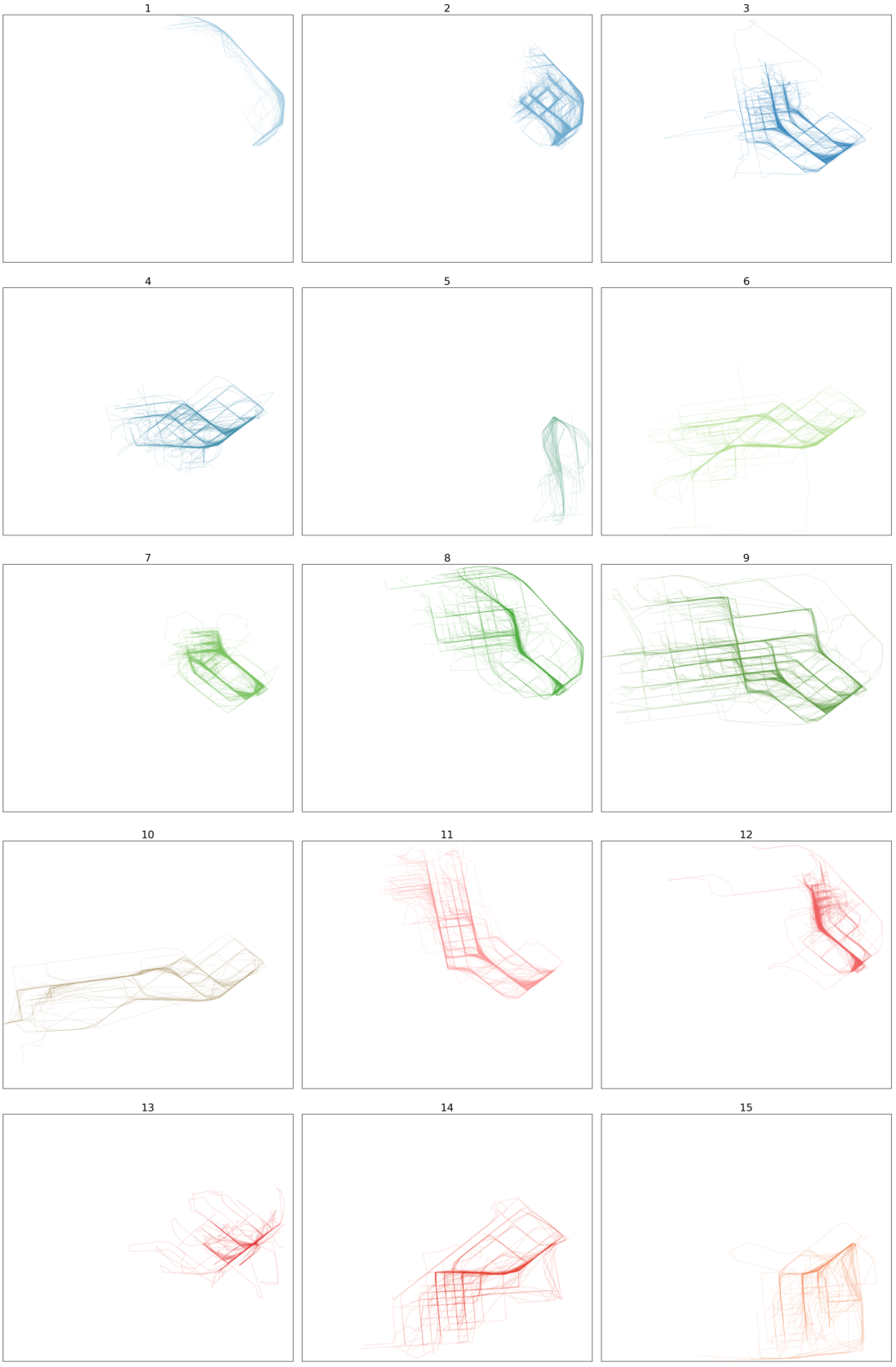


Figure 34: Caltrain Frechet Clusters

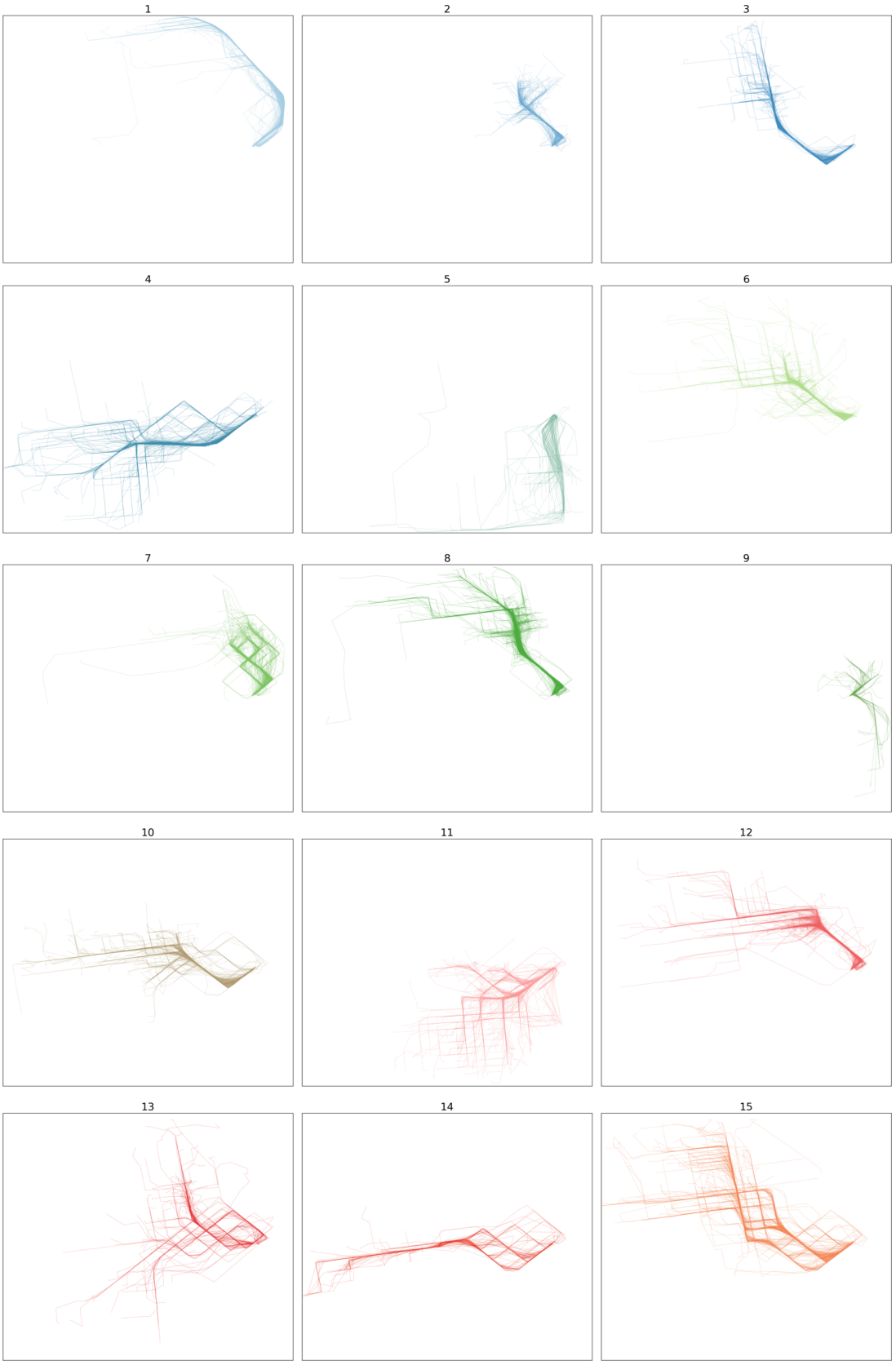


Figure 35: Caltrain HCSIM Clusters

# 6 Conclustions

Clustering trajectories is heavily dependent on the selection of an appropriate distance. In this thesis, we explored several distances concentrating on various characteristics of trajectories. All distances seems work well with hierarchical agglomerative clustering (HAC) method to cluster trajectories. Based on our experiment, shape-based distances give a better clusters than warping-base ones, especially HC-SIM enable us obtain good clusters that nearly match groundtruth data. Both clustering and validation methods that we use in this thesis work well and give a good results. This may be used to address a variety of problems. From learning moving objects' behavior, many applications are able to recommend locations to visit or arrange city's trip based on that result.

# References


Abbaspour, R. A., Shaeri, M., & Chehreghan, A. (2017). A method for similarity measurement in spatial trajectories. *Spatial Information Research*, *25*(3), 491–500.

Aghabozorgi, S., Shirkhorshidi, A. S., & Wah, T. Y. (2015). Time-series clustering–a decade review. *Information Systems*, *53*, 16–38.

Atev, S., Masoud, O., & Papanikolopoulos, N. (2006). Learning traffic patterns at intersections by spectral clustering of motion trajectories. In *2006 ieee/rsj international conference on intelligent robots and systems* (pp. 4851–4856).

Ball, G. H., & Hall, D. J. (1965). *Isodata, a novel method of data analysis and pattern classification* (Tech. Rep.). Stanford research inst Menlo Park CA.

Banerjee, A., Dhillon, I. S., Ghosh, J., Sra, S., & Ridgeway, G. (2005). Clustering on the unit hypersphere using von mises-fisher distributions. *Journal of Machine Learning Research*, *6*(9).

Berkhin, P. (2006). A survey of clustering data mining techniques. In *Grouping multidimensional data* (pp. 25–71). Springer.

Besse, P., Guillouet, B., Loubes, J.-M., & François, R. (2015). Review and perspective for distance based trajectory clustering. *arXiv preprint arXiv:1508.04904*.

Bian, J., Tian, D., Tang, Y., & Tao, D. (2019). Trajectory data classification: A review. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *10*(4), 1–34.

Caliński, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, *3*(1), 1–27.

Chen, J., Wang, R., Liu, L., & Song, J. (2011). Clustering of trajectories based on hausdorff distance. In *2011 international conference on electronics, communications and control (icecc)* (pp. 1940–1944).

Chen, M., Xu, M., & Franti, P. (2012). Compression of gps trajectories. In *2012 data compression conference* (pp. 62–71).

Chen, Z., Shen, H. T., Zhou, X., Zheng, Y., & Xie, X. (2010). Searching trajectories by locations: an efficiency study. In *Proceedings of the 2010 acm sigmod international conference on management of data* (pp. 255–266).

Cheng, Z., Jiang, L., Liu, D., & Zheng, Z. (2018). Density based spatio-temporal trajectory clustering algorithm. In *Igarss 2018-2018 ieee international geoscience and remote sensing symposium* (pp. 3358–3361).

Eiter, T., & Mannila, H. (1994). *Computing discrete fréchet distance.*

Ertöz, L., Steinbach, M., & Kumar, V. (2003). Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *Proceedings of the 2003 siam international conference on data mining* (pp. 47–58).

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd* (Vol. 96, pp. 226–231).

Faloutsos, C., Ranganathan, M., & Manolopoulos, Y. (1994). Fast subsequence matching in time-series databases. *Acm Sigmod Record*, *23*(2), 419–429.

Fowlkes, E. B., & Mallows, C. L. (1983). A method for comparing two hierarchical clusterings. *Journal of the American statistical association*, *78*(383), 553–569.

Fränti, P., & Mariescu-Istodor, R. (2019). Averaging gps segments challenge 2019. *unpublished work*.

Fränti, P., & Sieranoja, S. (2018). K-means properties on six clustering benchmark datasets. *Applied Intelligence*, *48*(12), 4743–4759.

Fränti, P., & Sieranoja, S. (2019). How much can k-means be improved by using better initialization and repeats? *Pattern Recognition*, *93*, 95–112.

Gaffney, S., & Smyth, P. (1999). Trajectory clustering with mixtures of regression models. In *Proceedings of the fifth acm sigkdd international conference on knowledge discovery and data mining* (pp. 63–72).

Gariel, M., Srivastava, A. N., & Feron, E. (2011). Trajectory clustering and an application to airspace monitoring. *IEEE Transactions on Intelligent Transportation Systems*, *12*(4), 1511–1524.

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Hartigan, J. A. (1975). *Clustering algorithms*. John Wiley & Sons, Inc.

Ignacio Gonzalez, S. L., et al. (2017). Hierarchical clustering tutorial [Computer software manual]. Retrieved from `http://genoweb.toulouse.inra.fr/~formation/CATIBIOS4BIOL_stats/Learning_clustering_current.pdf`

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, *31*(3), 264–323.

Kearney, J. K., & Hansen, S. (1990). *Stream editing for animation.*

Keogh, E. J., & Pazzani, M. J. (2000). Scaling up dynamic time warping for datamining applications. In *Proceedings of the sixth acm sigkdd international conference on knowledge discovery and data mining* (pp. 285–289).

Kriegel, H.-P., Kröger, P., Sander, J., & Zimek, A. (2011). Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *1*(3), 231–240.

Kruskal, J. B. (1983). An overview of sequence comparison: Time warps, string edits, and macromolecules. *SIAM review*, *25*(2), 201–237.

Krzanowski, W. J., & Lai, Y. (1988). A criterion for determining the number of groups in a data set using sum-of-squares clustering. *Biometrics*, 23–34.

Kuhn, H. W. (1955). The hungarian method for the assignment problem. *Naval research logistics quarterly*, *2*(1-2), 83–97.

Lee, J.-G., Han, J., & Whang, K.-Y. (2007). Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 acm sigmod international conference on management of data* (pp. 593–604).

Mariescu-Istodor, R. (2013). Detecting user actions in mopsi. *University of Eastern Finland School of Computing Thesis*.

Mariescu-Istodor, R., & Fränti, P. (2017). Grid-based method for gps route analysis for retrieval. *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, *3*(3), 1–28.

Meilă, M. (2003). Comparing clusterings by the variation of information. In *Learning theory and kernel machines* (pp. 173–187). Springer.

Myers, C., Rabiner, L., & Rosenberg, A. (1980). Performance tradeoffs in dynamic time warping algorithms for isolated word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *28*(6), 623–635.

Pfeifer, P. E., & Deutrch, S. J. (1980). A three-stage iterative procedure for space-time modeling phillip. *Technometrics*, *22*(1), 35–47.

Piorkowski, M., Sarafijanovic-Djukic, N., & Grossglauser, M. (2009). *Crawdad data set epfl/mobility (v. 2009-02-24).*

Priestley, M. (1980). State-dependent models: A general approach to non-linear time series analysis. *Journal of Time Series Analysis*, *1*(1), 47–71.

Qian, H., & Lu, Y. (2017). Simplifying gps trajectory data with enhanced spatial-temporal constraints. *ISPRS International Journal of Geo-Information*, *6*(11), 329.

Rezaei, M., & Fränti, P. (2016). Set matching measures for external cluster validity. *IEEE Transactions on Knowledge and Data Engineering*, *28*(8), 2173–2186.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, *20*, 53–65.

Su, H., Liu, S., Zheng, B., Zhou, X., & Zheng, K. (2020). A survey of trajectory distance measures and performance evaluation. *The VLDB Journal*, *29*(1), 3–32.

Sultana, S. I. (2020). How the hierarchical clustering algorithm works [Computer software manual]. Retrieved from `https://dataaspirant.com/hierarchical-clustering-algorithm/`

Van Rijsbergen, C. (1979). Information retrieval: theory and practice. In *Proceedings of the joint ibm/university of newcastle upon tyne seminar on data base systems* (pp. 1–14).

Vasquez, D., & Fraichard, T. (2004). Motion prediction for moving objects: a statistical approach. In *Ieee international conference on robotics and automation, 2004. proceedings. icra'04. 2004* (Vol. 4, pp. 3931–3936).

Waga, K., Tabarcea, A., Chen, M., & Fränti, P. (2012). Detecting movement type by route segmentation and classification. In *8th international conference on collaborative computing: networking, applications and worksharing (collaboratecom)* (pp. 508–513).

Wang, H., Su, H., Zheng, K., Sadiq, S., & Zhou, X. (2013). An effectiveness study on trajectory similarity measures. In *Proceedings of the twenty-fourth australasian database conference-volume 137* (pp. 13–22).

Xu, L. (1997). Bayesian ying–yang machine, clustering and number of clusters. *Pattern Recognition Letters*, *18*(11-13), 1167–1178.

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on neural networks*, *16*(3), 645–678.

Ying, J. J.-C., Lee, W.-C., Weng, T.-C., & Tseng, V. S. (2011). Semantic trajectory mining for location prediction. In *Proceedings of the 19th acm sigspatial international conference on advances in geographic information systems* (pp. 34–43).

Ying, J. J.-C., Lu, E. H.-C., Lee, W.-C., Weng, T.-C., & Tseng, V. S. (2010). Mining user similarity from semantic trajectories. In *Proceedings of the 2nd acm sigspatial international workshop on location based social networks* (pp. 19–26).

Yuan, G., Sun, P., Zhao, J., Li, D., & Wang, C. (2017). A review of moving object trajectory clustering algorithms. *Artificial Intelligence Review*, *47*(1), 123–144.

Zhao, L., & Shi, G. (2019). A trajectory clustering method based on douglas-peucker compression and density for marine traffic pattern recognition. *Ocean Engineering*, *172*, 456–467.

Zhao, Y., & Karypis, G. (2001). Criterion functions for document clustering: Experiments and analysis.