Contents

List of Tables

List of Figures

List of Equations

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 are in and then review common cluster techniques presented in the past few years. In this thesis, we start 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, WI-FI, and data storage hardware allow 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 the 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 (Mariescu-Istodor & Fränti, 2018), (Ahmed et al., 2015) and (Biagioni & Eriksson, 2012), detecting taxi fraud (Liu et al., 2013), data compression (M. Chen et al., 2012). Additionally, trajectory clustering is used to gather temporal-spatial information in the trajectory data and is widely 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 a 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). For instance, Figure 1 shows trajectories with different length. Multidimensional data is used in 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 for 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 several distance

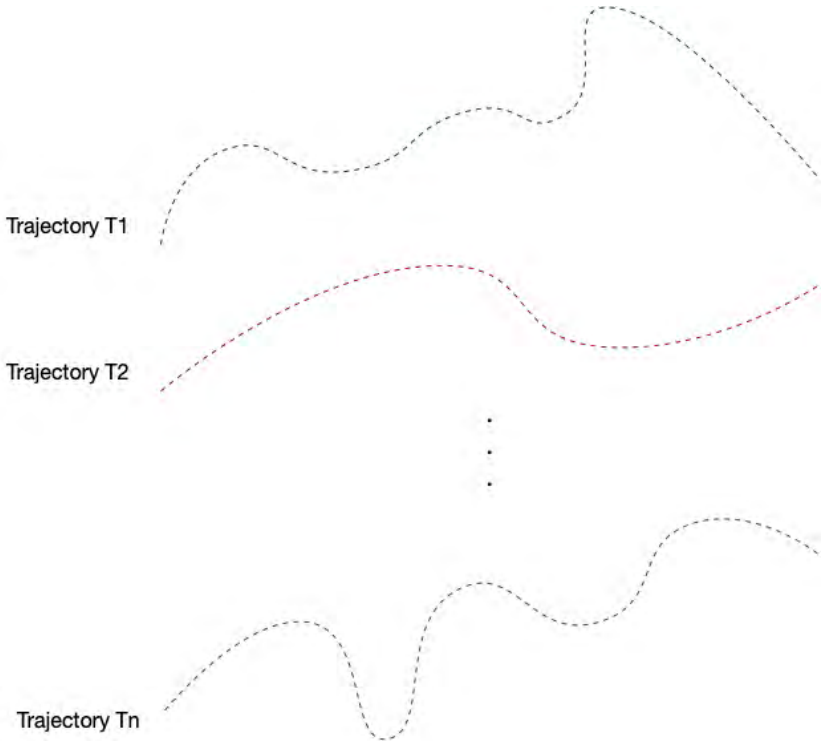

Figure 1: Trajectories with varying lengths

metrics used for trajectory data. E.g., the term "measure the sequence-only distance between trajectories" may include concepts such as Euclidean distance and Dynamic Time Wrapping Distance (DTW). Various approaches, including clustering which is illustrated in Figure 2 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).

There are two primary methodologies used for data clustering like trajectories. Specify trajectory clustering algorithms first, then use generic clustering methods to 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. To measure sim-

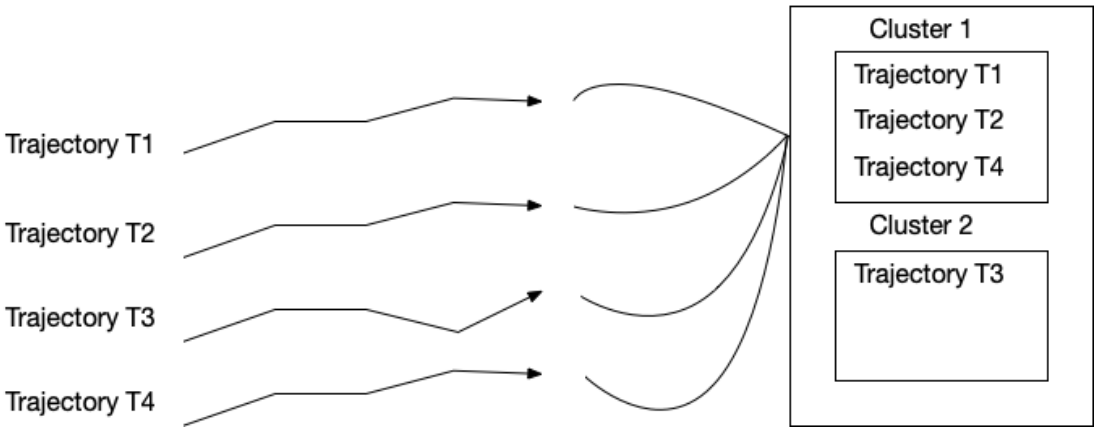

Figure 2: Clustering Methods

ilarity (inverse distance), it should be done before grouping. The definition of distance in spatial trajectories is significantly complicated. Trajectories are non-linear sequences of points with various lengths in multiple dimensions. Thus, 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, are temporal, spatio-temporal, and spatial. Euclidean, Fréchet, Hausdorff, DTW, and LCSS 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 a 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 trajectory complication, 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 data collection. Clustering techniques based on Euclidean distance provide erroneous results since 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 to obtain a perfect match. Another possibility is to concentrate on the shape of the trajectories. Hausdorff and Fréchet, for example, maybe adapted to trajectories. In the next section, we will study and compare several distances on the trajectory.

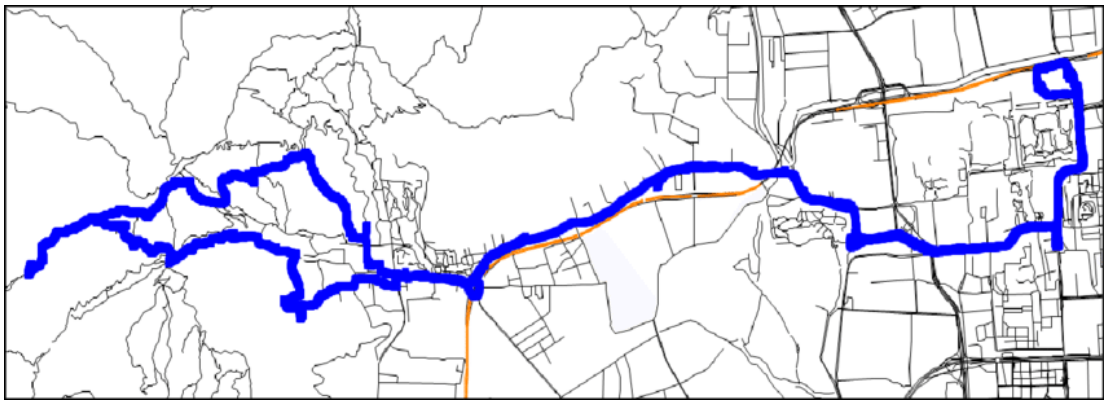

Figure 3: A GPS trajectory with 7076 points (Qian & Lu, 2017).

3 Trajectory Distance

A trajectory distance generalization that considered moving objects and 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).

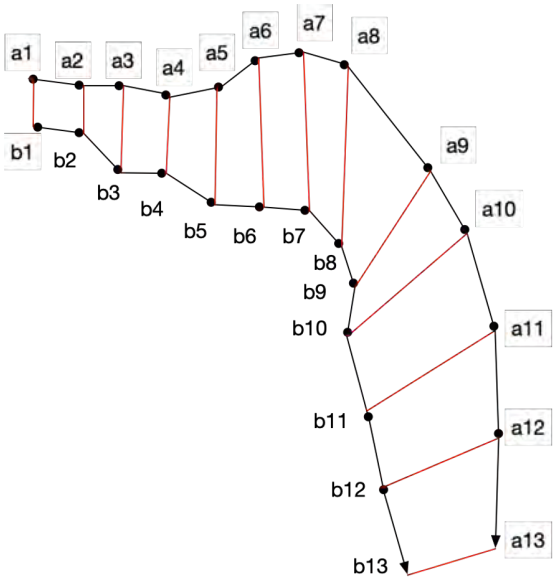

Figure 4: Euclidean distance

The Euclidean distance $d_{Euclidean}(A, B)$ as seen in Figure 4 between trajectories

$A\,and\,B$ with the same size n is defined in Equation 1:

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

Where a_i and b_i is 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 of 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 a similarity method for comparing two sequences. In the beginning, Myers et al. (1980) developed DTW to calculate time-series distance. In the 1980s, DTW was introduced to measure trajectory distance (Kruskal, 1983) and has become one of the most extensive methods for measuring trajectory distance. DTW explores all point alignments between two trajectories in search of the distance which is minimum (see Figure 5).

Specifically, Equation 2 defined the DTW distance $d_{DTW}(A, B)$ between two trajectories A and B:

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

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.

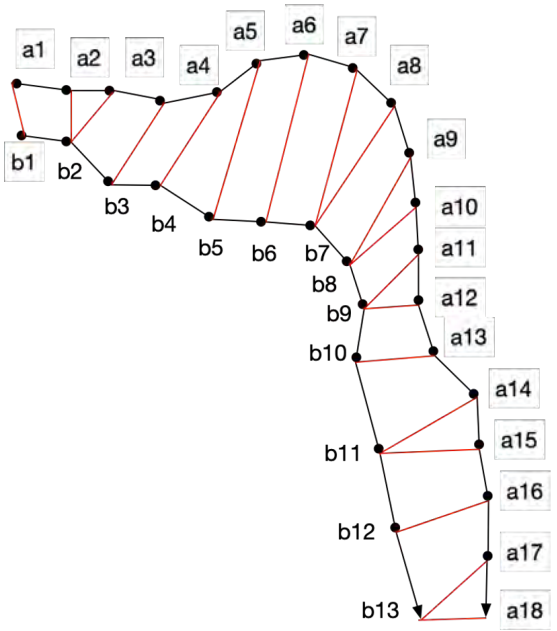

Figure 5: DTW distance

DTW's time complexity is $O(mn)$.

3.1.2 LCSS

Longest common subsequence (LCSS) is a well-known method for measuring string similarity that searches for 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 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 their distance is less than a threshold ϵ. Therefore, we will ignore some sample points of $A\,and\,B$ which are too far apart to contribute (see Figure 6).

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

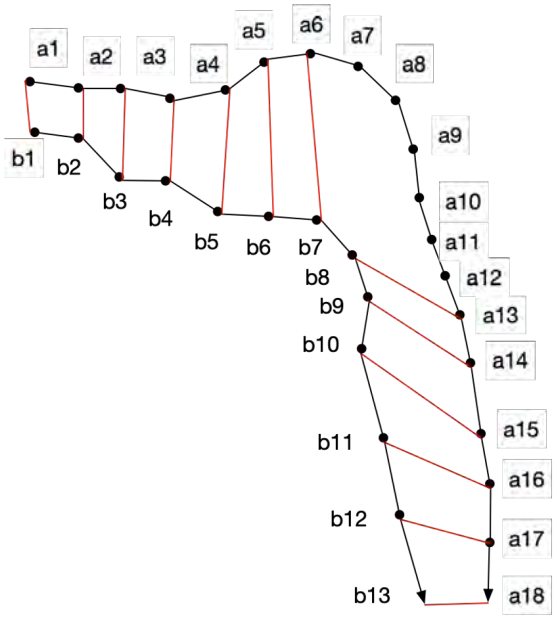

Figure 6: LCSS

defined as:

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

LCSS is similarity, so we will calculate the distance:

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

3.1.3 Pros and Cons

DTW and LCSS enable to comparison of 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 the 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 that measure similarity between curves considers both the location and sequence of points. The Fréchet distance between the two curves as shown in Figure 7 is the shortest length of leash necessary for both paths to be traversed.

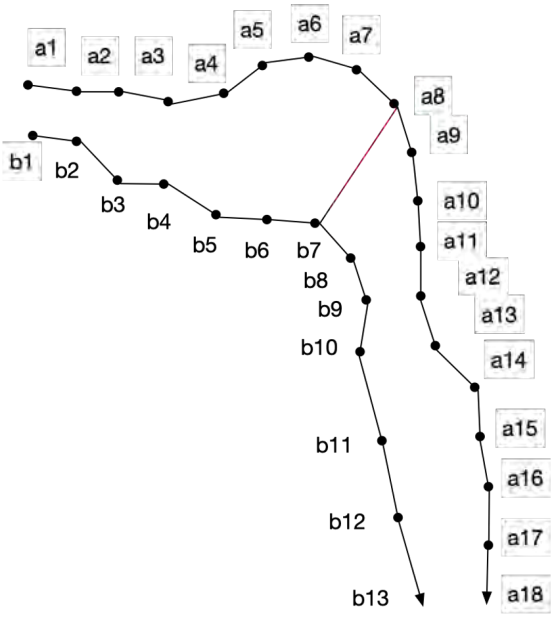

Figure 7: Fréchet distance

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 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 (see Figure 8).

The Hausdorff distance algorithm is shown as:

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

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 many 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.

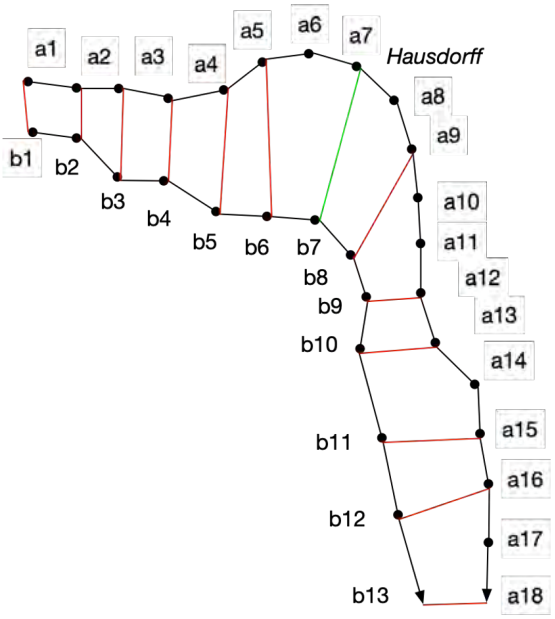

Figure 8: Hausdorff distance

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 (2021) proposed a new shape-based distance, HC-SIM (Hierarchical Cell Similarity), an independent variant of C-SIM measure as shown in Figure 10. 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.

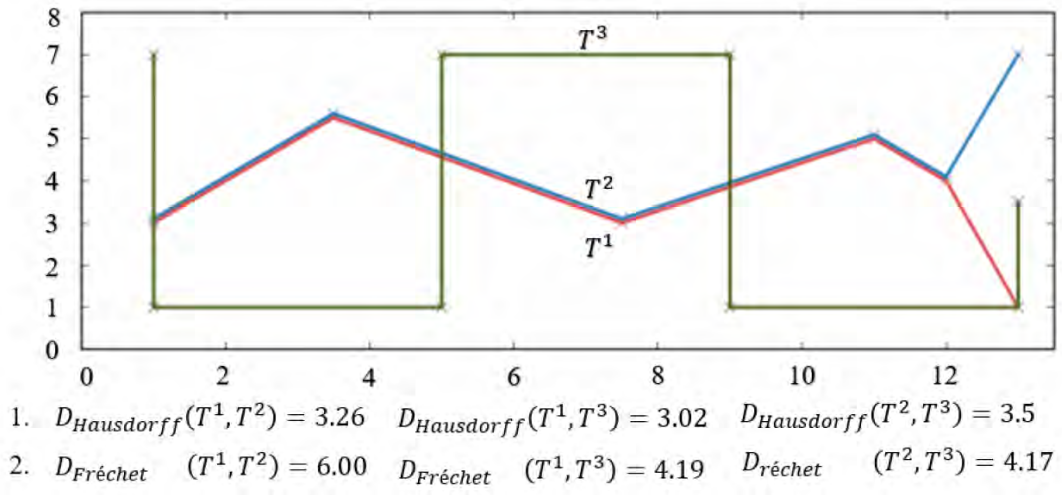

Figure 9: Distances calculated by Hausdorff and Fréchet (Besse et al., 2015)

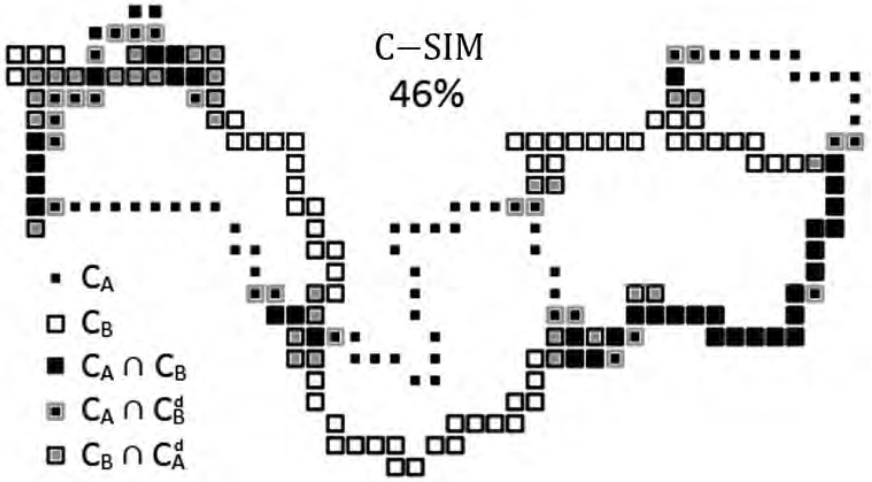

Figure 10: C-SIM

C-SIM algorithm was proposed by Mariescu-Istodor & Fränti (2017) to compute the 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 are shared by two segments to the total number of cells they occupy. In this thesis, we calculate HC-SIM distance through dissimilarity measurements:

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

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

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, numerous clustering algorithms have been evolved. In order to be successful with clustering, we first 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 into an algorithm. For instance, one may be interested to know whether the data that needs to be clustered is inputted into 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 are 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 the is predefined number of clusters. It classifies objects into a 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 matches the mean of data points.

The primary principle underlying k-means clustering is to define clusters that the 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 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 initialization 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 sensitivity to outliers and the ordering of the data. Additionally, (Fränti & Sieranoja, 2018) showed that when cluster size unbalances, k-means performs badly.

Balanced clustering is expected in many applications. Networking, for example, uses balanced clustering to minimize imbalanced energy consumption (Siavoshi et al., 2016). And in the traveling salesman problem, cities were clustered that each salesman in a cluster has an equal workload (Nallusamy et al., 2010). Furthermore, balanced clustering tends to avoid the emergence of outlier clusters (S. Zhong & Ghosh, 2003). Balanced K-means (Malinen & Fränti, 2014) is a general algorithm, which minimizes mean square error (MSE). The algorithm is the same as K-means but cluster sizes are equal. Instead of using linear programming in the assignment phase, Malinen & Fränti (2014) used the Hungarian algorithm (Burkard et al., 2012) to solve the problem.

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), as 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.

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 (see Figure 11). It takes an arbitrary point p as its starting point and generates its ϵ-neighborhood. If it is a core point, it will initiate a new cluster, which will be enlarged by including 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

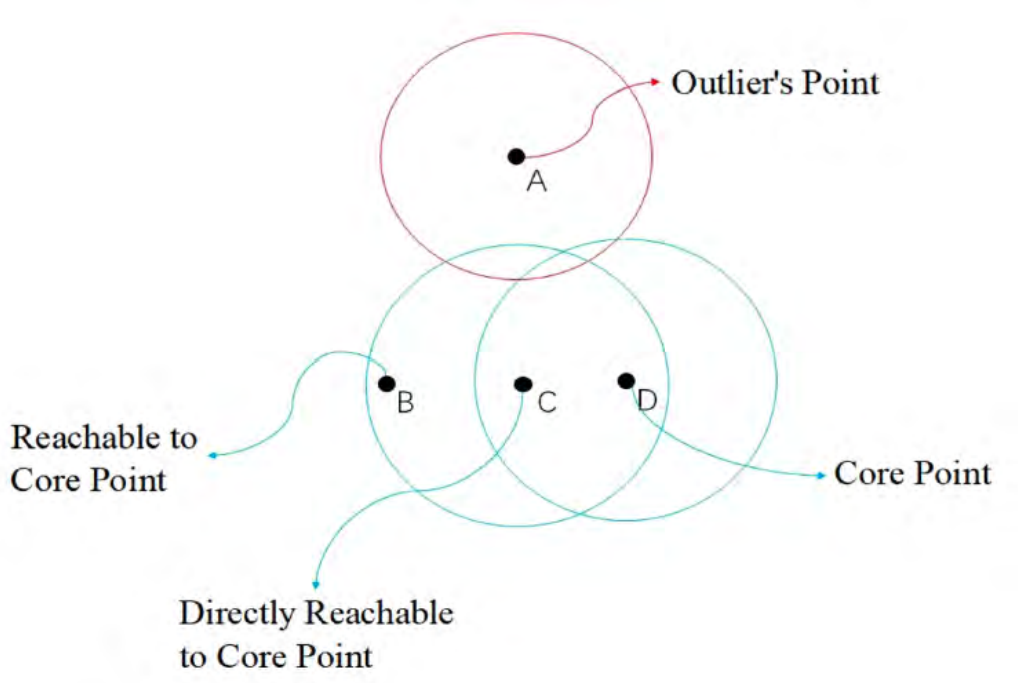

Figure 11: DBSCAN (Su et al., 2020)

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 example 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.

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. Valid metrics are required when implementing efficient algorithms like 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 varying lengths can be clustered. In this thesis, we will use HCA

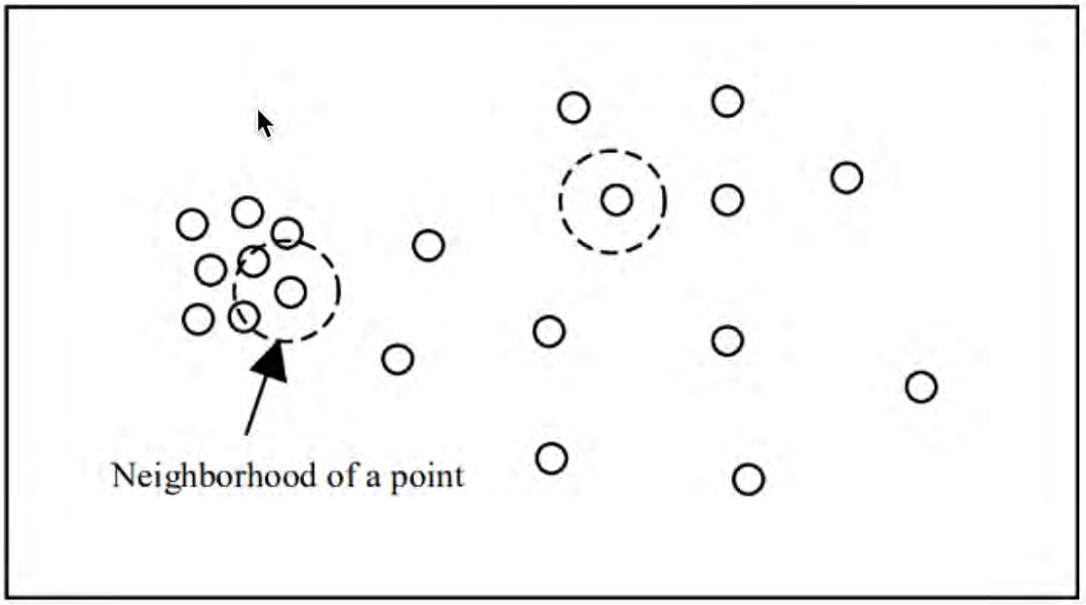

Figure 12: The distance threshold r. (Ertöz et al., 2003)

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, Figure 13 shows there are two basic strategies:

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

The agglomerative approach starts at the bottom of the tree. Each object is inserted into a leaf node representing a single cluster containing a single object. The algorithm proceeds by merging similar objects by constructing internal nodes

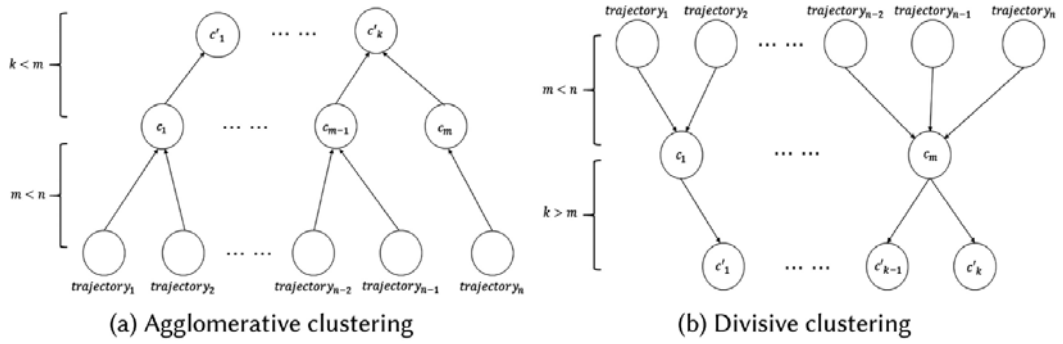

Figure 13: Hierarchical clustering models (Bian et al., 2019)

containing several nodes from the previous level. This is done recursively until there is a single node that defines the root of the tree.

The divisive approach works in the 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) which as an example of the hierarchical clustering algorithm. The main idea is to give some insight into 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 a 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 merged cost:

- Single Linkage.
- Complete Linkage.
- Average Linkage.

- Ward Linkage.

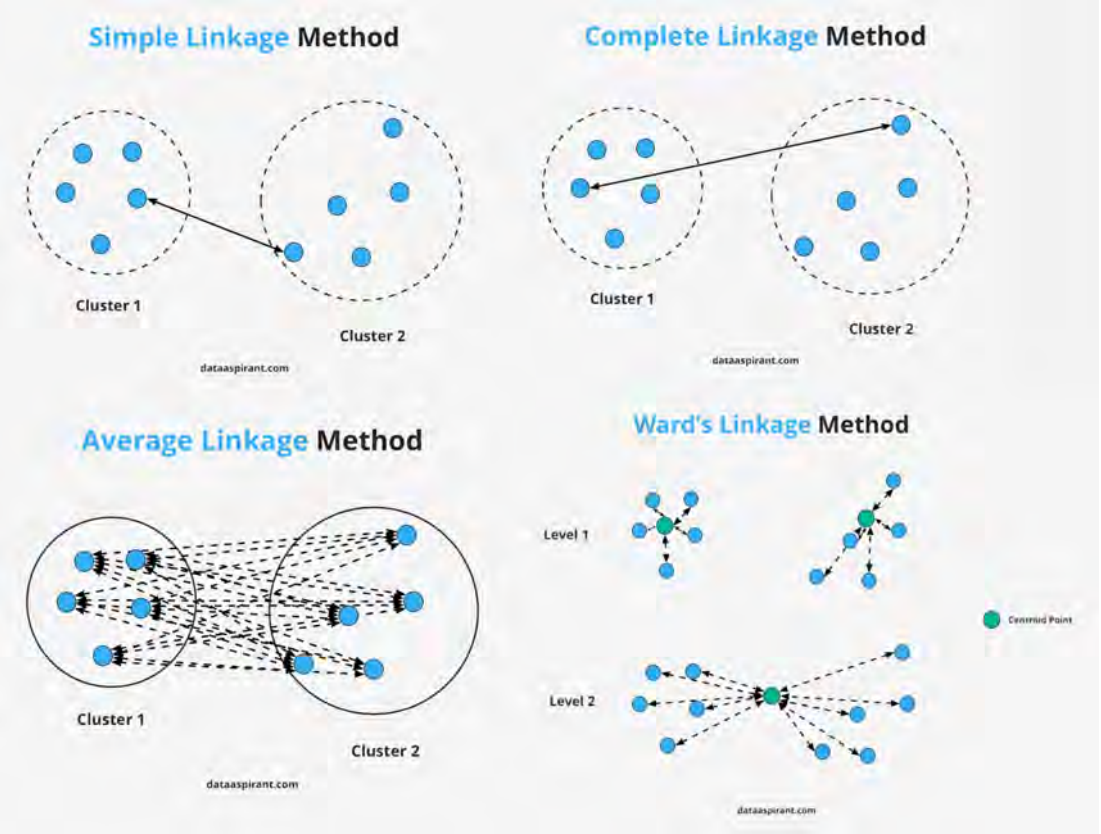

Figure 14: HAC clustering (Sultana, 2020)

The dissimilarity measurement is a derivation of the 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 between any two objects, $g \in G$ and $h \in H$:

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

The Average 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 contrast to the others, it evaluates cluster variance rather than calculating distance directly. Ward's method is most logical choice as it minimizes sum of square error (SSE) similarly as K-means. Fränti & Virmajoki (2006) had an experiment on Ward's method performance in minimizing SSE which is shown in Figure 16 (PNN is Ward's method).

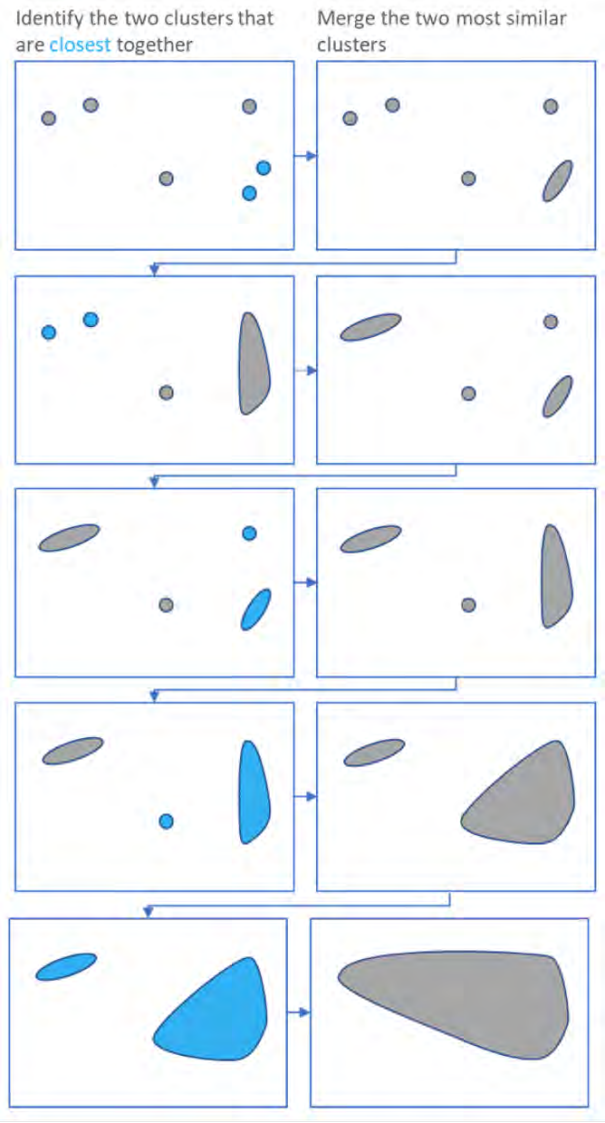

Figure 15: Ward Linkage (Ignacio Gonzalez et al., 2017)

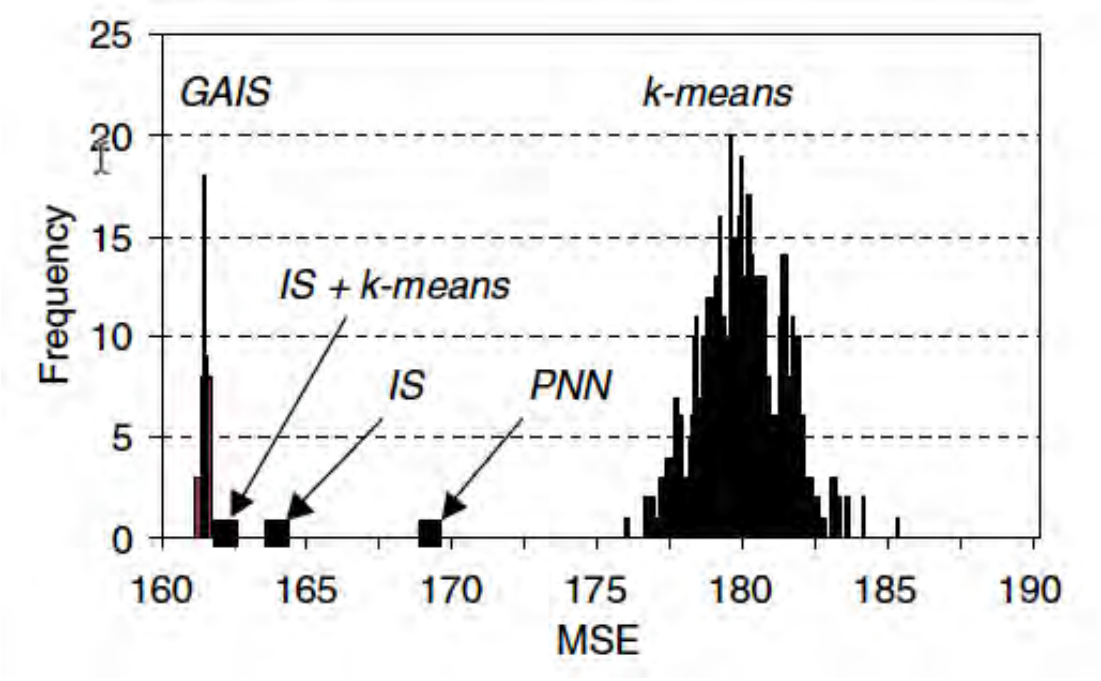

Figure 16: Histograms of the MSE-values of 50 runs (Fränti & Virmajoki, 2006)

4.1.4 Other Methods

In 2014, Malinen et al. (2014) proposed a different method for K-means clustering, putting the data into a specified clustering model that is best for identical pathological data of the same size and dimensions. Then apply an inverse transform from this pathological data back to the original data, while simultaneously improving the ideal clustering structure. K-means clustering is used to update the clustering model after each modification. If the changes are minor, the data vectors will eventually return to their original location without disrupting the clustering structure. The key items of this technique are determining an appropriate artificial data structure, performing the mapping, and controlling the inverse transformation illustrated in Figure 17. Alternatively, Fränti (2018) introduced using Random Swap algorithms which using prototype swaps to solve clustering, then using K-means to fine-tuning their positions. The basic idea is replacing the randomly picked centroid with the random data of the object and selecting centroids to replace by increasing the minimum cost function.

As trajectories vary in lengths, shapes, and densities, many clustering techniques are susceptible. C. Zhong et al. (2010) presented a graph-theoretical clustering

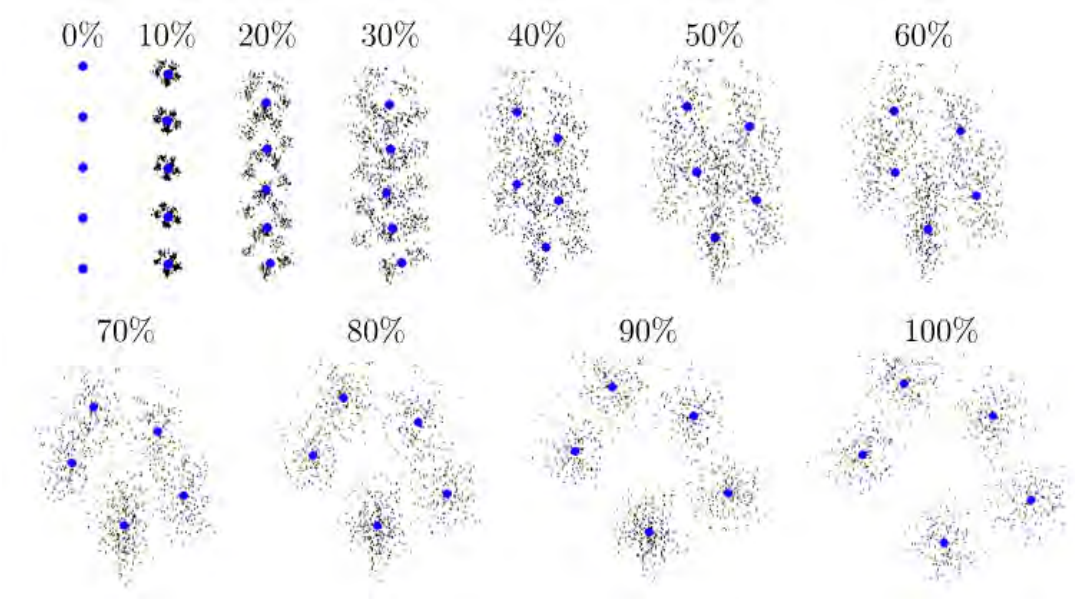

Figure 17: K-means* algorithm (Malinen et al., 2014)

method which is not impacted by those differences. In the beginning, two round minimal spanning trees are used to construct a graph and identify separate clusters that include clusters separated by distance and density divided by comparing the cuts in the two-round minimal spanning trees.

Density peaks algorithm (Rodriguez & Laio, 2014) is another option. The algorithm initially computes the densities of all points, followed by the distances between them and the nearest point with a greater density (delta). Points with high delta and density values are chosen as cluster centers while the others joined other clusters with the nearest greater density point. While popular, $O(N^2)$ time complexity has limited its application. Sieranoja & Fränti (2019) proposed a new algorithm called FastDP illustrated in Figure 18 eliminating the quadratic time complexity constraint of density peaks and enables for the clustering of big datasets.

Since GPS trajectories are spatial data, there are some clustering algorithms tailored for spatial data. Q. Zhao et al. (2015) utilizes a grid structure which example in Figure 19, takes into account a grid growth approach on the grid structure. This method combines the features of k-means with DBSCAN and does not require any inputs about the number of clusters. Rezaei & Franti (2018) provided another approach using grid-based for clustering. The approach was initially segmented by splitting each dimension into a predetermined number of

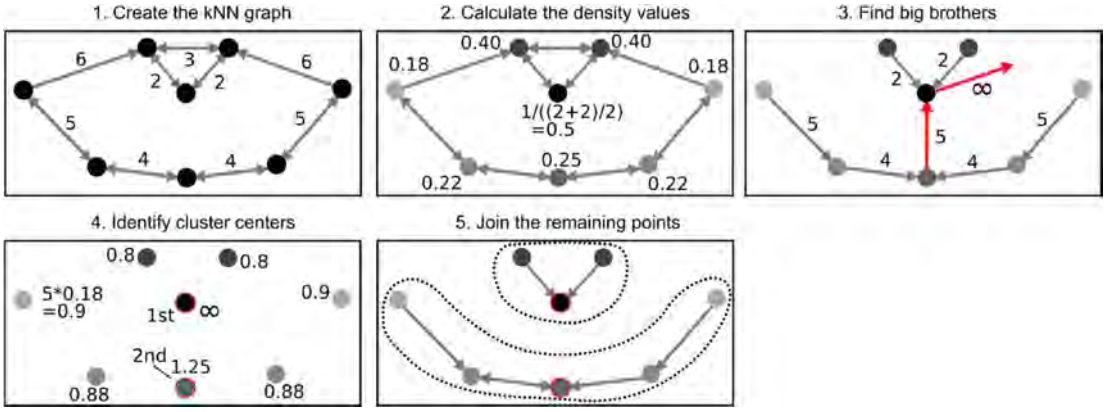

Figure 18: Fast density peaks algorithm (Sieranoja & Fränti, 2019)

bins. In the second phase, initial clusters are generated by simply indexing each item and allocating it to a cell without computing distance. Each cell represents a single cluster. In the final phase, nearby cells are merged to form final clusters based on some criteria such as density or connectivity.

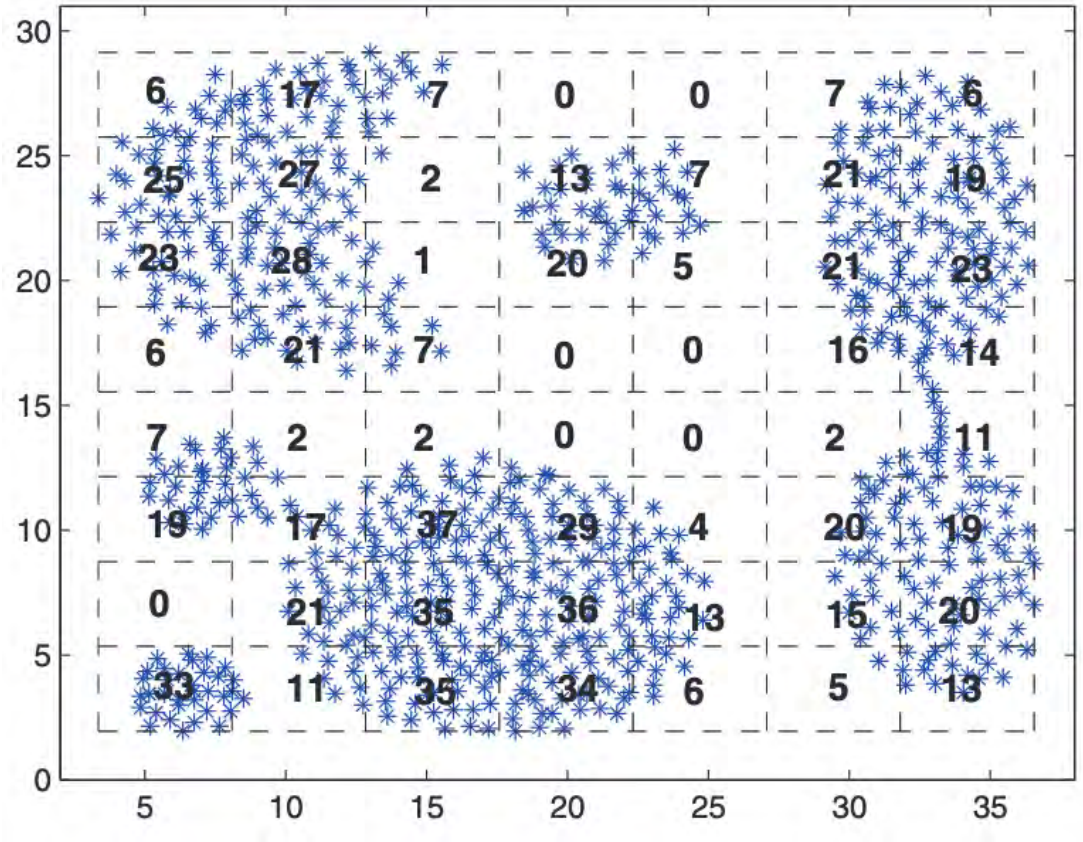

Figure 19: Grid Structure example (Q. Zhao et al., 2015)

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, the 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 ground-truth.

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.

Table 1: Internal Validation Indexes (Q. Zhao & Fränti, 2014)

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 (Dunn, 1974)	$d\left(c_i, c_j\right) = \min_{x \in c_i, x' \in c_j} \|x - x'\|^2$ $\text{diam}\left(c_k\right) = \max_{x, x' \in c_k} \|x - x'\|^2$

	Dunn $= \frac{\min_{i=1}^{M} \min_{j=i+1}^{M} d(c_i,c_j)}{\max_{k=1}^{M} \mathrm{diam}(c_k)}$
Davies & Bouldin (Davies & Bouldin, 1979)	$R_{ij} = \frac{S_i+S_j}{d_{ij}}, i \neq j$ 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 (Zoubi & Rawi, 2008)	$a(x_i) = \frac{1}{n_m-1} \sum_{j=1,j\neq i}^{n_m} \|x_i - x_j\|^2_{x_i,x_j \in C_m}$ $b(x_i) = \min_t \left\{ \frac{1}{n_t} \sum_{j \in C_t} \|x_i - x_j\|^2 \right\}_{x_i \notin C_t}$ $s(x_i) = \frac{b(x_i)-a(x_i)}{\max(a(x_i),b(x_i))}$ $SC = \frac{1}{N} \sum_{i=1}^{N} s(x_i)$
Bayesian Information Criterion (BIC) (Q. Zhao et al., 2008)	$BIC = L * N - \frac{1}{2} M(D+1) \sum_{i=1}^{M} \log(n_i)$
Xie-Beni (Xie & Beni, 1991)	$XB = \frac{\sum_{i=1}^{N} \sum_{k=1}^{M} u_{ik}^2 \|x_i - C_k\|^2}{N \min_{t \neq s} \{\|C_t - C_s\|^2\}}$

External validity indexes evaluate how closely the clustering matches 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, van Dongen criteria, and Pair Sets Index (PSI) (Rezaei & Fränti, 2016) are set matching indices. Cluster-level indexes such as Centroid Index (CI) and Centroid Similarity Index (CSI) (Fränti et al., 2014) take advantage of close connection between centroids and partitions, use cluster representatives rather than point-level partitions.

Some indices can be used to quickly identify the number of clusters by finding the minimum or maximum value, while others cannot. Q. Zhao et al. (2008) proposed

Table 2: External Validation Indexes (Rezaei & Fränti, 2016)

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 (Hubert & Arabie, 1985)	$RI = \frac{\left[\binom{n}{2} - \sum_i \binom{2^{n_i}}{2} - \sum_j \binom{n^j}{2} + 2 \sum_{ij} \binom{n_{ij}}{2} \right]}{\binom{n}{2}}$
Adjusted rand index (Hubert & Arabie, 1985)	$ARI = \frac{RI - E(RI)}{1 - E(RI)}$
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	$MS = \frac{\sqrt{\sum_i \binom{2^{n_i}}{2} + \sum_j \binom{n^{\cdot j}}{2} - 2 \sum_{ij} \binom{n_{ij}}{2}}}{\sqrt{\sum_j \binom{n^{\cdot j}}{2}}}$
Goodman-Kruskal	$GK = \sum_i p_i \left(1 - \max_j \frac{p_{ij}}{p_i} \right)$
Centroid Index (CI) (Fränti et al., 2014)	$CI_1(P,G) = \sum_{i=1}^{K'} \text{orphan}\left(G_i\right)$ $CI_2(P,G) = \max\left(CI_1(P,G), CI_1(G,P)\right)$
Centroid Similarity Index (CSI) (Fränti et al., 2014)	$CSI = \frac{\sum_{i-1}^{K} n_{ij} + \sum_{j-1}^{K'} n_{ji}}{2N}$ i, j : indexes of matched clusters
Pair Sets Index (PSI) (Rezaei & Fränti, 2016)	$\begin{cases} \frac{S - E(S)}{\max(K,K') - E(S)} & S \geq E(S), \max\left(K, K'\right) > 1 \\ 0 & S < E(S) \\ 1 & K = K' = 1 \end{cases}$ $S = \sum_{i=1}^{\min(K,K')} \frac{n_{ij}}{\max(n_i, m_j)}$ i, j : indexes of paired clusters

an approach called knee point detection which was illustrated in Figure 20 for BIC partitioning. The basic idea of this technique is to run an algorithm for several values of a number of clusters, compare using some internal indices, then we will detect the change in the curve which is called knee or jump point. However, because of some indices, SSW and log-likelihood are monotonous so there is no obvious knee point. Q. Zhao & Fränti (2014) used WB-Index, a sum-of-squares-based index (Q. Zhao et al., 2009) to find the optimal number of clusters. When the proper number of clusters is acquired, this index will reach its minimum value.

WB-Index:

$$WB - Index = M \frac{SSW(M)}{SSB(M)} \tag{14}$$

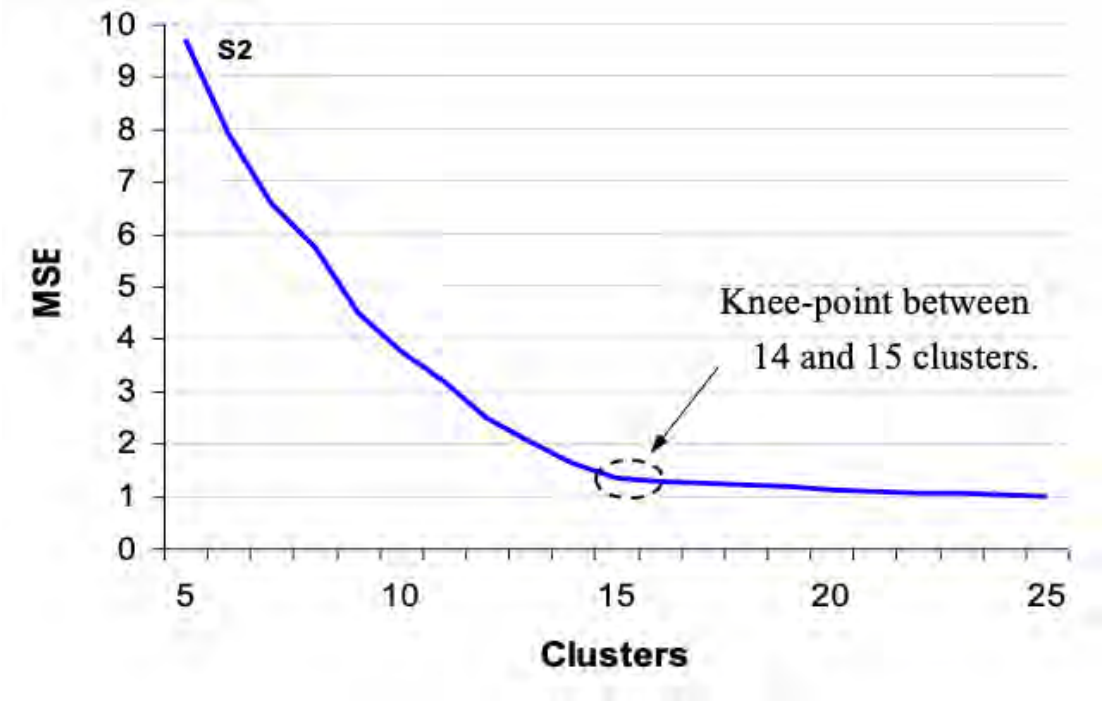

Figure 20: Knee Point Detection

- SSW: Sum of squares within the clusters
- SSB: Sum of squares between the clusters

In this thesis, we will use Silhouette Coefficient as an 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:

- **Cohesion(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*.

$$s = \frac{b - a}{max(a, b)} \tag{15}$$

Silhouette coefficient varies between -1 and 1, as a high value indicates a 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 the Hungarian algorithm (Kuhn, 1955) as illustrated in Figure 21, a set matching measure based on Braun-Banquet (BB), and a chance correction.

$$PSI = \begin{cases} \frac{S - E(S)}{max(K, K') - E(S)}, if S \geq E(S), max(K, K') > 1 \\ 0, if S < E(S) \\ 1, if K = K' = 1 \end{cases} \tag{16}$$

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 the same number of clusters.

5 Experiments

In this section, we analyze and evaluate 5 distances DTW, LCSS, Hausdorff, Fréchet and HC-SIM. Those distances were evaluated in two datasets: one is known data, one is unknown data then we used Silhouette to detect the number of clusters. For the ground-truth 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 them. 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.

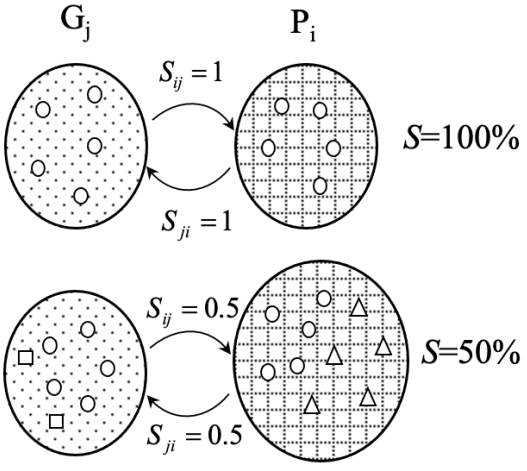

Figure 21: Pairing by Hungarian

Mopsi data is classified into two types: geotagged 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. The data set we use in this thesis were collected from Mopsi 2014 dataset (Mariescu-Istodor & Fränti, 2017) including 6779 trajectories were recorded by 51 users. We selected 10 trajectories groups which are in and around Joensuu, Finland, with the first group contains 20 trajectories and 10 for other groups. Figure 22 shows extracted data from Mopsi and Table 3 summarizes the data. Green points are the start points and red points are the end points when users stop their routes. Figure 23 illustrates 10 trajectories groups defined by Mopsi users. Table 4 shows the trajectory data structure.

Table 3: Mopsi Data Summary

Trajectories	**Points**	**Kilometers**
110	53976	336.419

Table 4: 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

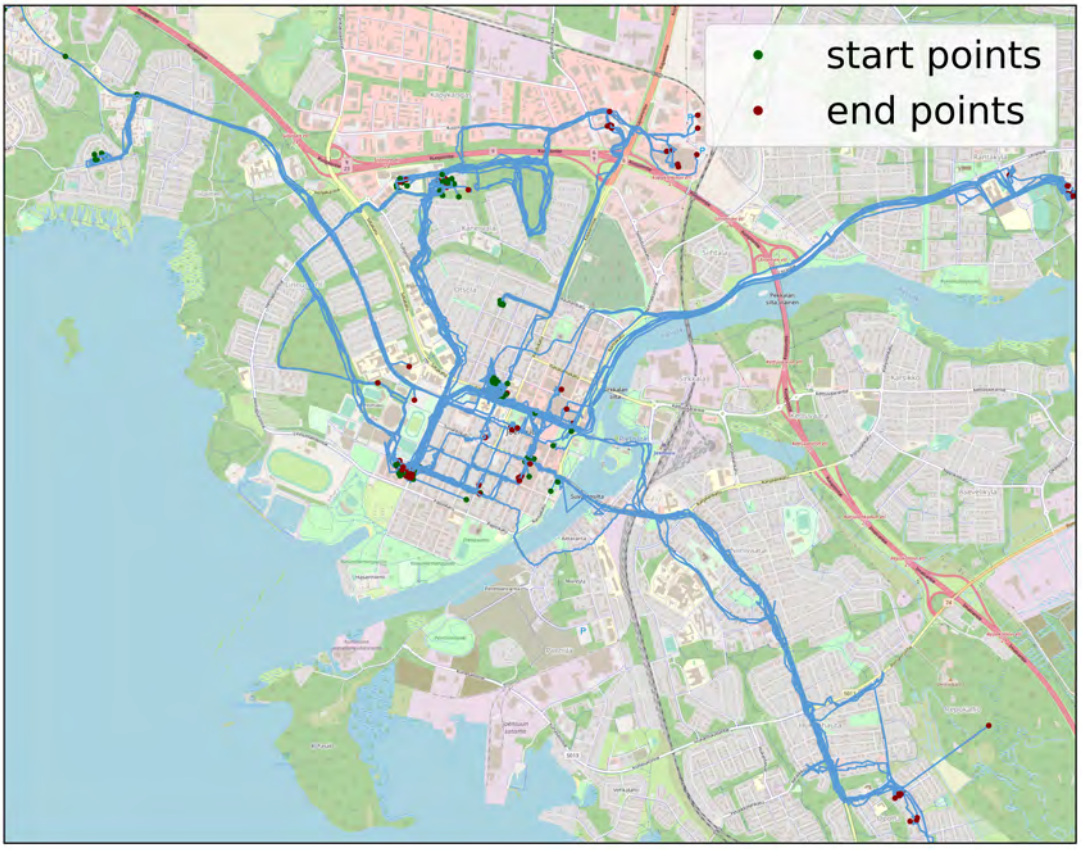

Figure 22: Mopsi Trajectories

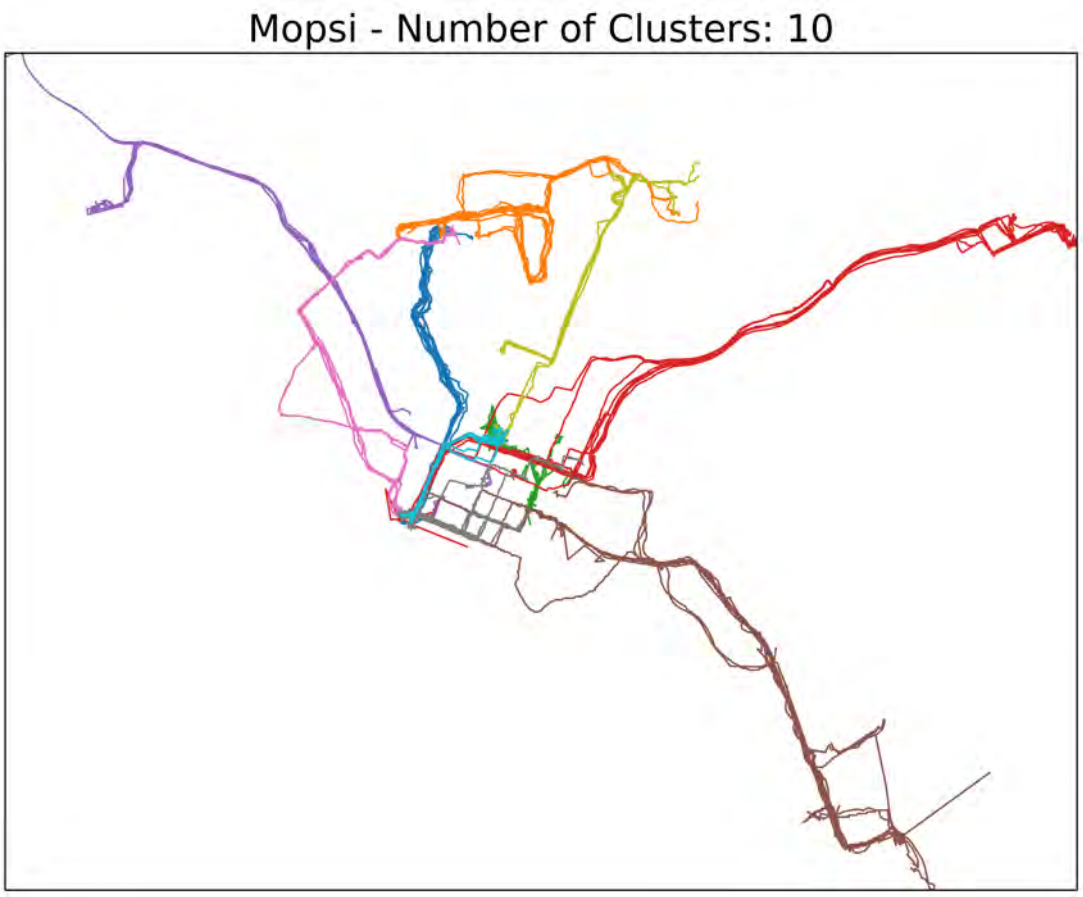

Figure 23: Mopsi Clusters

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 25 illustrates the process computed trajectory similarity.

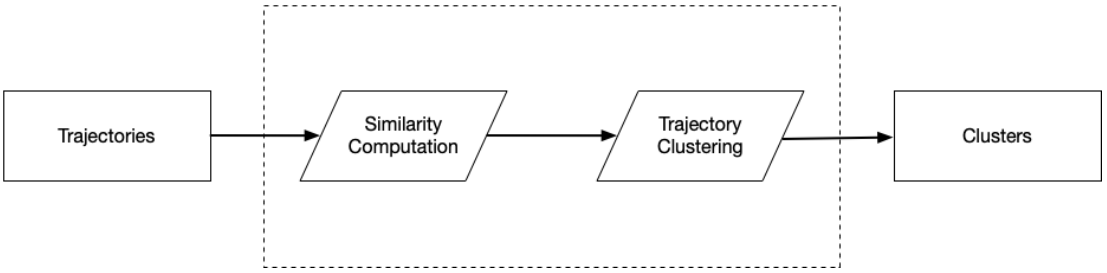

Figure 24: Overview of our process

Figure 24 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. Figure 25, 26, 27, 28, 29 are Mopsi clustering results using 5 distances DTW, LCSS, Hausdorff, Fréchet and HC-SIM with Ward Linkage cluster method. Due to ground-truth data and result from Silhouette scores in Figure 30, we used 10 as the number of clusters for both distances.

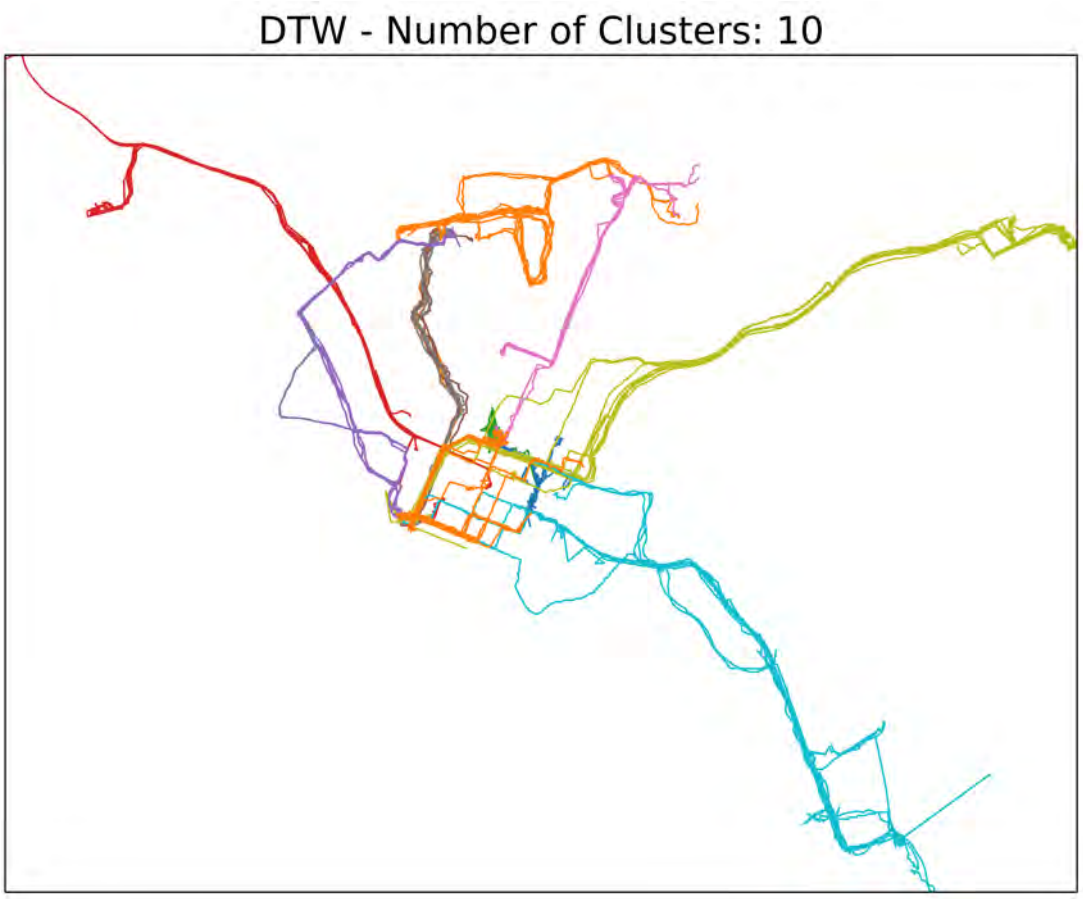

Figure 25: DTW Trajectories Clustering

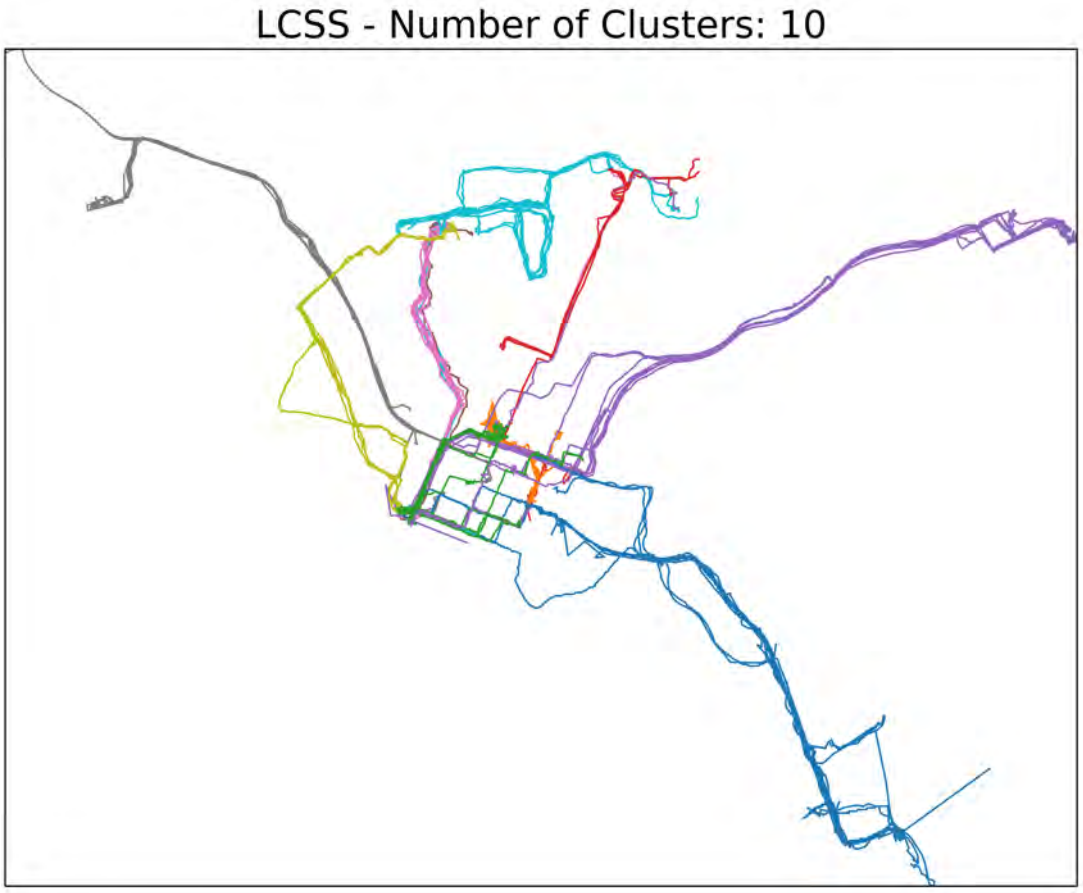

Figure 26: LCSS Trajectories Clustering

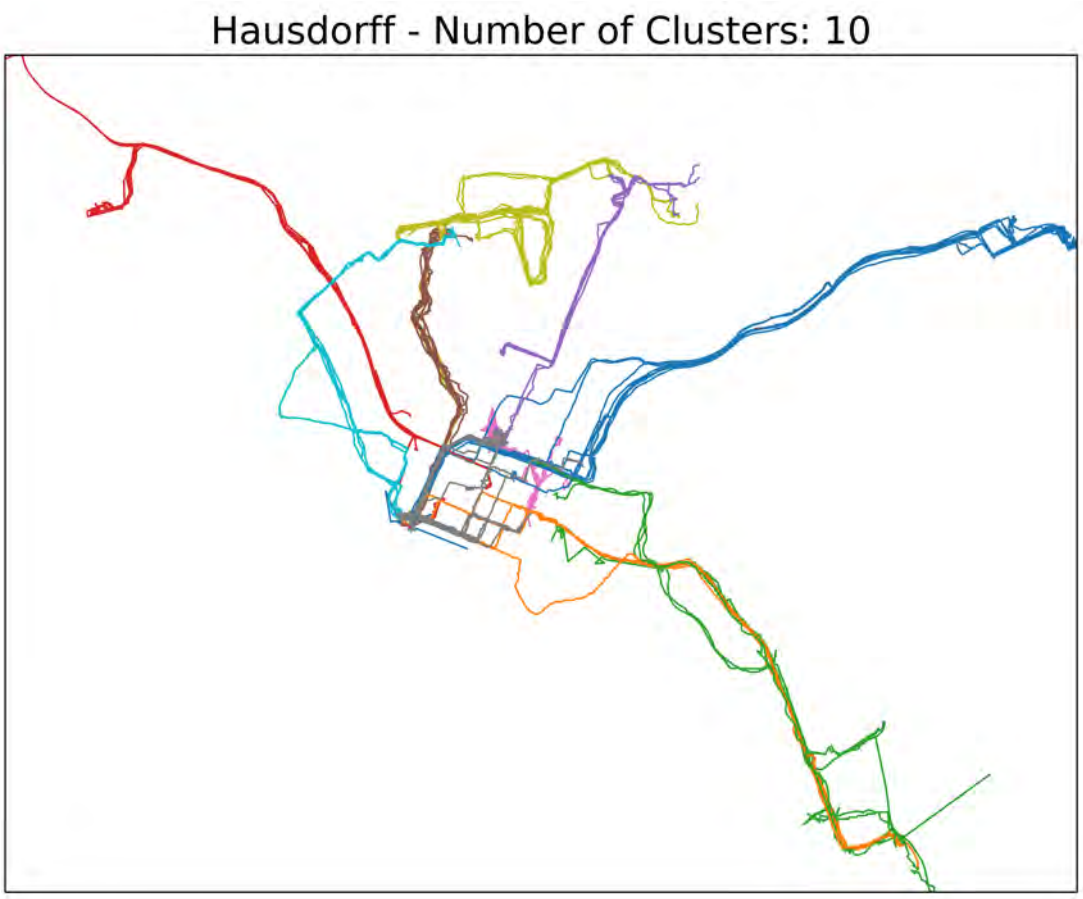

Figure 27: Hausdorff Trajectories Clustering

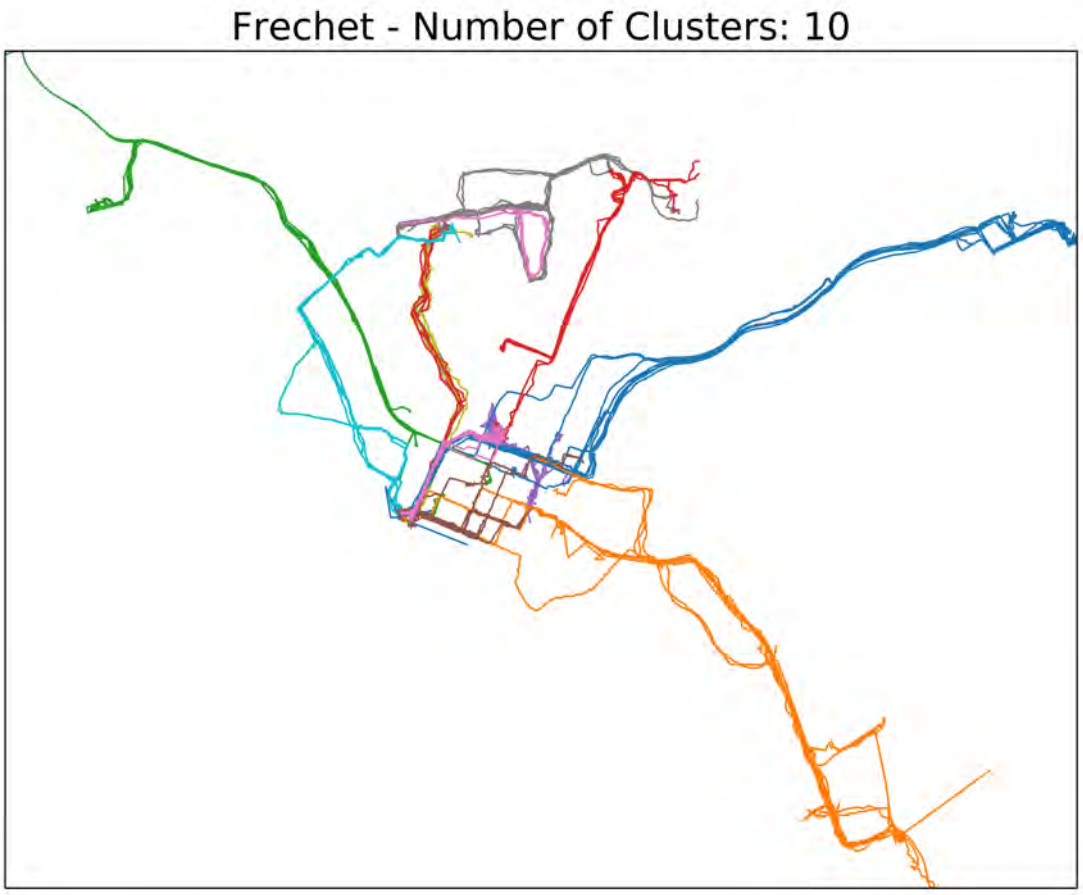

Figure 28: Fréchet Trajectories Clustering

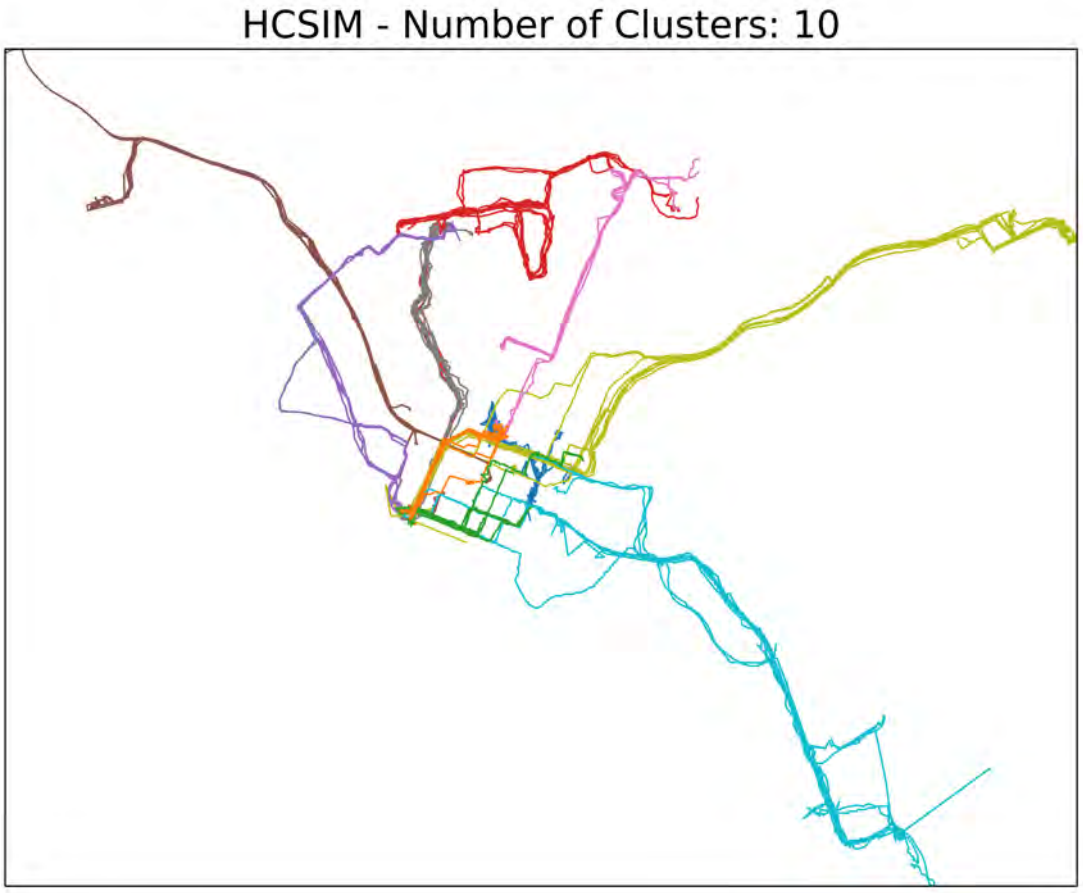

Figure 29: HC-SIM Trajectories Clustering

5.1.4 Clusters Validation

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

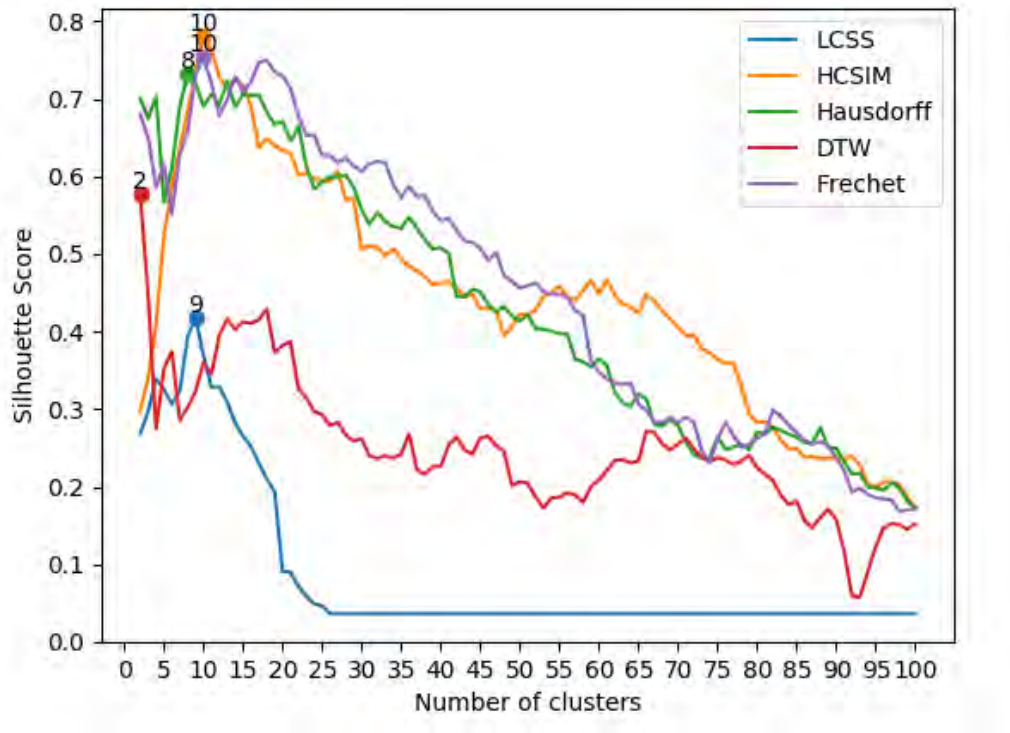

Figure 30: Silhouette Score depending on cluster

Figure 30 illustrates Silhouette Score based on number of clusters using 5 distances. 9 clusters are found with LCSS, 8 with Hausdorff and 10 with Fréchet, HC-SIM while DTW maximum at 2 clusters. HC-SIM and Fréchet give the best results as those equal with the ground-truth data. Table 5 summarizes Silhouette Scores for each distance with the number of clusters range from 2 to 100.

Table 5: Mopsi Silhouette Scores

Clusters	**HC-SIM**	**DTW**	**LCSS**	**Frechet**	**Hausdorff**
2	0.298768	**0.576919**	0.268120	0.680111	0.700506
3	0.340146	0.447950	0.297330	0.650049	0.674261
4	0.409051	0.274001	0.339466	0.586412	0.703117
5	0.523275	0.352317	0.324976	0.615615	0.566408
6	0.585378	0.374094	0.306599	0.550898	0.610376
7	0.635762	0.286120	0.323258	0.629292	0.689953
8	0.684482	0.302140	0.393962	0.657553	**0.731702**
9	0.735657	0.322905	**0.417437**	0.731107	0.720382
10	**0.780062**	0.360064	0.368766	**0.753502**	0.689732
11	0.762915	0.344460	0.328166	0.720096	0.705839
12	0.727740	0.395269	0.328786	0.678943	0.692692
13	0.715265	0.416145	0.309088	0.699268	0.722890
14	0.727732	0.402953	0.282231	0.728727	0.689695
15	0.716207	0.412412	0.264837	0.709048	0.705672
16	0.691088	0.410503	0.252075	0.721149	0.704394
17	0.637067	0.415109	0.230326	0.746946	0.704632
18	0.648691	0.429014	0.209700	0.749312	0.683025
19	0.640025	0.373099	0.193859	0.734810	0.666438
20	0.633123	0.382110	0.090747	0.729470	0.671273
21	0.630702	0.387021	0.089919	0.713444	0.645389
22	0.601537	0.325926	0.071233	0.678058	0.664584
23	0.605009	0.314910	0.058920	0.653062	0.613892
24	0.596298	0.297126	0.048962	0.653399	0.583749
25	0.595820	0.293018	0.046647	0.627256	0.592238
26	0.592593	0.279156	0.036524	0.626806	0.599689
27	0.606669	0.283243	0.036524	0.617713	0.600050
28	0.569360	0.266102	0.036524	0.624000	0.601084

29	0.571490	0.258856	0.036524	0.613484	0.585828
30	0.506735	0.262066	0.036524	0.606522	0.556518
31	0.509607	0.239391	0.036524	0.616589	0.538377
32	0.508362	0.236489	0.036524	0.619321	0.553602
33	0.497016	0.240112	0.036524	0.617608	0.541344
34	0.506722	0.236717	0.036524	0.588846	0.535160
35	0.494255	0.242230	0.036524	0.572727	0.533088
36	0.483844	0.268424	0.036524	0.587528	0.547418
37	0.478537	0.221440	0.036524	0.573856	0.533254
38	0.470792	0.215936	0.036524	0.575094	0.517699
39	0.460636	0.225875	0.036524	0.555199	0.505916
40	0.462903	0.226292	0.036524	0.542707	0.508282
41	0.464056	0.253523	0.036524	0.545994	0.500828
42	0.455542	0.264238	0.036524	0.530499	0.445557
43	0.445354	0.247471	0.036524	0.516365	0.444546
44	0.448261	0.241870	0.036524	0.515366	0.454323
45	0.429110	0.262778	0.036524	0.507534	0.451549
46	0.429727	0.265252	0.036524	0.491600	0.434674
47	0.427603	0.252751	0.036524	0.501876	0.425555
48	0.394285	0.245809	0.036524	0.472177	0.432253
49	0.406075	0.201550	0.036524	0.464704	0.420008
50	0.422484	0.206519	0.036524	0.456097	0.413475
51	0.421439	0.204488	0.036524	0.458883	0.421552
52	0.427637	0.186626	0.036524	0.463039	0.403804
53	0.445642	0.172709	0.036524	0.451198	0.402914
54	0.450296	0.185395	0.036524	0.448540	0.400690
55	0.458587	0.186349	0.036524	0.447741	0.397080
56	0.445061	0.192119	0.036524	0.443642	0.396747
57	0.439998	0.189489	0.036524	0.427029	0.364683
58	0.453032	0.179846	0.036524	0.420762	0.360847
59	0.465762	0.199841	0.036524	0.362922	0.354533
60	0.448620	0.208322	0.036524	0.347272	0.364386
61	0.466573	0.220661	0.036524	0.339197	0.356293
62	0.446709	0.233885	0.036524	0.334048	0.322012
63	0.435967	0.234664	0.036524	0.331575	0.308222

64	0.431730	0.230722	0.036524	0.333847	0.303543
65	0.423495	0.232958	0.036524	0.306664	0.319391
66	0.447461	0.271923	0.036524	0.298539	0.313171
67	0.439251	0.269800	0.036524	0.284006	0.280589
68	0.427755	0.255420	0.036524	0.282251	0.278021
69	0.417001	0.247607	0.036524	0.288925	0.286622
70	0.404768	0.254682	0.036524	0.283375	0.278888
71	0.394307	0.262851	0.036524	0.289291	0.257771
72	0.395823	0.248873	0.036524	0.283508	0.241314
73	0.377412	0.238570	0.036524	0.246551	0.236627
74	0.372619	0.231679	0.036524	0.230566	0.237244
75	0.364679	0.237797	0.036524	0.261544	0.264191
76	0.358742	0.234606	0.036524	0.282366	0.248876
77	0.359416	0.228435	0.036524	0.265959	0.250805
78	0.329923	0.233125	0.036524	0.250782	0.254994
79	0.293541	0.240507	0.036524	0.254785	0.247424
80	0.283138	0.224590	0.036524	0.261961	0.270319
81	0.283623	0.216256	0.036524	0.268322	0.268467
82	0.276640	0.208296	0.036524	0.298819	0.276532
83	0.260265	0.188966	0.036524	0.290552	0.272658
84	0.248923	0.177245	0.036524	0.279467	0.267682
85	0.249352	0.181567	0.036524	0.269299	0.263888
86	0.237731	0.156187	0.036524	0.260316	0.256741
87	0.238893	0.147227	0.036524	0.254717	0.255690
88	0.236788	0.160385	0.036524	0.257936	0.276045
89	0.237053	0.170921	0.036524	0.253230	0.250608
90	0.237544	0.157940	0.036524	0.237115	0.249617
91	0.236719	0.117585	0.036524	0.220939	0.232712
92	0.238922	0.059248	0.036524	0.192998	0.216464
93	0.227743	0.057335	0.036524	0.198360	0.217302
94	0.204179	0.090383	0.036524	0.189835	0.199266
95	0.199148	0.122596	0.036524	0.185711	0.198327
96	0.205939	0.147056	0.036524	0.183896	0.195421
97	0.207049	0.152479	0.036524	0.182956	0.205165
98	0.202183	0.150998	0.036524	0.167628	0.197993

99	0.185975	0.145077	0.036524	0.170454	0.181664
100	0.174085	0.152291	0.036524	0.170831	0.171941

Additionally, we used Pair Sets Index to evaluate cluster results from those 5 distances. Figure 31 shows that HC-SIM (shaped base distance) give the better cluster results than warping-based distances with Pair Sets Index is the highest with red color indicates the index is over 0.8 while blue indicates value between 0.7 and 0.8, green indicates value is less than 0.7 (details in Table 6).

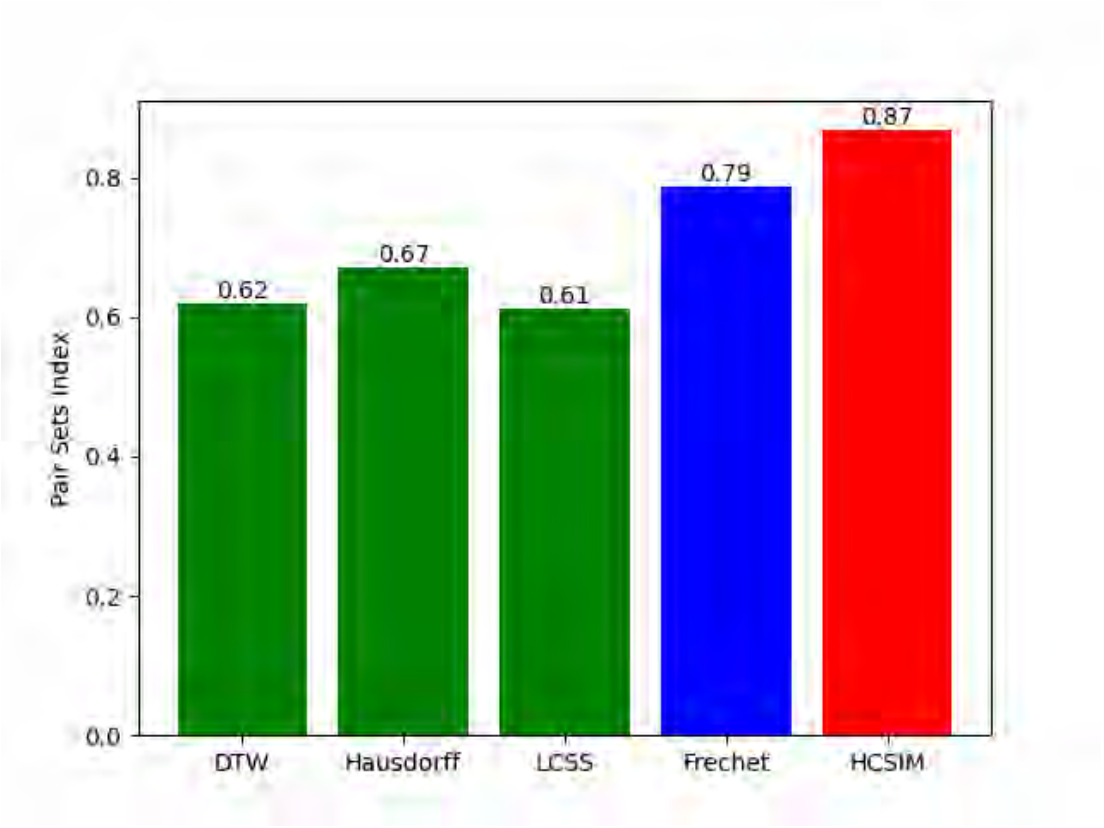

Figure 31: Pair Sets Index based on distance

Table 6: Pair Sets Index

DTW	**LCSS**	**Fréchet**	**Hausdorff**	**HC-SIM**
0.62	0.61	0.79	0.67	0.87

Figure 32, 33, 34, 35, 36, 37, 38, 39, 40, 41 show that all distances seem identical with ground-truth data in some clusters. However, HC-SIM does the best, nearly match ground-truth data while the other distances have some conflicts.

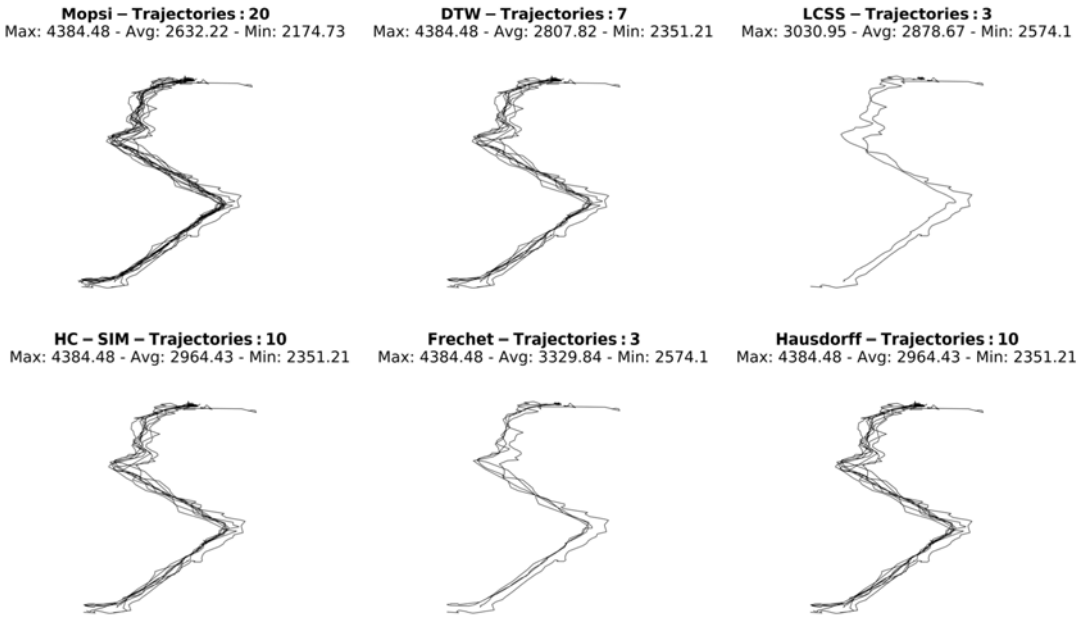

Figure 32: Trajectories Clusters 1

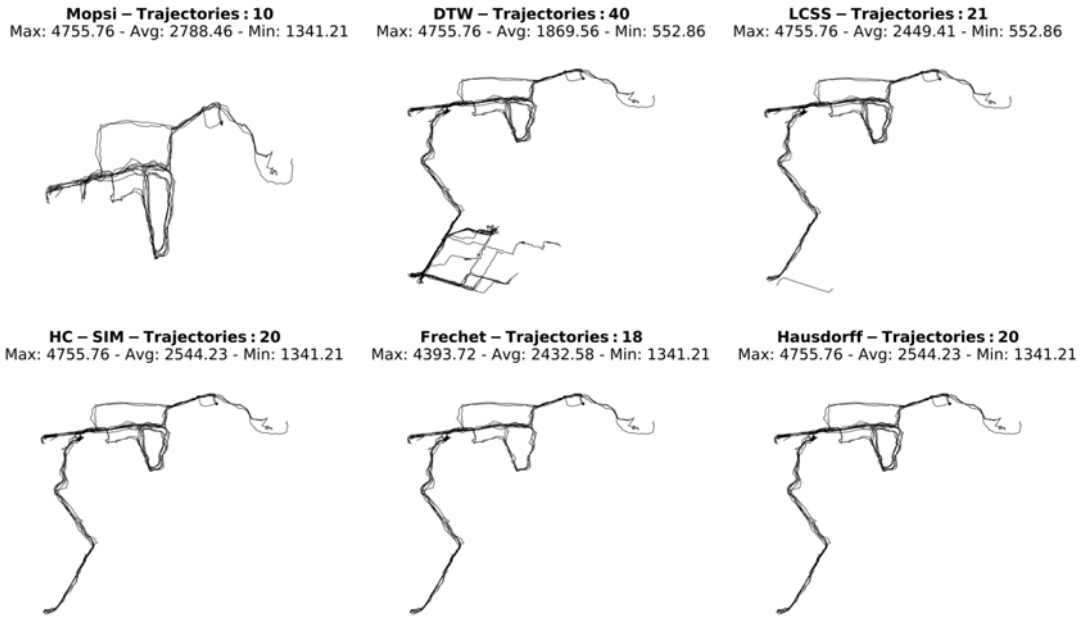

Figure 33: Trajectories Clusters 2

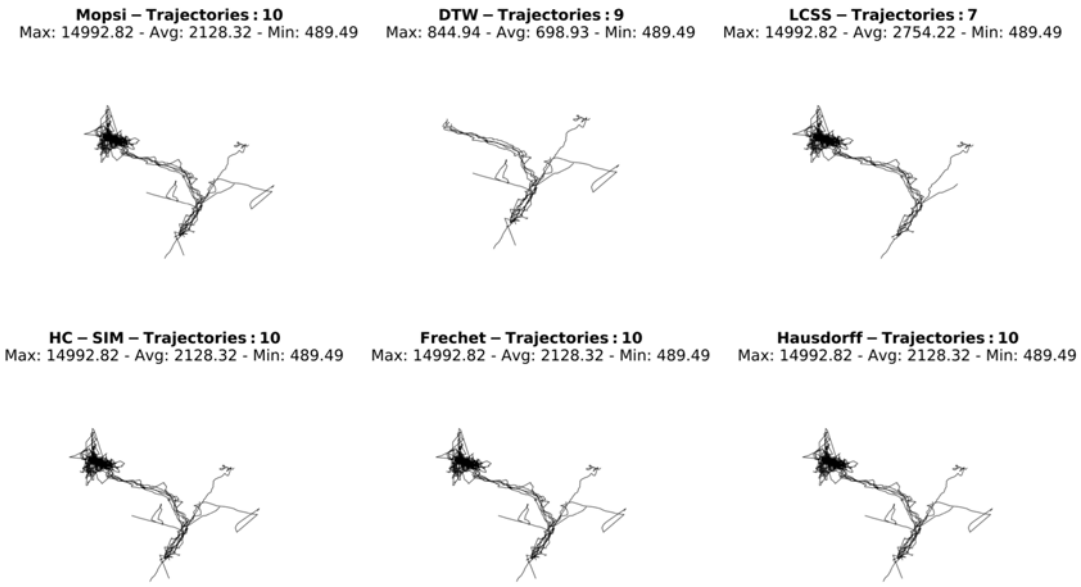

Figure 34: Trajectories Clusters 3

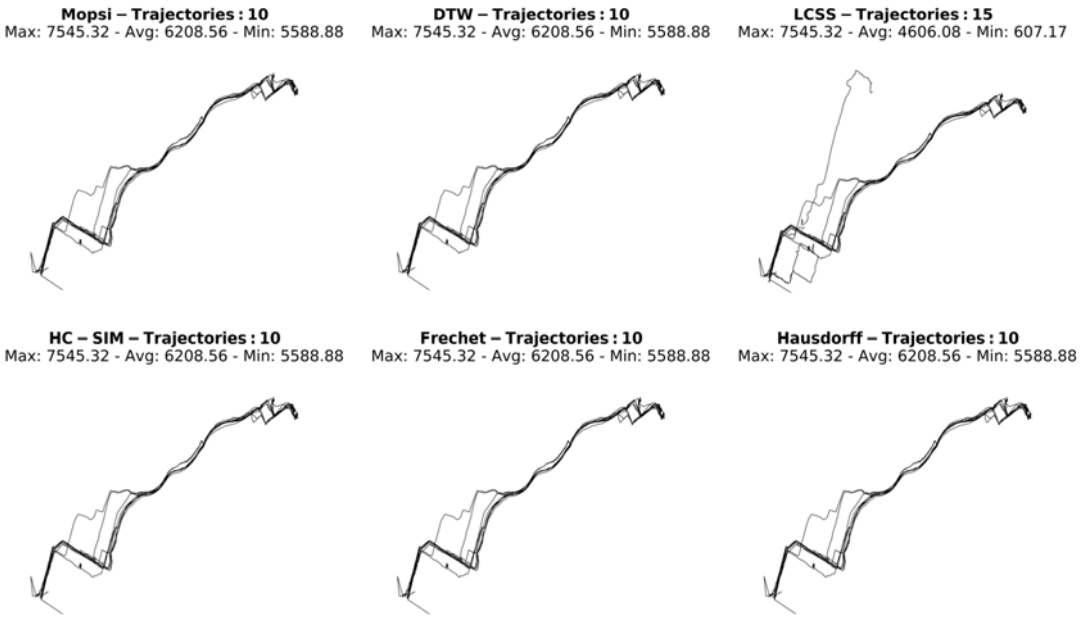

Figure 35: Trajectories Clusters 4

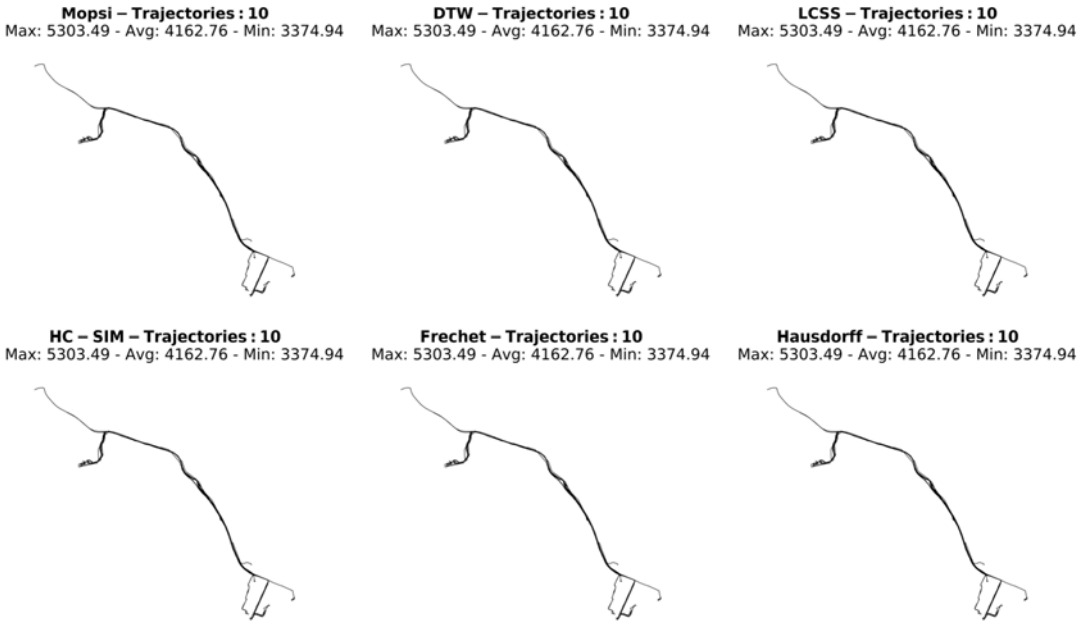

Figure 36: Trajectories Clusters 5

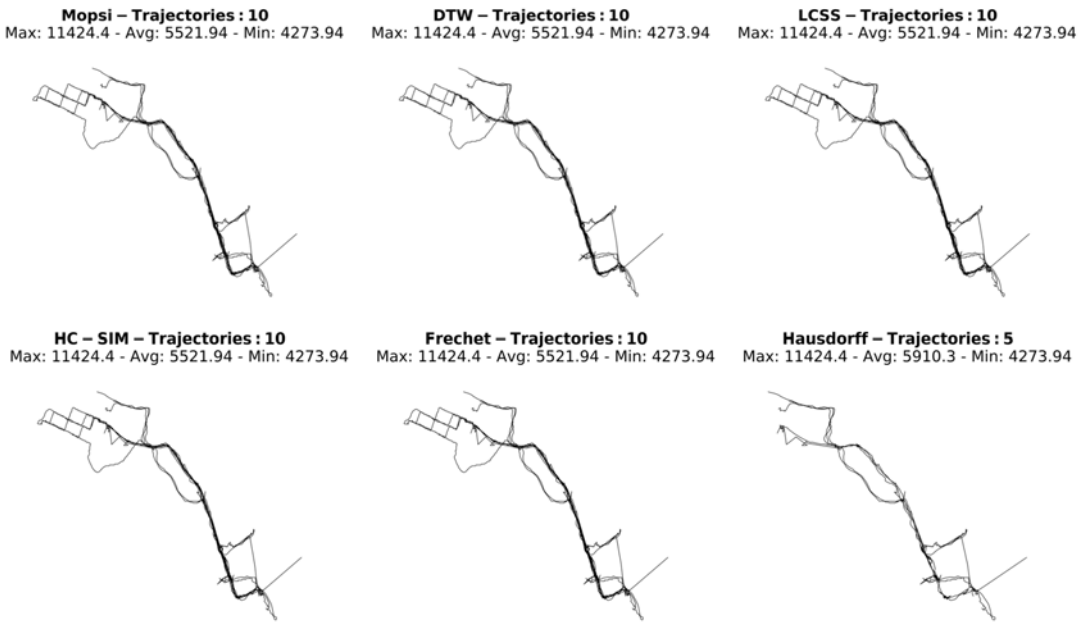

Figure 37: Trajectories Clusters 6

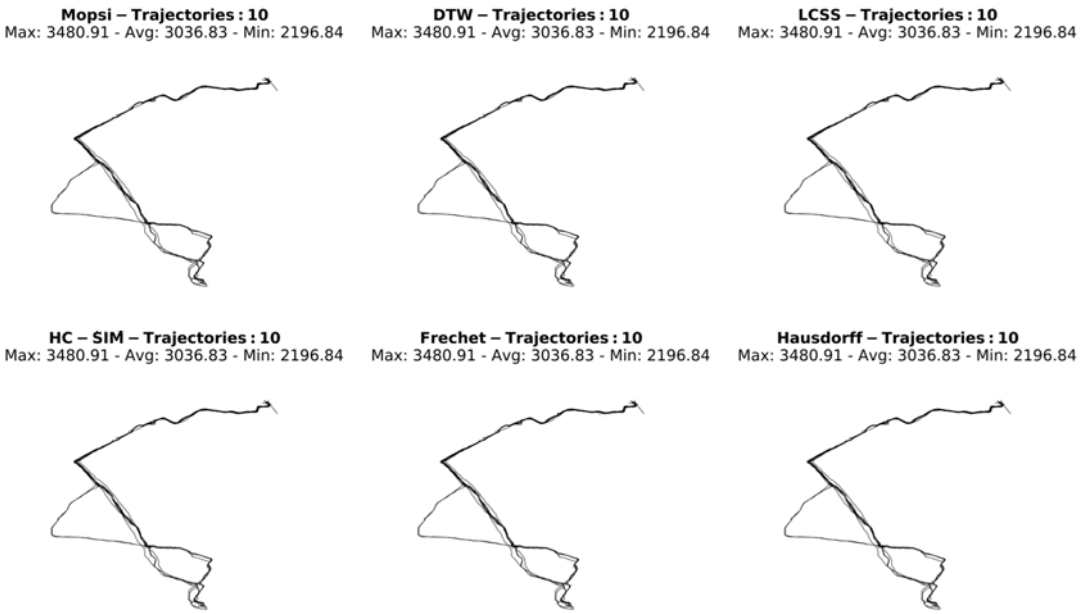

Figure 38: Trajectories Clusters 7

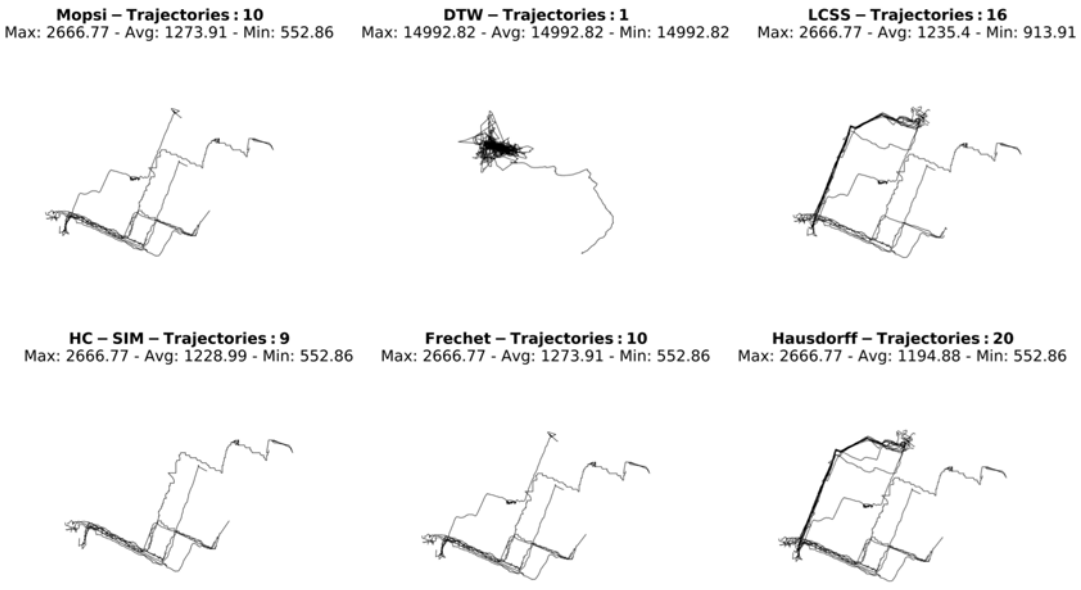

Figure 39: Trajectories Clusters 8

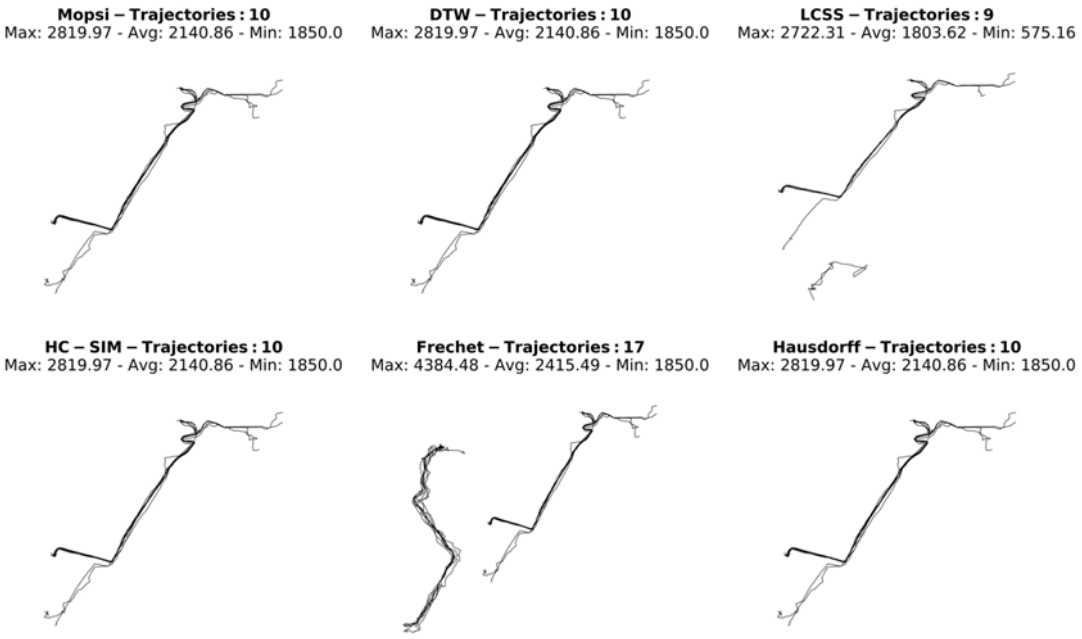

Figure 40: Trajectories Clusters 9

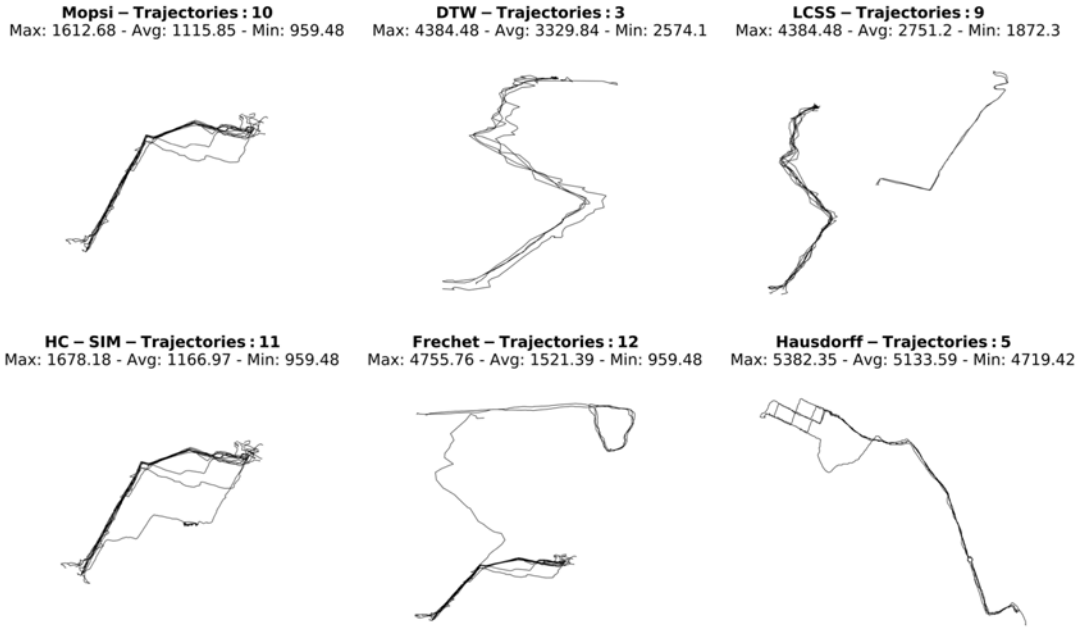

Figure 41: Trajectories Clusters 10

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 preprocessed data extracting 2574 trajectories with the same start point as black a point in Figure 42 at Caltrain station and red points indicate the end points which are drop-off locations.

Figure 42: Caltrain Trajectories Dataset

5.2.2 Analysis of the distances

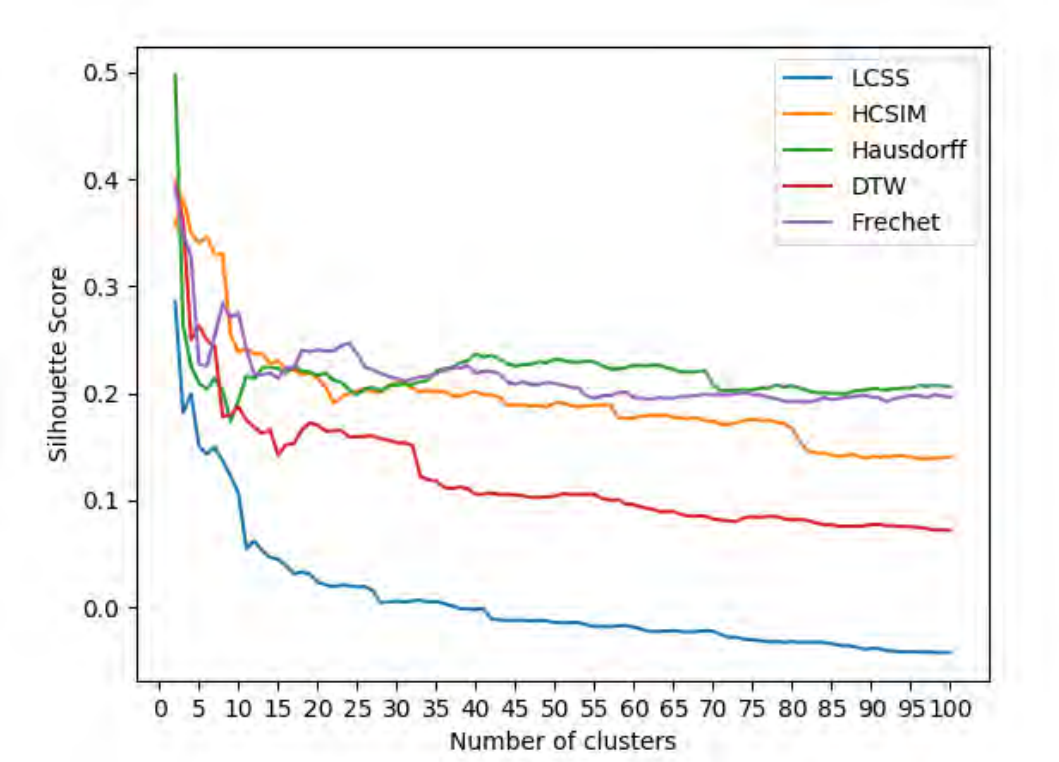

Figure 43: Silhouette Score

Figure 43 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, even LCSS got negative scores. Table 7 summarizes Silhouette Scores for each distance with the number of clusters range from 2 to 100. We chose 15 as the number of clusters for all distances then compared the results.

Table 7: Caltrain Silhouette Scores

Clusters	**HC-SIM**	**DTW**	**LCSS**	**Frechet**	**Hausdorff**
2	0.356769	0.396301	0.285650	0.392788	0.497644
3	0.383545	0.361110	0.181747	0.347808	0.263201
4	0.349602	0.250431	0.199539	0.328405	0.225499
5	0.340964	0.262857	0.150745	0.227275	0.209657
6	0.346317	0.250004	0.143371	0.225573	0.203689
7	0.329542	0.244371	0.149779	0.254266	0.214275

8	0.330136	0.178117	0.136986	0.283555	0.202706
9	0.254469	0.180204	0.122737	0.270990	0.173631
10	0.238627	0.187296	0.106858	0.275026	0.193501
11	0.241977	0.175068	0.055147	0.241094	0.216469
12	0.236761	0.168340	0.062208	0.217643	0.213491
13	0.237156	0.162590	0.052939	0.217235	0.223876
14	0.227760	0.165999	0.046684	0.219286	0.224394
15	0.230400	0.142160	0.045190	0.214171	0.223323
16	0.222628	0.152119	0.038973	0.224474	0.219284
17	0.221845	0.153264	0.031790	0.222994	0.225172
18	0.217480	0.165421	0.033072	0.240360	0.221023
19	0.219536	0.172252	0.031080	0.239041	0.219824
20	0.214589	0.170462	0.023866	0.240447	0.216707
21	0.206516	0.164810	0.021384	0.239794	0.218972
22	0.191256	0.164906	0.019465	0.239200	0.212498
23	0.196210	0.165379	0.020849	0.244760	0.210494
24	0.200200	0.159268	0.020066	0.246877	0.203208
25	0.201813	0.159507	0.019265	0.237252	0.198961
26	0.204181	0.159889	0.019530	0.224376	0.204022
27	0.201373	0.160609	0.015587	0.222180	0.205813
28	0.201114	0.157332	0.004293	0.218555	0.202371
29	0.207140	0.156469	0.005048	0.215885	0.206669
30	0.211101	0.153877	0.005816	0.214025	0.207684
31	0.210175	0.153821	0.005007	0.211167	0.208809
32	0.205037	0.151434	0.006237	0.213311	0.208455
33	0.201024	0.122437	0.006821	0.215603	0.210904
34	0.203483	0.119329	0.004895	0.216018	0.212265
35	0.200473	0.118081	0.005463	0.217762	0.220704
36	0.202384	0.112060	0.003677	0.222247	0.223063
37	0.197390	0.111081	0.001623	0.224556	0.223387
38	0.197010	0.112646	-0.001083	0.223489	0.228722
39	0.200355	0.110885	-0.001366	0.226141	0.230493
40	0.201805	0.105598	-0.001820	0.218708	0.236885
41	0.198826	0.106026	-0.001056	0.220735	0.233837
42	0.198403	0.106980	-0.011015	0.220628	0.235701

43	0.196807	0.105476	-0.011598	0.218004	0.231970
44	0.189037	0.105457	-0.012231	0.210478	0.227581
45	0.188693	0.105263	-0.012043	0.208605	0.225654
46	0.189510	0.104289	-0.011958	0.210598	0.226802
47	0.187696	0.102868	-0.012662	0.208361	0.227238
48	0.188798	0.102793	-0.012089	0.208134	0.229546
49	0.187074	0.103403	-0.012561	0.210294	0.228537
50	0.191232	0.103788	-0.014111	0.209117	0.232126
51	0.191061	0.106281	-0.014281	0.207811	0.230924
52	0.188551	0.105691	-0.014113	0.205573	0.229375
53	0.187468	0.105492	-0.014115	0.205149	0.229310
54	0.187984	0.105837	-0.015897	0.197593	0.229700
55	0.188822	0.105679	-0.018104	0.195543	0.229799
56	0.189136	0.101598	-0.017671	0.197964	0.225874
57	0.189378	0.100226	-0.018083	0.197667	0.222617
58	0.177276	0.100837	-0.017139	0.200659	0.222915
59	0.176738	0.096370	-0.016870	0.201189	0.224060
60	0.177556	0.096224	-0.018398	0.195921	0.226399
61	0.178655	0.093656	-0.020441	0.195627	0.225817
62	0.179329	0.092366	-0.022498	0.194352	0.226085
63	0.179318	0.089905	-0.022514	0.196077	0.226631
64	0.179784	0.089470	-0.022062	0.195626	0.224136
65	0.177671	0.089635	-0.021617	0.195710	0.222778
66	0.176759	0.086727	-0.022680	0.197178	0.219825
67	0.176586	0.085264	-0.022528	0.197277	0.220204
68	0.177366	0.085366	-0.022400	0.198324	0.220077
69	0.174087	0.085466	-0.021788	0.199188	0.221409
70	0.174291	0.082765	-0.022104	0.198879	0.207809
71	0.170972	0.081721	-0.025239	0.199053	0.202763
72	0.170995	0.080905	-0.027962	0.198781	0.202938
73	0.172749	0.080132	-0.027751	0.199451	0.202961
74	0.174867	0.084109	-0.029748	0.200830	0.203637
75	0.175850	0.084758	-0.030406	0.198680	0.202993
76	0.174353	0.084321	-0.030973	0.198616	0.205167
77	0.174953	0.085100	-0.032215	0.196021	0.204680

78	0.173249	0.084901	-0.031974	0.195357	0.208015
79	0.171855	0.082722	-0.032416	0.192647	0.206051
80	0.167248	0.081761	-0.032063	0.192481	0.207178
81	0.154736	0.082040	-0.032430	0.192582	0.205017
82	0.146156	0.081407	-0.032513	0.192138	0.202493
83	0.144325	0.078929	-0.032250	0.193020	0.200483
84	0.144089	0.077166	-0.032722	0.196295	0.200646
85	0.142328	0.077269	-0.033828	0.194151	0.200086
86	0.141368	0.075914	-0.035315	0.195398	0.200021
87	0.142270	0.075813	-0.035765	0.196780	0.200021
88	0.142786	0.075910	-0.036718	0.197130	0.202505
89	0.139398	0.076090	-0.038886	0.198587	0.202735
90	0.140799	0.077447	-0.038330	0.196552	0.204501
91	0.140396	0.077685	-0.038376	0.195879	0.203706
92	0.140974	0.076298	-0.040412	0.192209	0.203204
93	0.141698	0.076249	-0.040812	0.195462	0.204461
94	0.141920	0.075422	-0.041572	0.196538	0.205025
95	0.140590	0.075472	-0.041368	0.198024	0.204963
96	0.139228	0.074794	-0.041593	0.198072	0.207391
97	0.139199	0.073772	-0.041751	0.195956	0.206780
98	0.139347	0.072257	-0.041865	0.199066	0.207357
99	0.139909	0.072388	-0.041977	0.197660	0.206953
100	0.140607	0.072018	-0.041886	0.196479	0.206427

Figure 44 illustrates visual results for all distances with 15 clusters. Based on Mopsi experiment, HC-SIM is expected to be the best. We observe that trajectories are well classified by HC-SIM according to their paths in Figure 45, 46, 47, 48, 49. The clusters using shape-based distances seem to be consistent as the trajectories grouped better with less noise than warping-based distances. LCSS gives the poorest result containing a lot of noise in each cluster.

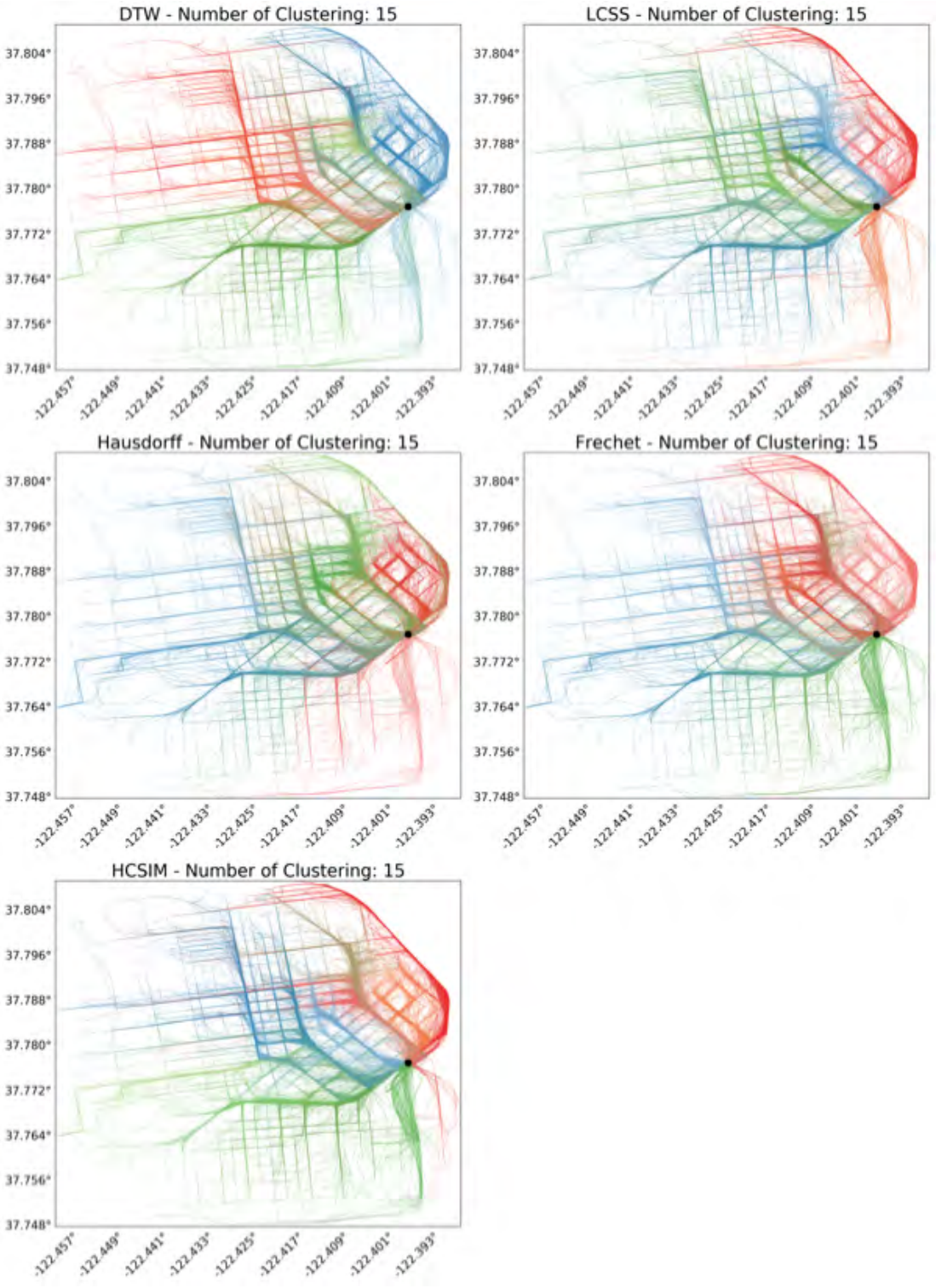

Figure 44: Caltrain Clusters

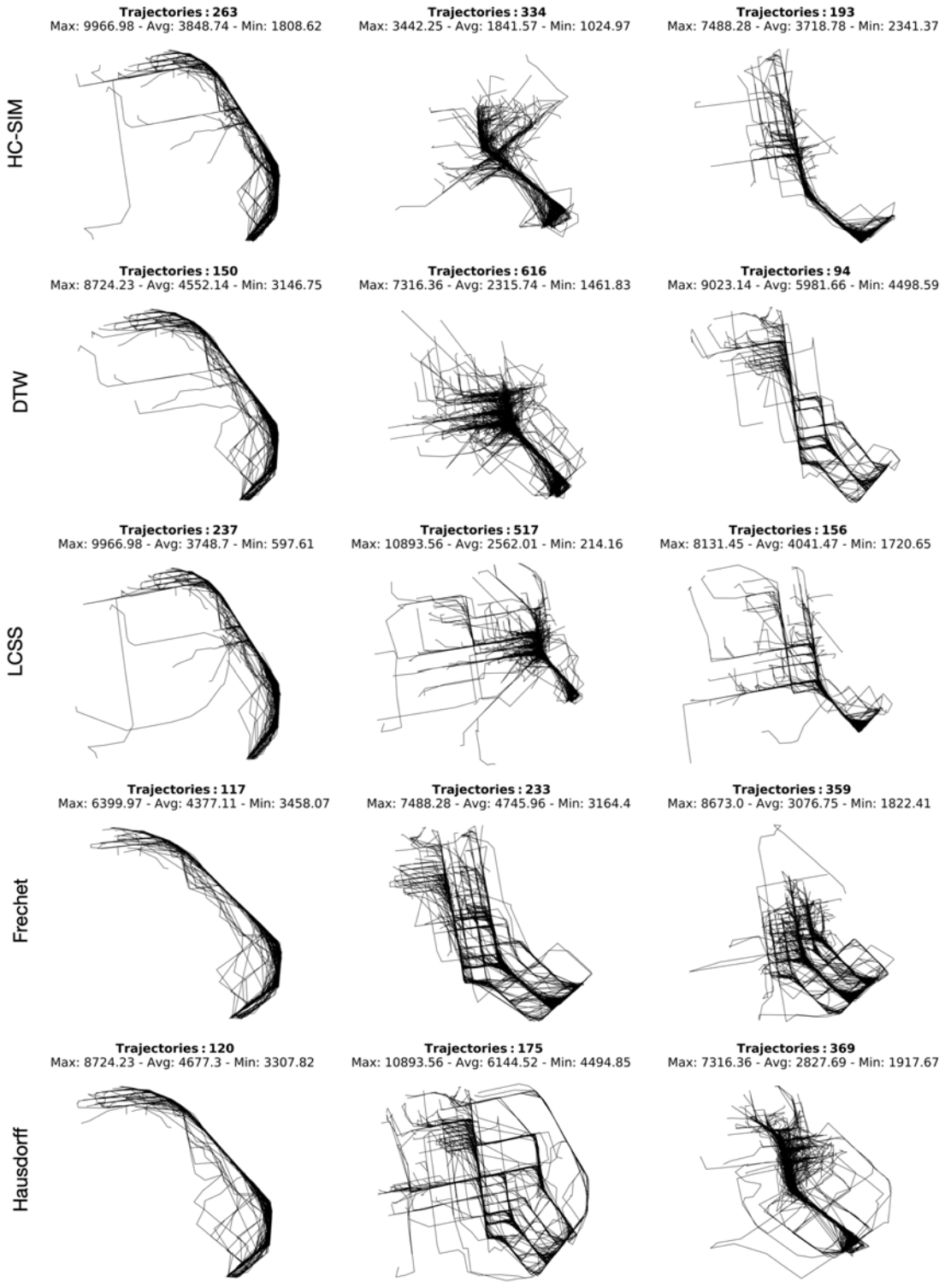

Figure 45: Caltrain Clusters 1

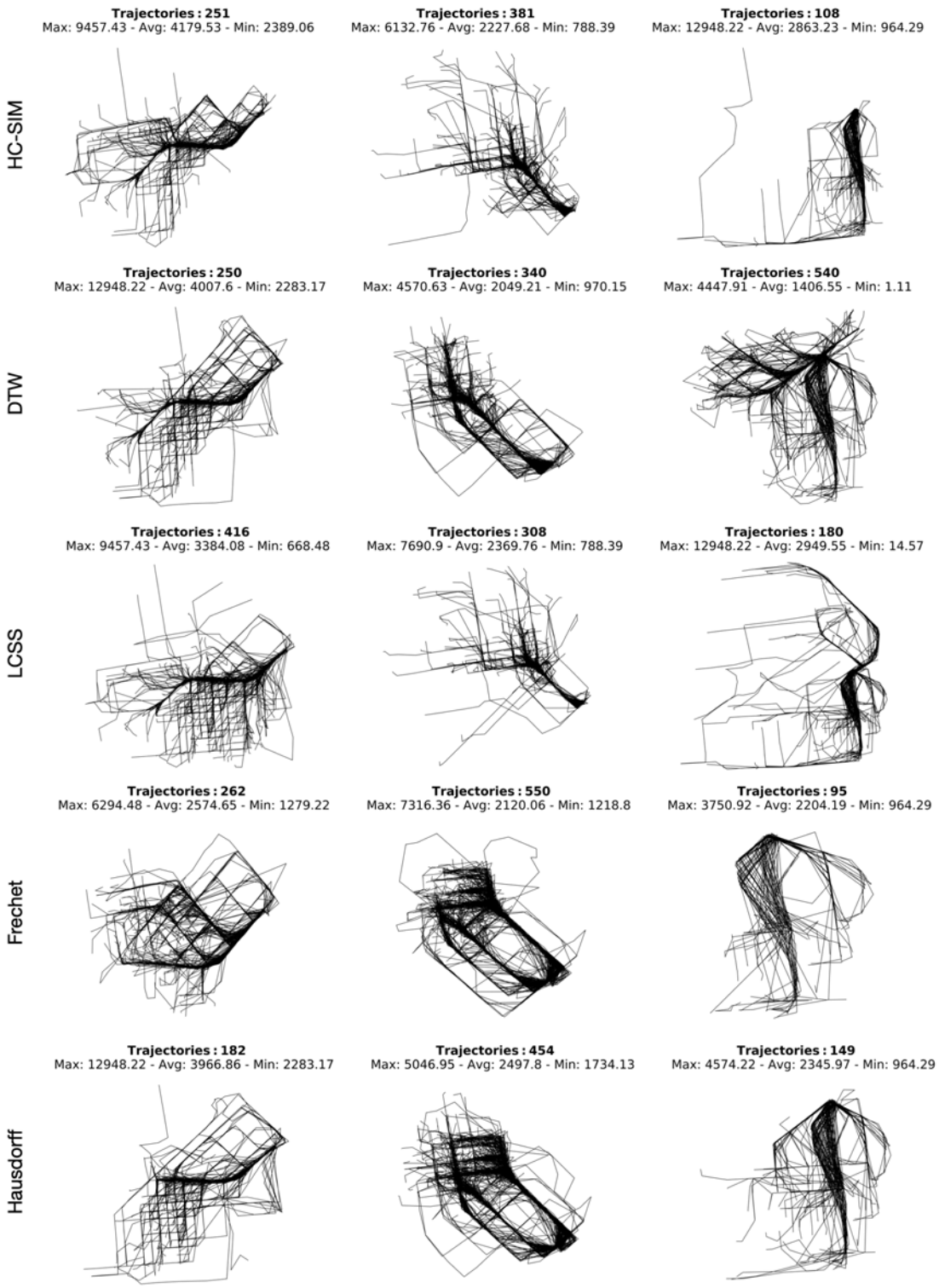

Figure 46: Caltrain Clusters 2

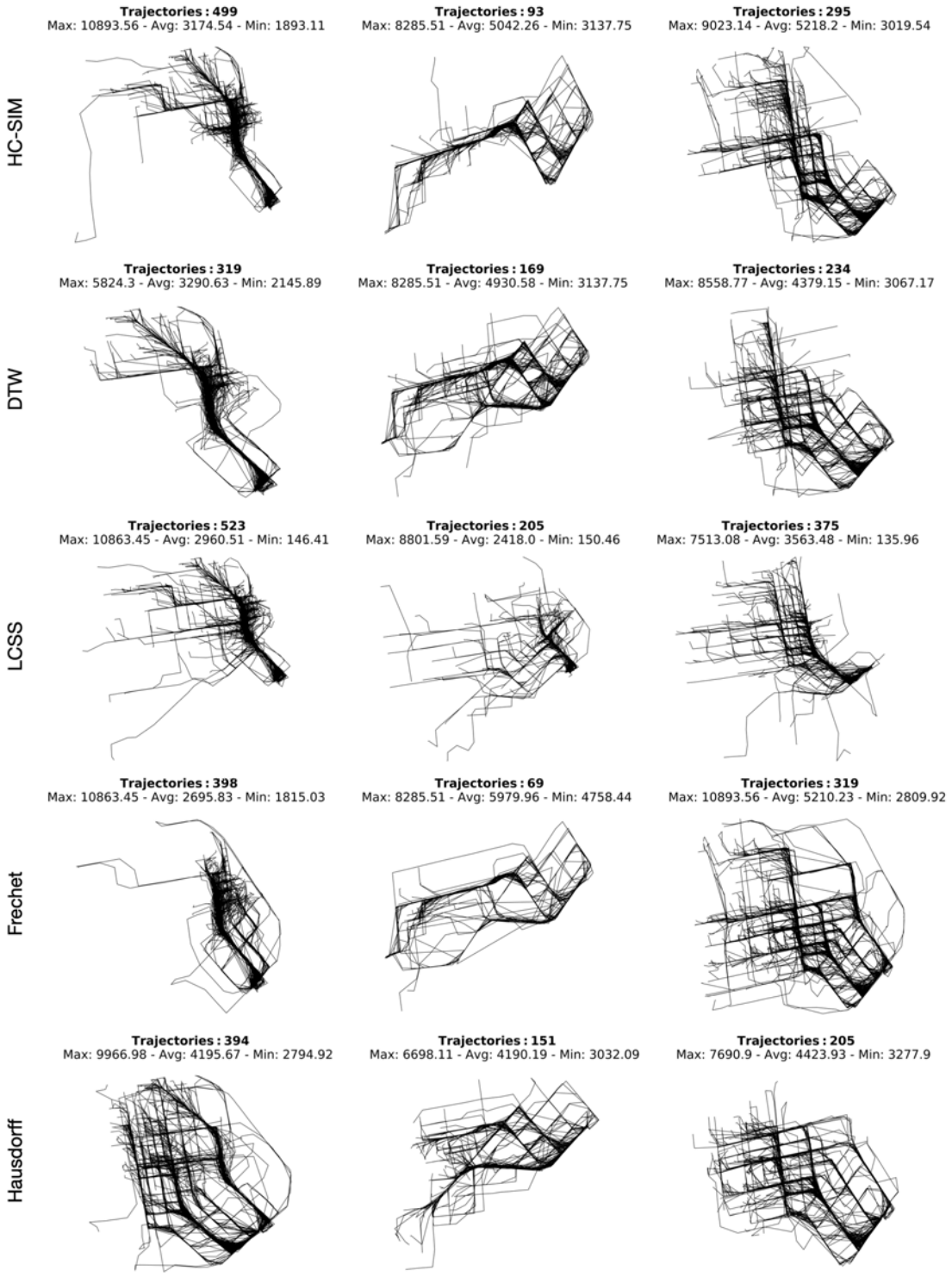

Figure 47: Caltrain Clusters 3

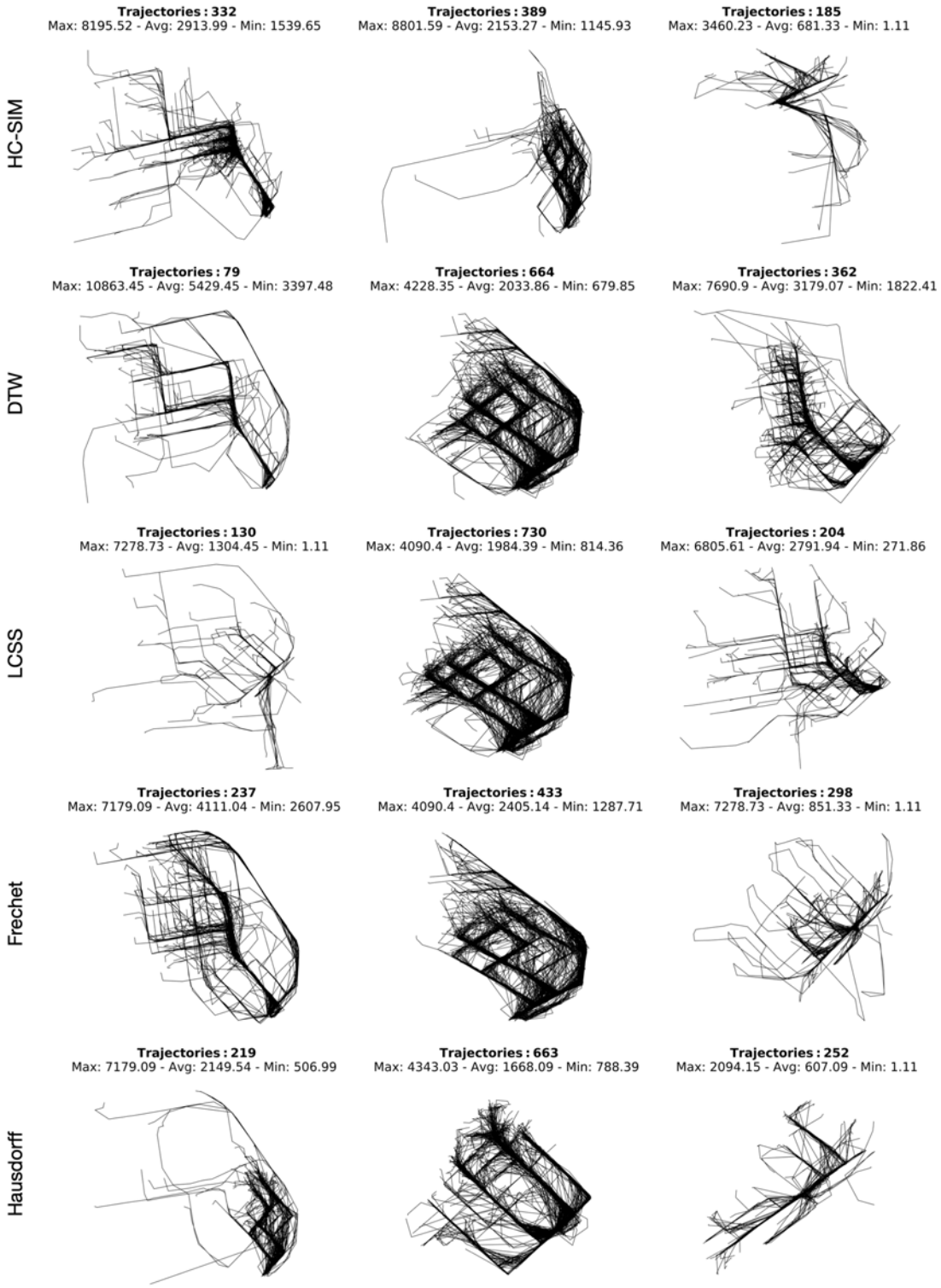

Figure 48: Caltrain Clusters 4

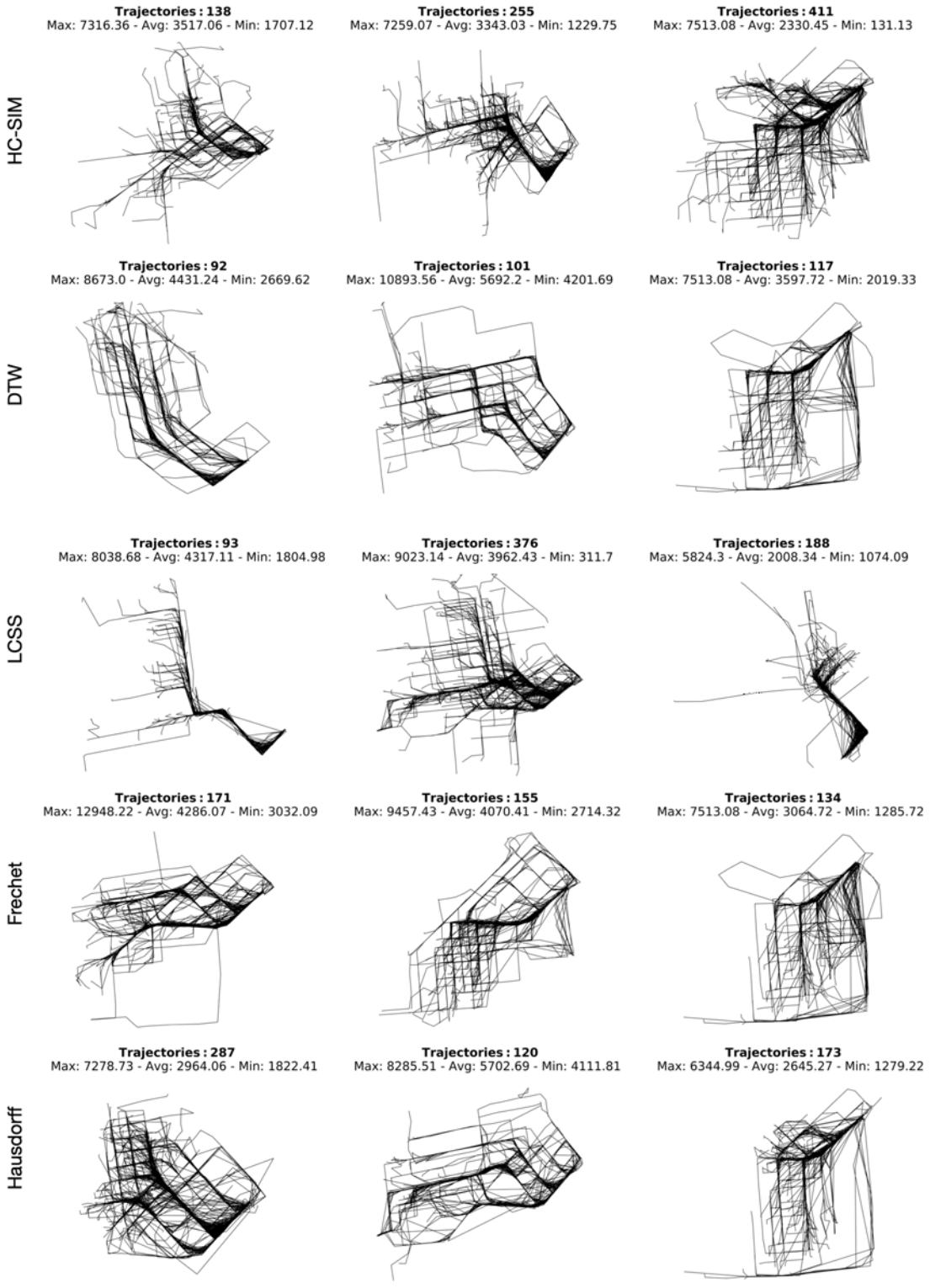

Figure 49: Caltrain Clusters 5

6 Conclustions

Clustering trajectories is heavily dependent on the selection of an appropriate distance. In this thesis, we explored several distances (DTW, LCSS, Fréchet, Hausdorff, and HC-SIM) concentrating on various characteristics of trajectories. All distances seem to work well with the hierarchical agglomerative clustering (HAC) method to cluster trajectories. Based on our experiment, shape-based distances give better clusters than warping-base ones, especially HC-SIM enables us to obtain good clusters that nearly match ground-truth data.

We used the Silhouette score and Pair Sets Index to validate trajectories' clusters. Both clustering and validation methods that we used in this thesis work well and give good results. This may be used to address a variety of problems. From learning moving objects' behavior, many applications can recommend locations to visit or arrange a 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.

Ahmed, M., Karagiorgou, S., Pfoser, D., & Wenk, C. (2015). A comparison and evaluation of map construction algorithms using vehicle tracking data. *GeoInformatica*, *19*(3), 601–632.

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*.

Biagioni, J., & Eriksson, J. (2012). Inferring road maps from global positioning system traces: Survey and comparative evaluation. *Transportation research record*, *2291*(1), 61–71.

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.

Burkard, R., Dell'Amico, M., & Martello, S. (2012). *Assignment problems: revised reprint*. SIAM.

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).

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*(2), 224–227.

Dunn, J. C. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, *4*(1), 95–104.

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. (2018). Efficiency of random swap clustering. *Journal of Big Data*, *5*(1), 1–29.

Fränti, P., & Mariescu-Istodor, R. (2021). Averaging gps segments competition 2019. *Pattern Recognition*, *112*, 107730.

Fränti, P., Rezaei, M., & Zhao, Q. (2014). Centroid index: Cluster level similarity measure. *Pattern Recognition*, *47*(9), 3034–3045.

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.

Fränti, P., & Virmajoki, O. (2006). Iterative shrinking method for clustering problems. *Pattern Recognition*, *39*(5), 761–775.

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.

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of classification*, *2*(1), 193–218.

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).

Liu, S., Ni, L. M., & Krishnan, R. (2013). Fraud detection from taxis' driving behaviors. *IEEE Transactions on Vehicular Technology*, *63*(1), 464–472.

Malinen, M. I., & Fränti, P. (2014). Balanced k-means for clustering. In *Joint iapr international workshops on statistical techniques in pattern recognition (spr) and structural and syntactic pattern recognition (sspr)* (pp. 32–41).

Malinen, M. I., Mariescu-Istodor, R., & Fränti, P. (2014). K-means∗: Clustering by gradual data transformation. *Pattern Recognition*, *47*(10), 3376–3386.

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.

Mariescu-Istodor, R., & Fränti, P. (2018). Cellnet: Inferring road networks from gps trajectories. *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, *4*(3), 1–22.

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.

Nallusamy, R., Duraiswamy, K., Dhanalaksmi, R., & Parthiban, P. (2010). Optimization of non-linear multiple traveling salesman problem using k-means clustering, shrink wrap algorithm and meta-heuristics. *International Journal of Nonlinear Science*, *9*(2), 171–177.

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.

Rezaei, M., & Franti, P. (2018). Real-time clustering of large geo-referenced data for visualizing on map. *Advances in Electrical and Computer Engineering*, *18*(4), 63–74.

Rodriguez, A., & Laio, A. (2014). Clustering by fast search and find of density peaks. *science*, *344*(6191), 1492–1496.

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.

Siavoshi, S., Kavian, Y. S., & Sharif, H. (2016). Load-balanced energy efficient clustering protocol for wireless sensor networks. *IET Wireless Sensor Systems*, *6*(3), 67–73.

Sieranoja, S., & Fränti, P. (2019). Fast and general density peaks clustering. *Pattern Recognition Letters*, *128*, 551–558.

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).

Xie, X. L., & Beni, G. (1991). A validity measure for fuzzy clustering. *IEEE Transactions on pattern analysis and machine intelligence*, *13*(8), 841–847.

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, Q., & Fränti, P. (2014). Wb-index: A sum-of-squares based index for cluster validity. *Data & Knowledge Engineering*, *92*, 77–89.

Zhao, Q., Shi, Y., Liu, Q., & Fränti, P. (2015). A grid-growing clustering algorithm for geo-spatial data. *Pattern Recognition Letters*, *53*, 77–84.

Zhao, Q., Xu, M., & Fränti, P. (2008). Knee point detection on bayesian information criterion. In *2008 20th ieee international conference on tools with artificial intelligence* (Vol. 2, pp. 431–438).

Zhao, Q., Xu, M., & Fränti, P. (2009). Sum-of-squares based cluster validity index and significance analysis. In *International conference on adaptive and natural computing algorithms* (pp. 313–322).

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

Zhong, C., Miao, D., & Wang, R. (2010). A graph-theoretical clustering method based on two rounds of minimum spanning trees. *Pattern Recognition*, *43*(3), 752–766.

Zhong, S., & Ghosh, J. (2003). Model-based clustering with soft balancing. In *Third ieee international conference on data mining* (pp. 459–466).

Zoubi, M. B. A., & Rawi, M. a. (2008). An efficient approach for computing silhouette coefficients. *Journal of computer science*, *4*(3), 252.