A Scalable Framework for Clustering Vehicle Trajectories in a Dense Road Network

Dheeraj Kumar*, Sutharshan Rajasegarar§#, Marimuthu Palaniswami*,
Xiaoting Wang§, and Christopher Leckie§#
*Electrical and Electronic Engineering §Computing and Information Systems
*§The University of Melbourne, Australia #National ICT Australia - Victoria
{dheerajk@student., sraja@, palani@, wangx5@student., caleckie@} unimelb.edu.au

ABSTRACT

Cluster analysis is a fundamental challenge in trajectory mining. However, existing trajectory clustering algorithms are not well suited to large numbers of trajectories in a city road network because of inadequate distance measures between two trajectories. In this paper we propose a novel Dijkstra based Dynamic Time Warping (DTW) distance measure, trajDTW between two trajectories, which is suitable for large numbers of overlapping trajectories in a dense road network. We show the superiority of trajDTW over previously proposed distance measures *Dissimilarity with Length* (DSL) and Hausdorff distance for point sets using a few sample trajectories on a road network. We then show how our sampling based clustering algorithm clusiVAT can suggest the number of clusters, and identify and visualize the trajectories belonging to each cluster. We also detect anomalous trajectories in a given dataset using clusiVAT. Experimental results on a large scale T-Drive taxi trajectory dataset consisting of 43,405 trajectories on a road network having 100 nodes and 141 edges reveals the presence of 12 clusters having an average of 2,029 trajectories each. We compare the trajectory clusters obtained using the clusiVAT algorithm employing trajDTW distance measure with those obtained using the NETSCAN trajectory clustering method proposed in the literature. Furthermore, we identify the top 100 anomalies corresponding to a few vehicles taking unusually warped paths for their commute. These anomalous trajectories have their maximum traffic density in geographically distinct sections of the road network.

Keywords

Trajectory clustering, Road graph network, Scalable clustering, Distance measures

1. INTRODUCTION

In recent years, manufacturers are increasingly offering vehicles with geomatics services like *Global Positioning System* (GPS), which provide accurate measurements of the location of a vehicle, and are used for real time traffic information, car accident alarms, and efficient path planning. GPS location information from vehicles can be readily used to find the trajectory of a vehicle, which is a sequence of sampled locations and time stamps along the route of a moving object. A major challenge for road authorities and users is how to analyse the vast amount of trajectory data generated everyday. One of the most useful types of analysis in this context is trajectory clustering. Trajectory clustering identifies distinct groups of trajectories, such that there is a greater similarity in motion patterns within a group than between groups. On a broad level, trajectories can be classified as belonging to a Euclidean space or a road network space. Euclidean space trajectories can be represented as a series of (x, y, t) coordinates in 2-dimensional Euclidean space, but for most real world applications involving transportation, an object typically moves along the road segment and its trajectory can be represented as a 1-dimensional array of nodes or edges of the road network.

Literature is replete with various trajectory clustering approaches using different methods for trajectory representation, different distance measures between two trajectories, and various clustering techniques. Most of the approaches presented in the literature use synthetic datasets having small to medium numbers of trajectories, while a few use real trajectory datasets having a small number of trajectories for experimentation. The previously suggested trajectory clustering approaches are not suitable for clustering large numbers of overlapping vehicle trajectories in a city road network consisting of a large number of road segments. In this paper we aim to provide a scalable framework for clustering large numbers of vehicle trajectories in a dense city road network.

Our main contributions in this paper are as follows:

- We propose a novel Dijkstra based dynamic time warping distance measure trajDTW between two trajectories, which is suitable for large numbers of overlapping trajectories in a dense road network. We also provide a comparison of trajDTW with the DSL [21] and Hausdorff distance [12, 17] measures used in literature.
- We show how our sampling based clustering algorithm clusiVAT [13] can be used to suggest the number of clusters, and identify and visualize the trajectories belonging to each cluster. Based on this clustering we can also detect and visualize anomalous trajectories.
- We perform numerical evaluation on a large scale T-Drive taxi trajectory dataset [23, 24] consisting of 43,405 trajectories on a road network having 100 nodes and 141 edges, revealing the presence of 12 clusters, each

UrbComp'15, August 10, 2015, Sydney, Australia.

having an average of 2,029 trajectories. Furthermore, we identify the top 100 anomalous trajectories. We also compare the trajectory clusters obtained using our trajDTW distance measure based clusiVAT algorithm with that obtained using the NETSCAN [12] trajectory clustering method proposed in literature.

The rest of the paper is organized as follows. Section 2 provides a detailed literature review of road network trajectory clustering methods. Section 3 formally defines the problem and describes our novel Dijkstra based DTW distance measure between two trajectories in Section 3.1. The clusiVAT algorithm for clustering and anomaly detection is described in Section 4. The time complexity of our trajDTW distance measure and clusiVAT clustering algorithm is discussed in Section 5. Numerical experiments on a real life taxi trajectory dataset are provided in Section 6 before concluding in Section 7.

2. RELATED WORK

The representation of trajectories is the first step in trajectory clustering. A common approach is to use a Euclidean space representation, in which sample points on a trajectory are represented by the triplet (x, y, t), and the trajectory is considered to be linear between each pair of sample points. However, a major drawback of using this representation for objects travelling on a road network is that two locations that are close in Euclidean space may not be reachable over the road network if no road exists between the two locations. Won et al. [21] represent a trajectory as a list of segments, each of which has its own identifier and length, however the method to divide the road segment is not explicitly described and is driven by intuition. The authors in [20, 11] represent the trajectory of a moving object on a road network as a sequence of interest points, such as the intersection of two road segments or a notable building or symbol. Guo et al. [7] used topological information from graph based structures to represent and analyze trajectory data. The trajectory of a moving object appears as a sequence of symbols in [12, 15, 4, 18], where each symbol refers to one road section. In addition to the sequence of symbols, [8] also included information about the offset of the exact GPS location from the start junction of the road segment in the edge sequence. In this paper, we use the same representation of road networks and trajectories as proposed in [12, 15, 4, 18].

The next pre-clustering task for trajectory clustering is to find a measure of similarity or distance between a pair of trajectories. The first attempt to provide a spatiotemporal distance between two trajectories using network distance was proposed in [11]. It is a two step process consisting of a filtering phase to find spatial similarity on the road network, and a refinement phase for discovering similar trajectories based on temporal distance. Won et al. [21] used the length of the disjoint segments between two trajectories as a distance measure called *dissimilarity with length* (DSL). For two trajectories $T_i = [t_i^1, t_i^2, ..., t_i^l]$ and $T_j = [t_j^1, t_j^2, ..., t_j^m]$, of length l and m respectively, the DSL distance between them is given by

$$DSL(T_i, T_j) = \frac{L_d(T_i, T_j)}{L_s(T_i) + L_s(T_j)}, \tag{1}$$

where $L_d(T_i, T_j)$ is the sum of lengths of disjoint segments of T_i and T_j, and $L_s(T_i)$ and $L_s(T_j)$ are the sum of lengths of segments in T_i and T_j respectively. However this measure is not suitable for a city road network, which consists of a parallel and perpendicular grid of road segments. An additional network level similarity between two transitions as the difference of their traffic density values was introduced in [12, 17]. The Hausdorff distance for point sets, which considers the maximum mismatching level between two point sets, but does not consider any temporal information of the trajectory, is used as a trajectory distance measure in [20, 4, 15]. For the trajectories T_i and T_j as described above, the Hausdorff distance is given by

$$\text{Hausdorff}(T_i, T_j) = \max_{t_i^a \in T_i} \min_{t_j^b \in T_j} d(t_i^a, t_j^b), \tag{2}$$

where $d(t_i^a, t_j^b)$ is the Dijkstra distance between the edges t_i^a and t_j^b. In this paper, we propose a novel Dijkstra based DTW distance measure, trajDTW between two trajectories, which is suitable for large numbers of overlapping trajectories in dense road networks.

The third and final step in the trajectory clustering problem is the clustering method itself and the representation of the final clusters. Kharrat et al. [12] proposes a clustering scheme named *NETSCAN*, similar to the popular clustering algorithm DBSCAN, which first finds the most dense road sections, and merges them to form dense paths on the road network and later classifies the trajectories of moving objects according to these dense paths. DBSCAN was used as is in [19] for trajectory clustering, however both these methods, NETSCAN and DBSCAN, are not suitable for large numbers of trajectories in a city environment as computation of the distance matrix is time intensive. Traffic load was represented as the edges of a graph with locations as nodes in [7]. The weight given to such an edge is the number of trajectories passing through that edge. A graph partitioning method is applied to find natural regions (or community structures), where locations inside a region share more trajectories with each other than with locations in other regions. The work presented in [12, 19, 7] clusters the road segments in the road network based on their proximity and traffic load instead of clustering the trajectory of vehicles traversing the segments. Authors in [15] use an efficient agglomerative hierarchical clustering method to reduce distance computations without requiring an index structure, with little loss in the quality of clustering results. The *NEetwork Aware Trajectory* (NEAT) model was introduced in [8], which first identifies the most critical and interesting part of the trajectories and uses them as basic building blocks for clustering. We have used our clusiVAT algorithm [13] for the clustering task, which yields an estimate of the number of clusters present in the dataset, and is suitable for datasets having large numbers of trajectories.

Most of the work done in the area of trajectory clustering uses synthetic datasets having small to medium numbers of data points. For example, [12] uses a constrained moving object generator on the road network of San Joaquin Bay, producing 2,064 trajectories of moving objects. A synthetically created road network with 500 nodes was used in [4] to generate a trajectory dataset having an average track length of 25 nodes distributed in 5 clusters. Authors in [8, 21] uses public event-based simulators to generate thousands of mobility traces on the road networks of North West Atlanta, West San Jose, and Miami-Dade. A few papers use real trajectory datasets for experimentation. For example

[7] uses the truck dataset [5] which has GPS traces of trucks in Athens, Greece, for a total of 276 trajectories. A real-life trajectory dataset [1] containing 214 trajectories, having an average trajectory length range of 18 to 1486 GPS points was used in [15]. The number of trajectories in both of the real life datasets [5, 1] is fairly small. We have used the T-Drive Dataset consisting of the GPS trajectories of 10,357 taxis. In all, we have performed our experiment on 43,405 trajectories, having lengths in the range of 5 to 200 road segments. To the best of our knowledge, this is the first time a clustering task has been performed on such a large number of real life road network trajectories. A systematic survey of major research into trajectory data mining providing a panorama of the field as well as the scope of its research topics is given in [25]. Next we formally define the road network constrained trajectory clustering problem framework.

3. PROBLEM DEFINITION

We represent the road network as an undirected graph

$$G_{RN} = (V, E), \tag{3}$$

where V is a set of intersections of the road network, and E is a set of road segments, $R_i \in E$ such that $R_i = (r_{i_s}, r_{i_e})$, where $r_{i_s}, r_{i_e} \in V$ and there exists a road between r_{i_s} and r_{i_e}. The edge R_i is given a weight equal to the distance between r_{i_s} and r_{i_e}. In the case of a road segment being curved, additional vertices are placed in the road segment so that the parts of the road segment between the consecutive edges are approximately straight lines. This is done to approximate the length of a road segment as the straight line distance between the two ends with a high degree of accuracy. For such a road network, a trajectory T of length l (which varies between trajectories) is defined as $T = [t_1, t_2, ..., t_l]$, where $t_j \in E, 1 \leq j \leq l$, and t_j and t_{j+1} are connected. Next we describe in detail our Dijkstra based DTW distance measure between two trajectories.

3.1 Distance measure (trajDTW)

Most of the similarity measures proposed in the literature either use the number of overlapping road segments or the minimum/maximum distance to go from one trajectory to another, which is not a good distance measure for a large number of overlapping trajectories in a dense road network as used in this paper. We propose a novel distance measure called trajDTW between two trajectories using DTW distance, where the distance between two edges is given as Dijkstra's shortest path distance. This approach considers the temporal information in the trajectory, which was missing in [15, 4]. Dijkstra's shortest path distance between any two edges in the road network is given by the well known Dijkstra's Algorithm, which is a graph search algorithm that solves the single-source shortest path problem for a graph with non-negative edge path costs, producing a shortest path tree. Since the road network is static, we can pre-compute and store the distance matrix of all the edges in G_{RN}, which is a $|E| \times |E|$ matrix D_{all}, where $|E|$ is the number of edges in E, and whose elements are given by

$$D_{all_{i,j}} = Dijkstra(E_i, E_j), \tag{4}$$

where $Dijkstra$ is the well known Dijkstra's shortest path algorithm. The pseudocode for our distance measure trajDTW is given in Algorithm 1. It is a normal DTW algorithm with window parameter w, which is set to half the length of the shorter of the two trajectories. The distance measure between two road segments t_1^i and t_2^j is denoted as $D_{all_{t_1^i, t_2^j}}$.

Algorithm 1: trajDTW

Input : $T_1 = [t_1^1, t_1^2, ..., t_1^l]$ − Trajectory 1 consisting of l consecutive road segments
$T_2 = [t_2^1, t_2^2, ..., t_2^m]$ − Trajectory 2 consisting of m consecutive road segments
Output: $dist$ − distance between T_1 and T_2

D_{all} − $|E| \times |E|$ distance matrix of all the edges in G_{RN} (Pre-computed)
$w = \frac{1}{2} \times min(l, m)$ − window parameter

```
for i ← 1 to l + 1 do
    for j ← 1 to m + 1 do
        A_{i,j} = ∞
    end
end
A_{1,1} = 0
for i ← 1 to l do
    for j ← max(i − w, 1) to min(i + w, m) do
        cost = D_all_{t_1^i, t_2^j}
        A_{i+1,j+1} = cost + min(A_{i,j+1}, A_{i+1,j}, A_{i,j})
    end
end
dist = A_{l+1,m+1}
```

This is a balanced distance measure between the trajectories of different lengths for a city road network which has a grid of closely spaced parallel and perpendicular road segments. If we just use the number of common road segments, or a function of the length of the matching sub-trajectories and the length of the gap between them as a measure of similarity as proposed in [21, 20], we overestimate the distance between two trajectories running on two nearby parallel road segments. Whereas if we use the average minimum distance from any node in one track to any node in the other track as a track similarity function as in [4], or the longest distance that an adversary can force you to travel from one road segment to another as in [15], we underestimate the distance between two trajectories that have only a few edges in common, and for the most part are far away from each other, or for two trajectories that intersect at a common node and then diverge in both directions. Although DTW is sensitive to noise, in trajDTW, we first map the GPS traces of a vehicle to the edge sequence of the road network graph, hence removing the "noise" part which makes DTW a poor distance function.

As an example, consider the road network shown in Figure 1 with nodes (road segment intersections) represented by red dots and edges (road segments) represented by blue lines. Consider six trajectories as shown by six different colors and marked as A, B, C, D, E, and F respectively using the same color as that of the trajectory. The trajDTW (Algorithm 3.1), DSL (1) and Hausdorff (2) distance between selected pairs of trajectories is given in Table 1. Intuitively, trajDTW distance gives a fair estimate of the distance between trajectories. For example, trajectories A (green) and B (black) seem close to each other in the road network, but do not have a common edge or node, so would not be consid-

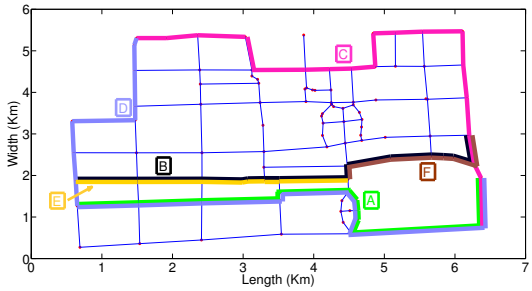


Figure 1: Trajectory distance measure example

Table 1: Distance matrix for the trajectories in Figure 1

Trajectories	Distance (trajDTW)	Distance (DSL)	Distance (Hausdorff)
A - B	1.36	1	2.43
A - C	3.51	0.88	4.76
A - D	1.12	0.23	4.35
B - C	2.77	0.92	4.10
B - E	0.96	0.27	3.01
B - F	0.94	0.40	3.58
E - F	3.14	1	3.58

ered close as per the DSL distance measure. In contrast, trajectory C (magenta), which for most of the time is at a large distance from trajectories A and B, but has one common edge with both of them, would be considered close to them as per the DSL distance measure (DSL(A-B)=1 is greater than DSL(A-C)=0.88 and DSL(B-C)=0.92). Whereas the trajDTW(A-B)=1.36, trajDTW(B-C)=2.77, and trajDTW(A-C)=3.51 seems reasonable from the trajectory plot shown in Figure 1. Trajectories E (yellow) and F (brown), which belong to adjacent but non overlapping parts of the same long road segment have a maximum value of DSL distance (DSL(E-F)=1), whereas they are close to each other in the road network. Trajectories A (green) and D (violet) for the most part have common paths but diverge at the end, so should be considered as close to each other, which justifies trajDTW(A-D)=1.12, but has a high Hausdorff distance (Hausdorff(A-D)=4.35) because of the divergence at the end. Similarly, trajectories E and F are sub-trajectories of B, so should be considered close to B. While trajDTW(B-E)=0.96 and trajDTW (B-F)=0.94 supports this observation, their Hausdorff distance is high (3.01 and 3.58 respectively). For some classes of trajectories, DSL and Hausdorff distance overestimate the actual distance, whereas trajDTW provides a balanced distance measure. Next we describe our clusiVAT clustering algorithm in detail.

4. CLUSIVAT ALGORITHM

Reordered dissimilarity images (RDIs) have been used for visual representation of the structure in unlabeled dissimilarity data since the 19^{th} century. The RDI highlights the potential cluster structure of the data by the set of dark blocks along its diagonal, where each block represents a different cluster. The *visual assessment of clustering tendency* (VAT) algorithm [2] reorders the input distance matrix D to obtain D^* using a modified Prim's algorithm. The image $I(D^*)$, when displayed as a gray-scale image, shows possible clusters as dark blocks along the diagonal.

Although VAT can provide a useful estimate of the number of clusters in a dataset, if the clusters are close to each other, the VAT image can be inconclusive. A new algorithm named *improved VAT* (iVAT) was proposed in [10] to alleviate this problem. iVAT provides better images by replacing input distances in the distance matrix $D = [d_{ij}]$ by geodesic distances $D' = [d'_{ij}]$, given by

$$d'_{ij} = \min_{p \in P_{ij}} \max_{1 \le h \le |p|} D_{p[h]p[h+1]}, \quad (5)$$

where P_{ij} is the set of all paths from trajectory i (T_i) to trajectory j (T_j) in the VAT generated minimum spanning tree (MST).

VAT and iVAT suffer from size limitations as they have a space and time complexity of $O(n^2)$. To overcome this limitation, *scalable-VAT* (sVAT) was introduced in [9], which works by sampling the big dataset and then constructing a VAT or iVAT image of the sample. sVAT finds a small D_n distance matrix (having size $n \times n$) of an n sized subset of the trajectory dataset, $\mathbf{T} = \{T_1, T_2, ..., T_N\}$, where N is large and n is a "VAT-sized" fraction of N. siVAT is just like sVAT, except that it uses iVAT after the sampling step.

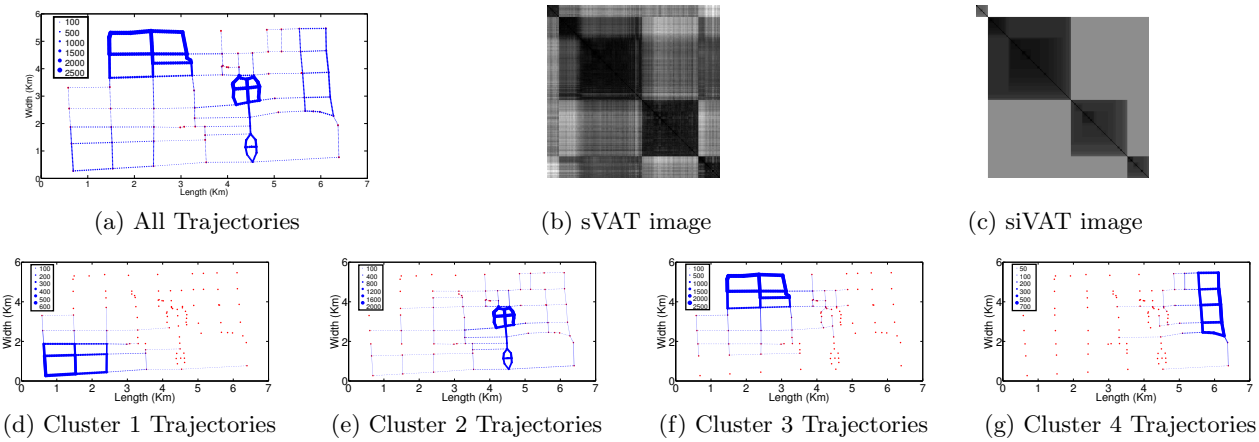
The MST built using Prim's algorithm in VAT and iVAT provides an array representing the edges of the MST, which is used in the reordering operation. Let us assume that the iVAT image suggests the presence of k clusters in the dataset $\mathbf{T}$. Having this estimate, we cut the $k-1$ largest edges in the MST, resulting in k connected subtrees (the clusters). The essential step in clusiVAT [13] is to extend this k-partition of D_n non-iteratively to the unlabeled objects in $\mathbf{T}$ using the nearest (trajectory) prototype rule (**NPR**). Pseudocode for our VAT, iVAT, siVAT, and clusiVAT algorithms are well documented in [2, 10, 9, 13], and hence are not reproduced here for brevity.

To illustrate clusiVAT consider Figure 2, which shows the clustering experiment performed on a synthetically generated dataset of 10,000 trajectories on a road segment consisting of 100 intersections and 141 road segments. View (a) shows the trajectory density map of all the trajectories, showing the presence of a large number of trajectories in 3 corners and the center of the road network. Its sVAT image for $k' = 10$ and $n = 500$ (siVAT parameters (details in [9, 13])) is shown in view (b), which indicates the presence of 4 clusters by 4 dark blocks along the diagonal. These dark blocks are much clearer in view (c), which is the siVAT image. Finally views (d-g) show different clusters of trajectories obtained by cutting the 3 largest edges in the MST and using clusiVAT. As expected, clusiVAT is able to extract the 4 clusters correctly.

If the dataset is complex, and the clusters are intermixed with each other and contain a number of anomalies, cutting the $k-1$ largest edges of the MST to obtain k clusters is not always a good strategy as the anomalies, which are at a large distance from the normal clusters, would constitute all the $k-1$ largest edges of the MST. A more useful approach in such a scenario is to manually select the dark blocks along the diagonal, find the sample trajectories representing this dark block and use **NPR** to find those trajectories in the dataset $\mathbf{T}$, whose nearest trajectory belongs to the sample trajectories of the cluster.

4.1 Anomaly Detection using clusiVAT

Anomaly detection can be regarded as a special case of data clustering in which clusters that are too far from the



(a) All Trajectories (b) sVAT image (c) siVAT image

(d) Cluster 1 Trajectories (e) Cluster 2 Trajectories (f) Cluster 3 Trajectories (g) Cluster 4 Trajectories

Figure 2: All trajectories, sVAT, siVAT and clustered trajectories for the synthetic trajectory dataset having 10,000 trajectories divided into 4 clusters

main clusters and have too few data points, are regarded as anomalies. We use this concept for anomaly detection using the clusiVAT algorithm. To determine the top p anomalies among the trajectories in $\mathbf{T}$, we find the highest p values of the MST cut magnitude d. We then find the trajectories that are nearer to the anomalous trajectories than any other trajectory using clusiVAT. If the number of such trajectories is small, they are branded as anomalous. Next we discuss the time complexity of the trajDTW distance measure and clusiVAT algorithm.

5. TIME COMPLEXITY

In this section, we discuss the time complexity of our proposed trajDTW distance measure and clusiVAT algorithm. trajDTW uses Dijkstra's shortest path distance in the normal DTW algorithm. The time complexity of Dijkstra's algorithm depends on the number of nodes and edges in the network. Its best average case time complexity is obtained when using binary heaps for storing the road network graph and is of the order of $O(|E| + |V|log(\frac{|E|}{|V|} + |V|))$ [14]. For two trajectories of length l and m, the time complexity of a standard DTW algorithm is $O(l \times m)$, but there exist approximate DTW algorithms like fastDTW [16], whose time complexity is linear in the average length of the trajectories, $O(l + m)$.

For the trajectory dataset T containing N trajectories, the first step in clusiVAT is the selection of k' distinguished trajectories which are at a maximum distance from each other. This step divides the entire dataset into k' almost equally sized partitions. This step has time complexity linear in k'. The next step in clusiVAT is to randomly select objects from the k' partitions to get a total of n samples. These n samples, which are just a small fraction of N, retain the approximate geometry of the dataset. In the next step, VAT is applied to the n samples, which (including construction of D_n from T) has a time complexity of $O(n^2)$. So the $N \times N$ distance matrix for the big dataset (D_N) is never needed, but just the $n \times n$ distance matrix of the sampled dataset (D_n).

6. NUMERICAL EXPERIMENTS

The numerical experiments to cluster road network trajectories using our clusiVAT clustering algorithm are performed on the T-Drive Taxi Trajectory dataset [23, 24], which contains the GPS traces of 10,357 taxis during the period of Feb. 2 to Feb. 8, 2008 within Beijing. The total number of GPS points in this dataset is about 15 million and the total distance of the trajectories is 9 million kilometers. For our experiment we have taken a subset of this dataset, which contains trajectories in a small road network in the center of the Beijing city as shown in Figure 3. This road network

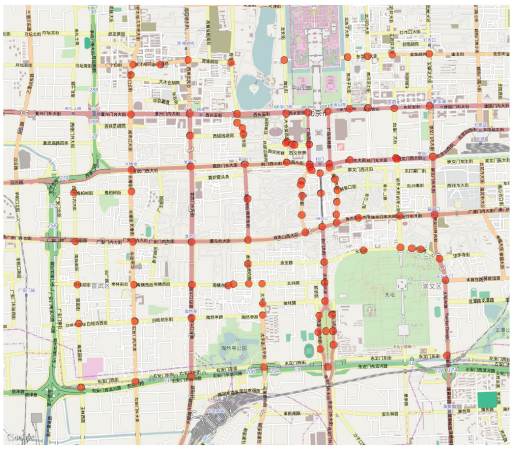


Figure 3: Road network in the center of Beijing city, which is used for the clustering experiment

consists of 100 nodes as shown by red dots in Figure 3 and 141 road segments (edges). The average sampling interval of the GPS points is about 177 seconds with a distance of about 623 meters, which is quite large for a city traffic environment as the length of many road segments is smaller than the average sampling distance. All programs are written in MATLAB 2012. The computational platform is OS: Windows 7 (64 bit); processor: Intel(R) Core(TM) i7-2600 @3.40GHz; RAM: 8GB.

As a pre-processing step to obtain the trajectories as a sequence of road segments, each of which has a common node with its former and latter road segment, we first map each GPS point to its nearest road segment (commonly known as the *Map Matching problem*). A few approaches presented in the literature for the purpose of map matching include [6, 3, 22]. The consecutive duplicate road segments in a trajectory are removed and if the two consecutive road segments do not have a common node, Dijkstra's algorithm is used to find and insert the minimum length road segment sequence between the two non-adjacent road segments. As an example, for the road network shown in Figure 4, the GPS trace of a vehicle is shown by green dots, which are mapped to road segments and interpolated when necessary to give the magenta trajectory.

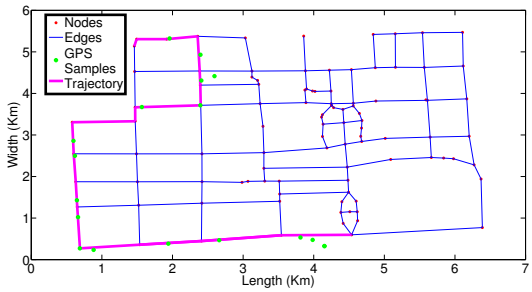


Figure 4: Extracted road network trajectories from GPS trace

After the preprocessing step, we are left with the trajectory dataset $\mathbf{T} = \{T_1, T_2, ..., T_N\}$ having N =43,405 trajectories, whose lengths lie in the range of 5 to 200 road segments and have an average of 14 road segments. Although 43,405 trajectories over a graph of 100 nodes is not exactly "big data", when taken in the context of the previous trajectory clustering literature which have used a few hundred trajectories from real life datasets, 43,405 trajectories having lengths in the range of 5 to 200 road segments, over a graph of 100 nodes is quite big. Using the value of the parameters $k' = 30$ and $n = 500$, the clusiVAT algorithm is applied to this trajectory dataset giving the clusiVAT image as shown in Figure 5. Since there are many overlapping trajectories in the dataset, the clusters are not clearly separated from each other, hence the dark blocks along the clusiVAT image are not very clear and are intermixed with each other. The top image in Figure 5 seems to have two primary dark blocks, and embedded in them, a fine structure that has many more. The zoomed images of the two dark blocks shown in the lower part of Figure 5 reveal the presence of 7 and 5 dark sub-blocks representing a total of 12 imbedded subclusters. These dark blocks are marked by red rectangles for clarity. To obtain the trajectories belonging to these clusters, we first find the indices of the trajectories belonging to each of the dark blocks using the siVAT algorithm. For the remaining trajectories, we find their cluster membership using **NPR** as described in Section 4. Figure 6 shows the trajectory density map of the trajectories belonging to the 12 different clusters found using the clusiVAT algorithm. The number of trajectories belonging to clusters 1 to 12 are 1,655, 3,976, 3,864, 1,359, 2,044, 1,105, 2,115, 917, 3,140, 1,864, 1,674, and 633 respectively.

For an unlabeled dataset such as the T-Drive Taxi Tra-

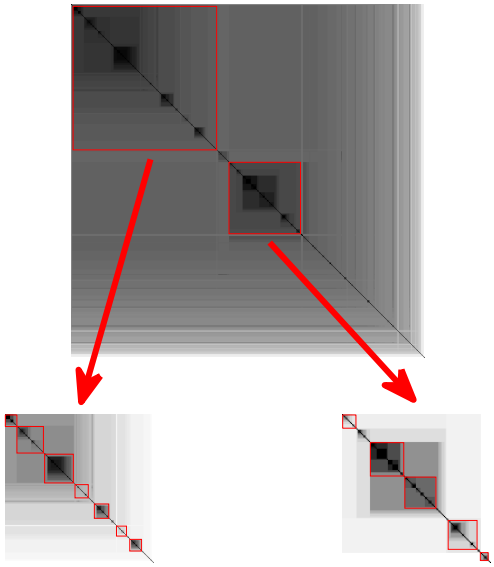
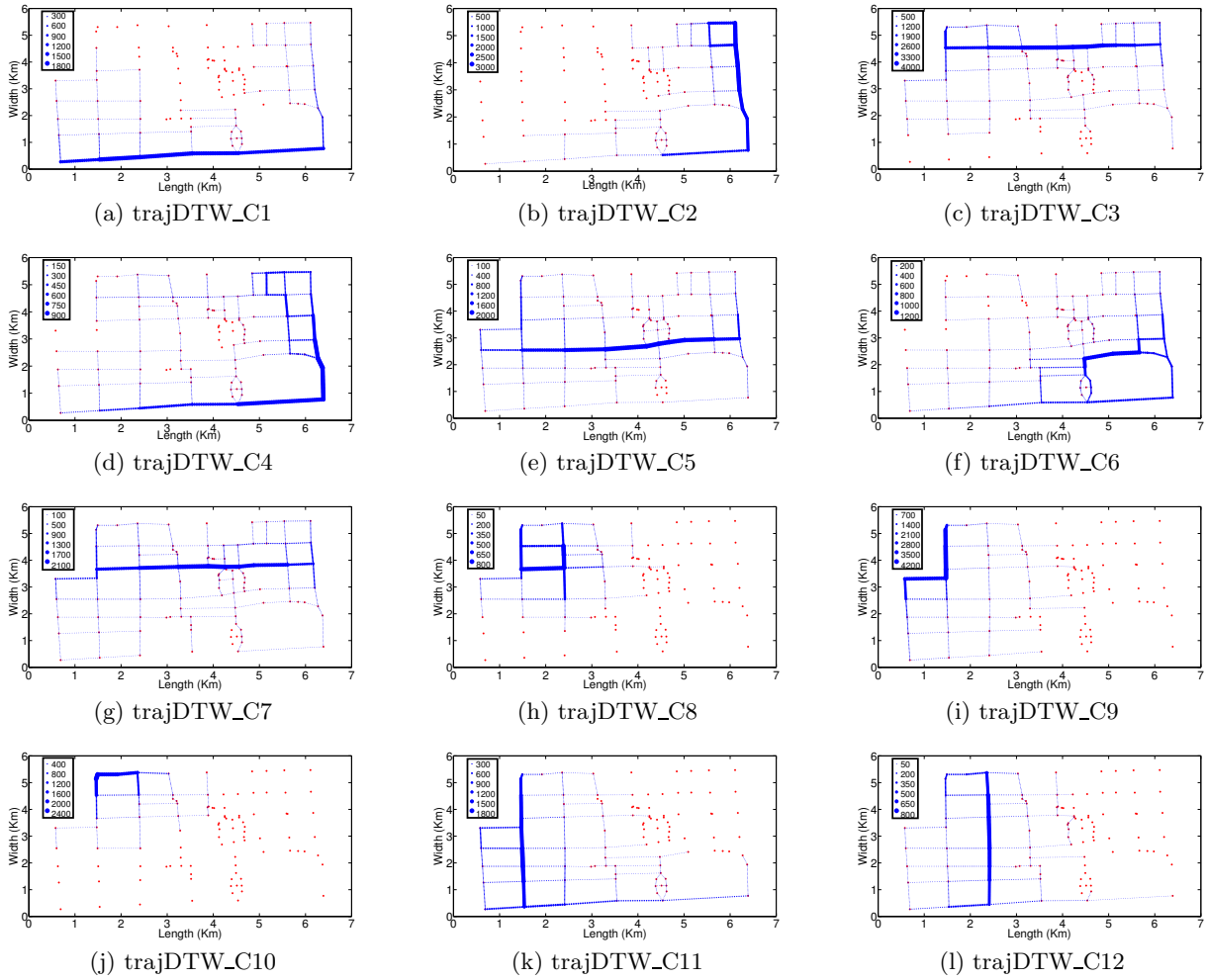


Figure 5: clusiVAT image of the trajectory dataset

jectory dataset, there is no quantitative measure to check how good or bad the clusters are. We believe the best way to check this is using the images of the various trajectory clusters, which clearly shows the trajectories of vehicles on different road segments clearly partitioned among various clusters. To illustrate the effectiveness of our method to discover cluster structure in the trajectory dataset, we compare the results obtained above with those obtained using the NETSCAN clustering method proposed in [12]. The first step in NETSCAN is to find the network paths that are the densest in terms of moving objects transiting on them, using a transition matrix relative to the road network. This step uses two user defined parameters: density threshold (α) and similarity threshold (ϵ). α is the minimum transition density, and ϵ is the maximum density difference between neighbouring road segments to be considered as dense paths. We experimented with various values of thresholds before setting them to $\alpha = 1,000$ and $\epsilon = 800$, which gives the maximum number of dense paths having at least four connected road segments. Figure 7(a) shows seven dense paths (by seven different colors) obtained in step 1 of the NETSCAN algorithm. In the next step, NETSCAN groups the trajectories according to their similarity to each dense path generated in the first step. The similarity measure used in this step compares two trajectories where one is the reference. This measure reflects the resemblance to an object and it is not symmetric. The similarity is computed as the ratio between the common length among a trajectory and the reference and the length of the reference trajectory

$$Sim\_traj = \frac{Lenght(common\_part)}{Lenght(ref\_traj)}. \quad (6)$$

For each dense path, NETSCAN computes the similarity with each trajectory. If the similarity is above another user defined threshold value (σ), then the trajectory is kept in



(a) trajDTW_C1 (b) trajDTW_C2 (c) trajDTW_C3

(d) trajDTW_C4 (e) trajDTW_C5 (f) trajDTW_C6

(g) trajDTW_C7 (h) trajDTW_C8 (i) trajDTW_C9

(j) trajDTW_C10 (k) trajDTW_C11 (l) trajDTW_C12

Figure 6: Trajectories belonging to different clusters for the T-Drive Taxi Trajectory dataset

the cluster. For this step, we set $\sigma = 0.8$ as suggested in [12]. Figure 7(b-f) shows the trajectories belonging to the different clusters obtained using NETSCAN.

The trajectory clusters obtained using the trajDTW distance measure based clusiVAT algorithm are better in comparison to those obtained using NETSCAN for the following reasons:

1. The clusiVAT clusters in Figure 6 have all the trajectories confined to a small part of the road network as evident by many unfilled edges, whereas all the trajectory clusters obtained by NETSCAN (Figure 7) are spread over the entire road network (all the edges are filled).

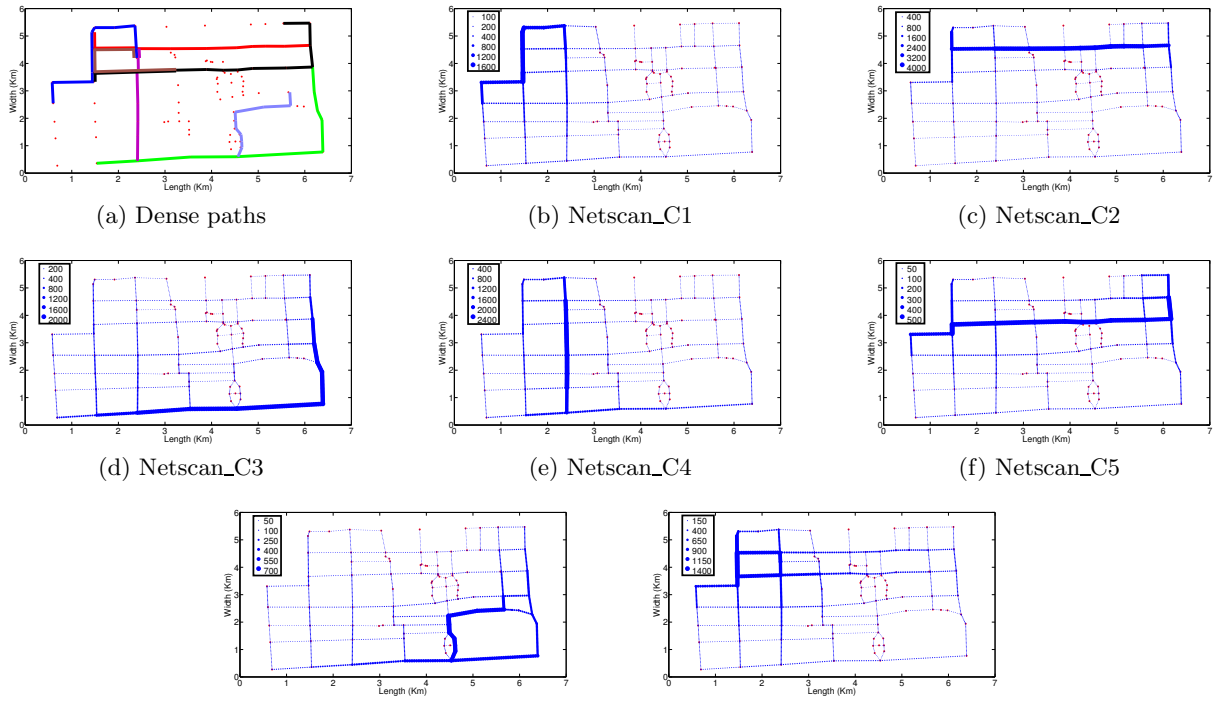
2. The number of clusters obtained using clusiVAT are more than that obtained using NETSCAN, and the major road sequence highlighted by dense paths of NETSCAN belong to separate clusters obtained using clusiVAT.

Table 2 compares the trajectory clusters obtained using the trajDTW based clusiVAT and NETSCAN algorithms. NETSCAN is able to identify some of the clusters correctly, for example, trajDTW clusters 3, 4, 12, 7, 6, and 8 are identified correctly as NETSCAN clusters 2, 3, 4, 5, 6, and 7 respectively, whereas it completely missed complex trajectory clusters such as trajDTW clusters 1, 2, 5, and 11. NETSCAN combines trajDTW_C9 and trajDTW_C10 to form NETSCAN_C1 as the major road segments of these two clusters are close to each other.

Table 2: Summary of clusters obtained using trajDTW based clusiVAT algorithm and NETSCAN

trajDTW-clusiVAT	**NETSCAN**
trajDTW_C3	NETSCAN_C2
trajDTW_C4	NETSCAN_C3
trajDTW_C12	NETSCAN_C4
trajDTW_C7	NETSCAN_C5
trajDTW_C6	NETSCAN_C6
trajDTW_C9 trajDTW_C10	NETSCAN_C1
trajDTW_C8	NETSCAN_C7
trajDTW_C1, trajDTW_C2, trajDTW_C5, trajDTW_C11	—

We have used the clusiVAT algorithm to find the anomalous trajectories. Figure 8 shows the top 100 anomalous



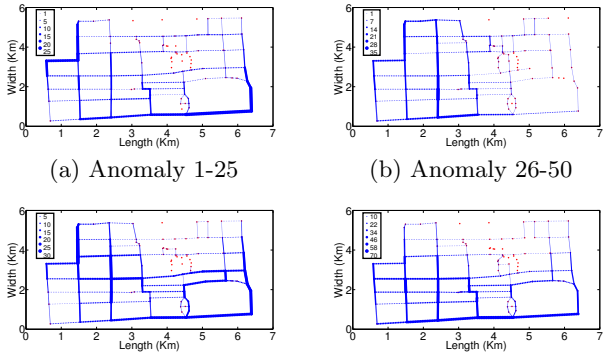
(a) Dense paths (b) Netscan_C1 (c) Netscan_C2

(d) Netscan_C3 (e) Netscan_C4 (f) Netscan_C5

(g) Netscan_C6 (h) Netscan_C7

Figure 7: (a) Dense paths and (b-f) Trajectories belonging to different clusters for the T-Drive Taxi Trajectory dataset using NETSCAN clustering method proposed in [12]

trajectories in the dataset. These anomalous trajectories represent a few vehicles taking an unusually warped trajectory for their commute, whose maximum traffic density is in geographically distinct sections of the road network.



(a) Anomaly 1-25 (b) Anomaly 26-50

(c) Anomaly 51-75 (d) Anomaly 76-100

Figure 8: Top 100 anomalous trajectories for the T-Drive Taxi Trajectory dataset

7. CONCLUSION

We have presented a novel Dijkstra based DTW distance measure between two trajectories, which is suitable for a large numbers of overlapping trajectories in a dense road network. We applied our new efficient clustering algorithm clusiVAT for a road network containing a large number of vehicle trajectories. We performed our experiments on 43,405 trajectories obtained from the real life T-Drive Taxi Trajectory dataset, which contains the GPS traces of taxis within Beijing. We first interpolated GPS samples in the T-Drive dataset to accurately establish topological relationships among trajectories and locations. Using the clusiVAT algorithm we are able to suggest the possible number of trajectory clusters in the dataset and visualize them. We compare the clusters obtained using our novel trajDTW distance measure based clusiVAT clustering with that obtained using the NETSCAN algorithm proposed in the literature. We conclude that the latter can not identify all the clusters present in the dataset. We are also able to find the top 100 anomalous taxi trajectories in the dataset. This analysis can effectively facilitate the understanding of spatial patterns in trajectories and is of great significance for decision-makers to understand road traffic conditions and to propose metro bus corridors and light rail systems for better public transport.

8. ACKNOWLEDGMENTS

We thank the support from UoM ECR; ARC (LP120100529, LE120100129, LP130101038); EU FP7 SocIoTal; and National ICT Australia (NICTA).

9. REFERENCES

[1] Real Trajectory Data. *http://www.cs.uic.edu/~boxu/mp2p/gps_data.html.*

[2] J. Bezdek and R. Hathaway. VAT: A tool for visual assessment of (cluster) tendency. *International Joint Conference on Neural Networks (IJCNN)*, pages 2225–2230, 2002.

[3] W. Chen, M. Yu, Z. Li, and Y. Chen. Integrated vehicle navigation system for urban applications. In *International Conference on Global Navigation Satellite Systems (GNSS)*, pages 15–22, 2003.

[4] M. R. Evans, D. Oliver, S. Shekhar, and F. Harvey. Fast and exact network trajectory similarity computation: A case-study on bicycle corridor planning. In *Proceedings of the 2nd ACM SIGKDD International Workshop on Urban Computing*, pages 9:1–9:8, New York, NY, USA, 2013. ACM.

[5] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In *13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 330–339, New York, NY, USA, 2007. ACM.

[6] J. Greenfeld. Matching gps observations to locations on a digital map. In *81st Annual Meeting of the Transportaion Research Board*, pages 576–582, 2002.

[7] D. Guo, S. Liu, and H. Jin. A graph-based approach to vehicle trajectory analysis. *Journal of Location Based Services*, 4(3-4):183–199, Sept. 2010.

[8] B. Han, L. Liu, and E. Omiecinski. Road-network aware trajectory clustering: Integrating locality, flow, and density. *IEEE Transactions on Mobile Computing*, 14(2):416–429, Feb 2015.

[9] R. Hathaway, J. Bezdek, and J. Huband. Scalable visual assessment of cluster tendency for large data sets. *Pattern Recognition*, 39:1315–1324, 2006.

[10] T. Havens and J. Bezdek. An efficient formulation of the improved visual assessment of cluster tendency (iVAT) algorithm. *IEEE TKDE*, 24(5):813–822, 2012.

[11] J. R. Hwang, H. Y. Kang, and K. J. Li. Spatio-temporal similarity analysis between trajectories on road networks. In *24th International Conference on Perspectives in Conceptual Modeling*, pages 280–289, Berlin, Heidelberg, 2005. Springer-Verlag.

[12] A. Kharrat, I. Popa, K. Zeitouni, and S. Faiz. Clustering algorithm for network constraint trajectories. In *Headway in Spatial Data Handling*, pages 631–647. Springer Berlin Heidelberg, 2008.

[13] D. Kumar, M. Palaniswami, S. Rajasegarar, C. Leckie, J. Bezdek, and T. Havens. clusiVAT: A mixed visual/numerical clustering algorithm for big data. In *IEEE International Conference on Big Data*, pages 112–117, Oct 2013.

[14] K. Mehlhorn and P. Sanders. *Algorithms and Data Structures: The Basic Toolbox*. Springer, Berlin, 2008.

[15] G. P. Roh and S. W. Hwang. NNCluster: An efficient clustering algorithm for road network trajectories. In *Database Systems for Advanced Applications*, volume 5982, pages 47–61. Springer Berlin Heidelberg, 2010.

[16] S. Salvador and P. Chan. Toward accurate dynamic time warping in linear time and space. *Intelligent Data Analysis*, 11(5):561–580, Oct. 2007.

[17] I. Sandu Popa, K. Zeitouni, V. Oria, and A. Kharrat. Spatio-temporal compression of trajectories in road networks. *GeoInformatica*, 19(1):117–145, 2015.

[18] S. Song, D. Kwak, Y. Kwak, K. Bok, and D. Ko. Segmentation based trajectory clustering in road network with location sensing technology. *Sensor Letters*, 11(9):1779–1782, 2013.

[19] X. Wang, X. Ma, and E. Grimson. Unsupervised activity perception by hierarchical bayesian models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, June 2007.

[20] Y. Wang, Q. Han, and H. Pan. A clustering scheme for trajectories in road networks. In *3rd International Conference on Teaching and Computational Science (WTCS'09)*, volume 117, pages 11–18. Springer Berlin Heidelberg, 2012.

[21] J. I. Won, S. W. Kim, J. H. Baek, and J. Lee. Trajectory clustering in road network environment. In *IEEE Symposium on Computational Intelligence and Data Mining, CIDM '09*, pages 299–305, March 2009.

[22] H. Yin and O. Wolfson. A weight-based map matching method in moving objects databases. In *Proceedings of the 16th International Conference on Scientific and Statistical Database Management*, SSDBM '04, pages 437–, Washington, DC, USA, 2004. IEEE Computer Society.

[23] J. Yuan, Y. Zheng, X. Xie, and G. Sun. Driving with knowledge from the physical world. In *17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 316–324, New York, NY, USA, 2011. ACM.

[24] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang. T-drive: Driving directions based on taxi trajectories. In *18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '10, pages 99–108, New York, NY, USA, 2010. ACM.

[25] Y. Zheng. Trajectory data mining: An overview. *ACM Trans. Intell. Syst. Technol.*, 6(3):29:1–29:41, May 2015.