Fast and Scalable Big Data Trajectory Clustering for Understanding Urban Mobility

Dheeraj Kumar, Huayu Wu, Sutharshan Rajasegarar, Christopher Leckie, Shonali Krishnaswamy, and Marimuthu Palaniswami, *Fellow, IEEE*

***Abstract*— Clustering of large-scale vehicle trajectories is an important aspect for understanding urban traffic patterns, particularly for optimizing public transport routes and frequencies and improving the decisions made by authorities. Existing trajectory clustering schemes are not well suited to large numbers of trajectories in dense city road networks due to the difficulty in finding a representative distance measure between trajectories that can scale to very large datasets. In this paper, 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 as found in major cities around the world. We also propose a novel fast-clusiVAT algorithm that can suggest the number of clusters in a trajectory dataset and identify and visualize the trajectories belonging to each cluster. We conduct experiments on a large-scale taxi trajectory dataset consisting of 3.28 million trajectories obtained from the GPS traces of 15 061 taxis within Singapore over a period of one month. Our analysis finds 13 trajectory clusters spanning the major expressways of Singapore, each of which can be further divided into two sub-clusters based on the travel direction. For each cluster, we provide a time-based distribution of trajectories to yield insights into how urban mobility patterns change with the time of day. We compare the trajectory clusters obtained using our approach with those obtained using popular general and trajectory specific clustering frameworks: DBSCAN, OPTICS, NETSCAN, and NEAT. We demonstrate that the clusters obtained using our novel fast-clusiVAT framework are better than those obtained using other clustering schemes, evaluated based on two internal cluster validity measures: Dunn's and Silhouette indices. Moreover, our fast-clusiVAT algorithm achieves significant speedup over a comparable approach without loss of cluster quality.**

***Index Terms*— Fast and scalable trajectory clustering, road network graphs, scalable big data clustering, trajectory distance measures.**

I. Introduction

DATA-DRIVEN analyses of vehicle trajectories in traffic is an important component of intelligent transport systems (ITSs) as it helps in understanding urban traffic patterns, improving the efficiency of public transport networks, providing a better experience for commuters, and making informed decisions by authorities. With the increased popularity of *Global Positioning System* (GPS)-enabled vehicles, huge volumes of vehicle trajectory data are routinely generated. If analyzed effectively, this trajectory data can reveal valuable insights into the urban mobility patterns. One of the most useful types of analysis in this context is trajectory clustering, which identifies distinct groups of trajectories, such that there is greater similarity of trajectories within each group than between groups. The resulting trajectory clusters can provide useful insights into traffic flow patterns, trip planning, predicting passenger demand, traffic monitoring, and location-based services [1]–[3]. A potential application of trajectory clustering is to find major passenger movement patterns in order to better plan public transport routes and frequencies [1], [4]. For example, the routes corresponding to major clusters of passenger movements should be considered as candidates for new public transportation services, such as buses or trains to serve large numbers of commuters. This analysis helps in designing bus routes that can service a high density of commuters while reducing the number of transit points. Another potential application of trajectory clustering is location-based services, where advertising material and special offers can be directed to the commuters most likely to be in the vicinity of a business based on their travel trajectory.

Various clustering approaches using different methods for trajectory representation and different distance measures between trajectories have been proposed in the literature. A detailed survey of the research in this domain can be found in [5]. Road network constrained trajectory clustering approaches can be divided into two broad categories. In the first category, trajectories are clustered using traditional clustering approaches such as k-means and DBSCAN with specially designed distance measures for trajectories [6]–[10]. However, these algorithms are not scalable to large numbers of trajectories as computing the distance matrix is time and

Manuscript received December 11, 2017; revised June 2, 2018; accepted July 3, 2018. This work was supported in part by EU FP7 SocIoTal and in part by H2020-ICT-2014-1 OrganiCity. The Associate Editor for this paper was R. Trasarti. *(Corresponding author: Dheeraj Kumar.)*

D. Kumar was with the Electrical and Electronic Engineering Department, The University of Melbourne, Melbourne, VIC 3010, Australia, and also with the Institute for Infocomm Research, A*STAR, Singapore 138632. He is now with the Lyles School of Civil Engineering, Purdue University, West Lafayette, IN 47907 USA (e-mail: kumar299@purdue.edu).

H. Wu was with the Data Analytics Department, Institute for Infocomm Research, A*STAR, Singapore 138632. He is now with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: wu_huayu@ntu.edu.sg).

S. Rajasegarar is with the School of Information Technology, Deakin University, Geelong, VIC 3125, Australia (e-mail: srajas@deakin.edu.au).

C. Leckie is with the Department of Computing and Information Systems, The University of Melbourne, Melbourne, VIC 3010, Australia (e-mail: caleckie@unimelb.edu.au).

S. Krishnaswamy was with the Data Analytics Department, Institute for Infocomm Research, A*STAR, Singapore 138632. She is now with the Data Science Research Institute, Swinburne University of Technology, Hawthorn, VIC 3122, Australia (e-mail: skrishnaswamy@swinburne.edu.au).

M. Palaniswami is with the Electrical and Electronic Engineering Department, The University of Melbourne, Melbourne, VIC 3010, Australia (e-mail: palani@unimelb.edu.au).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2018.2854775

space prohibitive. The second category of algorithms clusters road segment vehicle frequencies based on their density and flow [4], [11], [12]. These approaches, though scalable and automatic, produce clusters having a high intra-cluster variance that spans a large portion of the road network, and hence are not suitable for the problem described in this paper, i.e., analyzing passenger taxi trips to understand spatio-temporal urban mobility.

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 as typically found in major cities around the world. We also propose the Fast-clusiVAT algorithm, a novel scalable and fast version of our two-stage clusiVAT algorithm [13]–[15] to suggest the number of clusters in a dataset and identify and visualize the trajectories belonging to each cluster in a fraction of the time taken by clusiVAT. While the clusiVAT algorithm is fast and scalable for Euclidean distances, which can be calculated as a batch operation, it suffers from high computational overhead when computing trajDTW distances, as these need to be computed in a pair-wise manner as required, and cannot be computed in a batch. We use our proposed algorithm on a real-life dataset to understand the mobility patterns of taxi passengers. In particular, we aim to understand urban mobility patterns by identifying the most popular routes taken by taxi passengers when commuting in Singapore, as well as the time distribution of trips for each such route. We analyze the mobility patterns by clustering 3.28 million passenger taxi trips extracted from a large-scale Singapore taxi dataset consisting of the GPS traces of 15,061 taxis during a one month period. We then compare our results with a variety of popular general-purpose and trajectory specific clustering frameworks: DBSCAN [16], OPTICS [17], NETSCAN [11], and NEAT [4]. 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 propose the Fast-clusiVAT algorithm, which is a novel fast version of the two-stage clusiVAT algorithm, to suggest the number of clusters, and identify and visualize the trajectories belonging to each cluster
- We perform an empirical evaluation on a large scale taxi trajectory dataset consisting of 370 million GPS traces and 3.28 million passenger trips from 15,061 taxis during a one month period within Singapore. 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.
- We analyze the distribution of taxi trips over time to gain insights about how traffic flows change with time, thus suggesting how the frequency of public transport should vary with time of day on different routes.

The rest of the paper is organized as follows. Section II provides a literature review of the relevant work. Section III describes the limitations of two popular flow based trajectory clustering frameworks (NETSCAN and NEAT) for clustering dense road network trajectories in order to optimize public transport routes and frequencies in large cities. Section IV formally defines the problem, and describes our novel Dijkstra-based DTW distance measure between trajectories in Section IV-A. Our novel Fast-clusiVAT algorithm for trajectory clustering is described in Section V, and its time complexity is discussed in Section VI. The Singapore taxi trajectory dataset and the framework to extract trips from raw GPS samples are described in Section VII. Numerical experiments on the Singapore taxi trajectory dataset are provided in Section VIII before concluding in Section IX.

II. Related Work

The representation of trajectories is the first step in trajectory clustering. For vehicles moving along road segments, their trajectory can be represented as a 1-dimensional array of nodes or edges in the road network [4], [8], [11], [18]. The $(Latitude, Longitude)$ coordinate pair obtained from a GPS sensor can be associated with the corresponding road network edge using one of the many popular map-matching techniques [19], [20]. Flow and density based trajectory clustering schemes such as NETSCAN [11] and *NEetwork Aware Trajectory Clustering* (NEAT) [4] first summarize the trajectories into a road network edge density or edge transition matrix. Clustering is then performed on road segments based on their traffic flow to create a set of consecutive road segments having continuity of traffic density and flow. NETSCAN is similar to the popular clustering algorithm DBSCAN, it first finds the densest road sections and merges them to form dense paths on the road network. It then classifies the trajectories of moving objects according to these dense paths. The NEAT model was introduced in [4], which first identifies the most critical and interesting part of the trajectories and uses them as basic building blocks for clustering. The work presented in [4] and [11] clusters the road segments in the road network based on their proximity and traffic load, instead of clustering the trajectories of vehicles traversing the segments. Although these approaches are scalable, they produce clusters having a large intra-class variance that span a large portion of the road network and may not represent actual traffic flow. Hence, they are not suitable for people movement analysis. Recently, Hong *et al.* [12] proposed a flow-based spatiotemporal trajectory clustering method called ST-TOPOSCAN using time-dependent shortest-path distance measurements to discover the shared sub-paths among trajectories and construct clusters. This approach, being flow and density based, has similar shortcomings to [4] and [11].

For trajectory clustering approaches that use traditional clustering algorithms, the next pre-clustering task is to find a measure of similarity or distance between a pair of trajectories. Besse *et al.* [21] provided a comprehensive review of the different distances used in the literature to compare trajectories, and later proposed a new distance called *symmetrized segment-path distance* (SSPD). However, SSPD does not uses the underlying road network for distance computation, and hence is not relevant for the problem of clustering road network constrained trajectories. Two popular distance measures

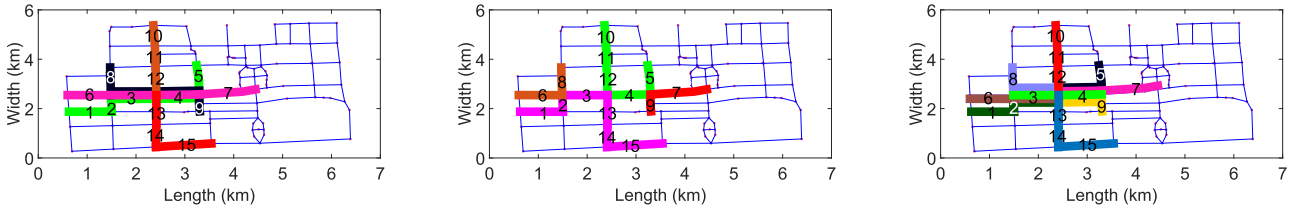


Fig. 1. Application of NEAT and NETSCAN to trajectory clusters having overlapping road segments. (a) Trajectory clusters ground truth. (b) NEAT trajectory clusters. (c) NETSCAN trajectory clusters.

used for network constrained trajectories are: 1) *Dissimilarity with length* (DSL) [6], which is the ratio of the length of common segments between two trajectories and the sum of their lengths, and 2) *Hausdorff distance* [7], [8], [18], which represents the maximum mismatch level between two point sets. Both of these distance measures are not suitable for city road networks, which generally consist of a parallel and perpendicular grid of road segments, as they do not consider the reachability between various roads in the network and disregard the temporal information in the trajectories. In this paper, we propose a novel Dijkstra based DTW distance measure, trajDTW, which is suitable for large numbers of overlapping trajectories in dense road networks. Previously, Wang *et al.* [22] used DBSCAN for trajectory clustering, whereas Roh and Hwang [8] used an efficient agglomerative hierarchical clustering method to reduce distance computations without requiring an index structure, with little loss in the quality of clustering results. However, these methods are not scalable to large numbers of trajectories in a city environment as the computation of the distance matrix is time intensive. DBSCAN is computationally efficient for equal-dimensional feature space representations of datapoints since data structures such as $k-d$ trees can be used to efficiently find the nearest neighbors without the need to calculate the distance matrix for the entire dataset. However, for road network constrained trajectories represented by the GPS coordinates of varying numbers of sample points along the path, DBSCAN would require the distance matrix of all the points in the dataset as an input. In this paper, we develop Fast-clusiVAT, a fast and efficient version of our two-stage intelligent sampling based scalable hierarchical clustering algorithm, clusiVAT [13], [15] for the clustering task. Fast-clusiVAT yields an estimate of the number of clusters present in the dataset and identifies and visualizes the trajectories belonging to each cluster while being scalable to large datasets.

Most of the work done in the area of trajectory clustering uses synthetic datasets having small to medium numbers of trajectories [4], [6], [11]. Only a few papers have used real trajectory datasets for experimentation. For example, Guo *et al.* [23] use a dataset consisting of the GPS traces of trucks in Athens, Greece, for a total of 276 trajectories. A real-life trajectory dataset containing 214 trajectories, having an average trajectory length in the range of 18 to 1486 GPS points were used in [8]. The number of trajectories in these real-life datasets is fairly small. In this paper, we use the data from the GPS traces of 15,061 taxis in Singapore over a period of one month. We extract 3.28 million passenger trajectories from this dataset as the basis for our clustering experiment. 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.

TABLE I
TRAJECTORY CLUSTERS SHOWN IN FIG. 1(A)

Cluster name	Trajectory	Representing color
Traj_Cluster_1	{1,2,3,4,5}	Green
Traj_Cluster_2	{8,3,4,9}	Black
Traj_Cluster_3	{6,3,4,7}	Magenta
Traj_Cluster_4	{10,11,12}	Brown
Traj_Cluster_5	{15,14,13}	Red

III. LIMITATIONS OF FLOW BASED SCHEMES

Flow-based trajectory clustering schemes such as NETSCAN [11] and NEAT [4] are the only options in the literature to cluster large volumes of road network constrained trajectory data since traditional clustering schemes are plagued by scalability issues. Although fast and efficient, flow-based clustering algorithms are not suitable for analyzing people movements and proposing public transport routes on a dense city road network for the following reasons:

1) They do not specifically cluster trajectories but instead cluster the road segments based on their traffic flow, and then create a set of consecutive road segments having continuity of traffic density and flow, which may not refer to a specific trajectory.
2) NEAT discards information about the number of trajectories as it retains only the information about density when finding base clusters. As a result, it is difficult to find the number of trajectories in each cluster, which is an important factor in planning public transport routes and frequencies.
3) NETSCAN can assign a trajectory to more than one cluster, which can lead to loosely bound clusters that can inflate the number of trajectories belonging to each cluster.

To illustrate the above-mentioned points, consider an example of five trajectory clusters shown in Fig. 1(a), where the ground truth partition is represented by five different colors. Table I lists the road edge sequence and the corresponding color for each of the five clusters shown in Fig. 1(a). This example depicts two common scenarios in a typical city road network environment. In the first scenario exemplified by Traj_Cluster_1, Traj_Cluster_2 and Traj_Cluster_3, different vehicles originate from different locations outside the city (edges 1, 8 and 6 for the three clusters respectively),

travel on several common road segments in the city center (edges 3 and 4) and then move in different directions to the other side (edges 5, 9 and 7). In the second scenario, trips originate from different parts of the city and end at one central location and vice versa. For example Traj_Cluster_4 and Traj_Cluster_5 start from the north and south extremes (edges 10 and 15) respectively, and end at the center (the common node between edges 12 and 13).

The trajectory clusters obtained by applying the NEAT and NETSCAN algorithms to the dataset shown in Fig. 1(a) are shown in Fig. 1(b,c) respectively using different colors. In terms of the road network edges, the clusters obtained by NEAT are: NEAT_Cluster_1 $= \{1, 2, 3, 13, 14, 15\}$, NEAT_Cluster_2 $= \{10, 11, 12, 4, 5\}$, NEAT_Cluster_3 $= \{6, 8\}$ and NEAT_Cluster_4 $= \{7, 9\}$. None of the trajectory clusters obtained by NEAT is representative of actual traffic flows as there is no trajectory in the ground truth dataset following these paths. This problem is encountered because NEAT assigns one road segment to only one cluster, and hence is not effective when two or more trajectory clusters share multiple road segments in their path. It also tends to join road segments to form the longest trajectory path when many trajectories originate or end at a particular location. The clusters obtained by NETSCAN represent the start, middle and end part of the three ground truth clusters (Traj_Cluster_1 - Traj_Cluster_3). Traj_Cluster_4 and Traj_Cluster_5 are however correctly classified by NETSCAN since they do not have any overlapping segments. The clustering results obtained using these flow based algorithms are inconclusive for passenger movement pattern analysis and for planning better public transport routes. In order to address these limitations, we propose a framework using a novel distance measure and a fast and scalable clustering algorithm. Next, we formally define the road network constrained trajectory clustering problem framework.

IV. Problem Definition

We represent the road network as an undirected graph

$$G_{RN} = (V, E), \tag{1}$$

where V is a set of intersections/end-points 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}. For such a road network, a trajectory T of length l (which varies between trajectories) is defined as $T = [t_1, t_2, \ldots, t_l]$, 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.

A. Distance Measure (trajDTW)

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. Dijkstra's shortest path distance between any two edges in the road network is the sum of the edge weights of the shortest path tree obtained for a graph with non-negative edge path costs. 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

$$D_{all_{i,j}} = Dijkstra(E_i, E_j). \tag{2}$$

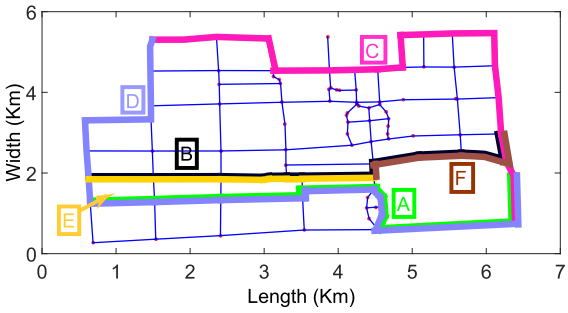


Fig. 2. Trajectory distance measure example.

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.

Algorithm 1 trajDTW

Input : $T_1 = [t_1^1, t_1^2, \ldots, t_1^l]$ − Trajectory 1
$T_2 = [t_2^1, t_2^2, \ldots, t_2^m]$ − Trajectory 2
D_{all} − Distance matrix of all the edges in G_{RN}
Output: $dist$ − distance between T_1 and T_2

$w = \frac{1}{2} \times min(l, m)$ − window parameter

for $i \leftarrow 1$ **to** $l + 1$ **do**
  for $j \leftarrow 1$ **to** $m + 1$ **do**
    $A_{i,j} = \infty$
  end
end
$A_{1,1} = 0$
for $i \leftarrow 1$ **to** l **do**
  for $j \leftarrow max(i - w, 1)$ **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 that has a grid of closely spaced parallel and perpendicular road segments. As an example, consider the road network shown in Fig. 2 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 1), DSL, and Hausdorff distance between selected pairs of trajectories is given in Table II. Intuitively, trajDTW distance gives a fair estimate of the distance between trajectories. For example,

TABLE II
DISTANCE MATRIX FOR THE TRAJECTORIES IN FIG. 2

Trajectories	trajDTW	DSL	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

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 considered 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 Fig. 2. Trajectories E (yellow) and F (brown), which belong to adjacent but nonoverlapping 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 have a high Hausdorff distance (Hausdorff(A-D) $=$ 4.35) because of the divergence at the end. Similarly, trajectories E and F are subtrajectories of B, so should be considered close to B. While trajDTW(B-E) $=$ 0.96 and trajDTW (B-F) $=$ 0.94 support 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. 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 that can affect DTW as a distance function.

B. Nondirectional trajDTW

The movement direction of trajectories can result in misleading distances between them, which in turn may cause incorrect clustering results. The problem of having an erroneous distance measure due to the directionality of the trajectories can be easily solved by reversing one trajectory (so that the starting point becomes the ending point and vice versa) and taking the minimum value of the distance between the first trajectory and the second trajectory, and the first trajectory and the reversed second trajectory. We denote this measure as nondirectional trajDTW distance, which is given as

$$\begin{aligned}&\text{nondirectional\_trajDTW}(T_i, T_j)\\&\quad = min(trajDTW(T_i, T_j), trajDTW(\text{flip}(T_i), T_j)),\end{aligned} \quad (3)$$

where flip(T_i) reverses the coordinate order of T_i. Next we describe in detail our novel Fast-clusiVAT algorithm, which makes use of this distance measure.

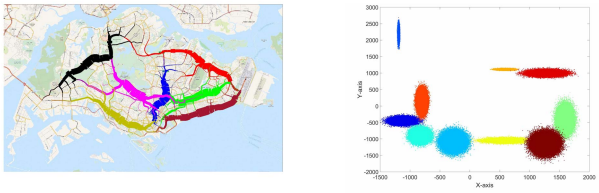
V. FAST-CLUSIVAT ALGORITHM

We propose a fast version of our clusiVAT algorithm [13], [15] for efficiently clustering large volumes of trajectory data. The clusiVAT algorithm for clustering big data uses *Reordered dissimilarity images* (RDIs) for the visual representation of the structure in unlabeled dissimilarity data. It finds its roots in the *visual assessment of clustering tendency* (VAT) [24] and *improved visual assessment of clustering tendency* (iVAT) [25] algorithms. VAT/iVAT reorder and modifies the input distance matrix D of the N datapoints by using a modified Prim's algorithm and applying a geodesic graph distance conversion [26]. The reordered and modified distance matrix, when displayed as a gray-scale image, shows possible clusters as dark blocks along the diagonal. However, VAT/iVAT suffer from size limitations as they have a space and time complexity of $O(N^2)$. To overcome this limitation, Kumar *et al.* [13], [15] proposed an intelligent sampling based algorithm, clusiVAT to assess cluster tendency and subsequent clustering for big data. The clusiVAT algorithm essentially consists of the following four steps:

1) **Maximin sampling:** This step select k' (an approximation of the number of clusters in the dataset (given as an input)) distinguished objects that partition the dataset into k' (almost) equally sized partitions using the maximin sampling scheme [27].
2) **Choosing n samples:** Trajectories are then randomly chosen from each of the k' partitions to generate a total of n sample trajectories (given as an input), where n is small so that VAT/iVAT can be effectively applied to the n samples.
3) **VAT/iVAT:** VAT/iVAT is then applied to the small D_n distance matrix of the n samples to provide an estimate of the number of clusters in the dataset and the subsequent clustering.
4) **NPR:** The k-partition of the n samples is non-iteratively extended to the remaining objects in the dataset using the nearest prototype rule.

The algorithmic implementations and pseudocode of the VAT, iVAT and clusiVAT algorithms are well documented in [15], [24], and [25] and are not reproduced here for brevity.

The clusiVAT algorithm was developed with the assumption that the distance function (usually Euclidean distance) can be computed quickly and can be performed as a batch operation, i.e., the Euclidean distance of a datapoint from $M \gg 1$ datapoints can be computed as a single operation using matrix properties. This assumption, however, does not hold good for the trajDTW distance measure proposed in this paper which is computationally expensive and can be computed pair-wise only. To illustrate this point, we consider two datasets, the first one is a collection of 1 million trajectories consisting of seven clusters along the major expressways of Singapore and the second one consisting of 1 million 2-dimensional (2D) points distributed among 10 clusters as shown in Fig. 3 (different clusters shown by different colors). We execute clusiVAT algorithm on both the datasets, where we use trajDTW distance measure for the trajectory dataset shown in Fig. 3(a) and Euclidean distance for the 2D dataset shown in Fig. 3(b).



(a) (b)

Fig. 3. Two datasets containing 1 million datapoints each. (a) Trajectory data (7 clusters). (b) 2D data (10 clusters).

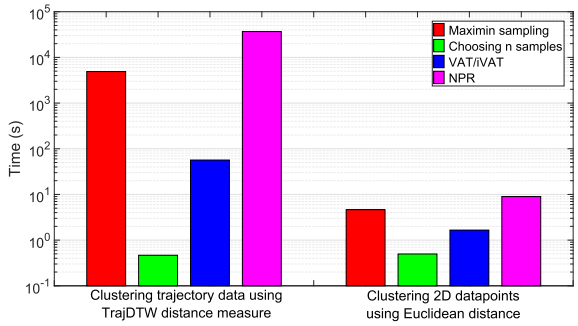


Fig. 4. Time taken for the four steps of the clusiVAT algorithm for the two datasets shown in Fig. 3.

For both the experiments, the clusiVAT parameters are set to $k' = 100$, and $n = 1000$. Fig. 4 shows the computation time (in log scale) of the four steps of the clusiVAT algorithm (explained above) for both the datasets. The first and the fourth step, i.e., maximin sampling and NPR, took around 12 hours to run and account for 99.86% of the total runtime of the clusiVAT algorithm for the trajectory dataset using the trajDTW distance measure. In contrast, the total runtime of the clusiVAT algorithm (all four steps) on a dataset of 1 million 2D points using Euclidean distance is just 16 seconds. In this paper, we propose Fast-clusiVAT, an adaptation of clusiVAT for time-consuming distance measures. Essentially, we propose modifications for the two most time-consuming steps: maximin sampling and NPR for faster runtime without significantly compromising on the accuracy.

A. Choosing Optimal k' for Maximin Sampling

The first step (maximin sampling) consists of $k' \times N$ distance computations to select from the dataset k' distinguished objects that partition the dataset into k' (almost) equally sized partitions. N being the size of the dataset is large and fixed, so an optimal value of k' could reduce the computation time of this step significantly. k' is an estimate of the actual number of clusters in the data and is usually chosen to be a large value to be on the safe side. The aim of this step and the next (choosing n samples) is to intelligently choose n samples that are almost equally distributed among the different clusters as the N trajectories in the big dataset, i.e., to obtain a representative sample. To demonstrate that an optimal value of k' exists, and beyond which the n samples are as equally distributed among the k ground truth clusters as the entire N trajectories, we experiment on the 1-million trajectory dataset shown in Figure 3(a) and increase k' from 1 to 100 (for example) and for each k', choose n random

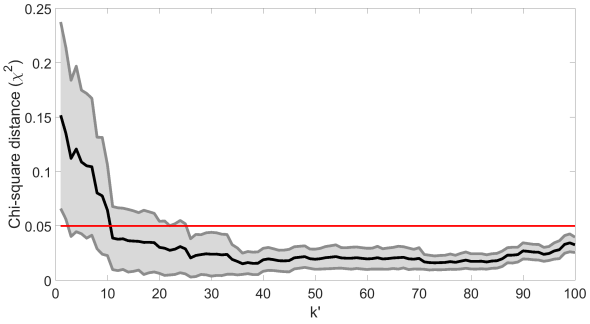


Fig. 5. Chi-square distance between ground truth distribution of N trajectories and n samples for different values of k'.

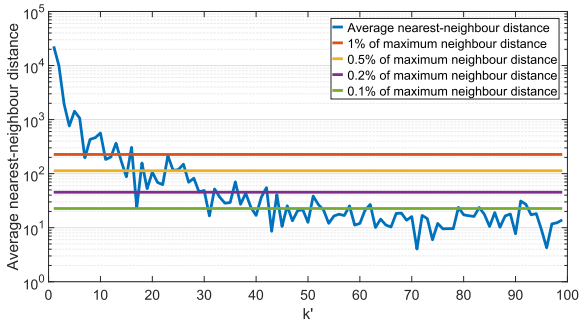


Fig. 6. Average nearest-neighbor distance for various k'.

samples from the k' partitions of the trajectory dataset and compare the distribution of the ground truth clusters among the n samples and N trajectories. We use the *Chi-square distance* [28] to compare the histograms of the n samples and N trajectories among the ground truth labels. The Chi-square distance between two distributions (histograms) A, and B is defined as follows:

$$\chi^2(A, B) = \frac{1}{2}\sum_{i=1}^{k}\frac{(A_i - B_i)^2}{A_i + B_i}, \tag{4}$$

where k, the number of ground truth clusters, is the number of bins in the histograms of each set. $\chi^2(A, B) \in [0, \infty)$, where lower values imply higher similarity between the two distributions (samples being representative) and vice versa. Fig. 5 shows the average value of χ^2 enveloped by the standard deviation for n randomly chosen samples for different values of k' (where the experiment is repeated 1000 times). As Fig. 5 demonstrates, χ^2 drops sharply as k' increases and becomes less than the statistically significant value of $\chi^2 = 0.05$ ($\sim 98\%$ confidence) for $k' > 10$, suggesting that any value of $k' > 10$ is expected to produce equally representative samples. However, the computation time increases linearly with k', i.e., an experiment with $k' = 10$ is expected to produce similar results as with $k' = 100$ in only 10% of the time.

Unfortunately, it is not always possible to find the optimal value of k' using the Chi-square distance described above due to the non-availability of ground truth labels. To solve this problem, we make use of the change in the average *nearest-neighbor* (NN) distance between the k' samples as a fraction of the maximum neighbor distance as k' increases. Fig. 6 shows a plot of the average nearest neighbor distance between the k'

TABLE III

CHI-SQUARE DISTANCE STATISTICS AND COMPUTATION TIME FOR DIFFERENT VALUES OF AVERAGE NN DISTANCE RATIO (α)

α	Chi-square	Computation time (% of $k' = 100$)
1	0.1045 ± 0.0629	7.40
0.5	0.03614 ± 0.02783	14.65
0.2	0.03494 ± 0.02952	16.79
0.1	0.02415 ± 0.01959	30.09

distinguished trajectories for different values of k' (blue plot). As k' increases, the k' distinguished trajectories are closer to each other, and the rate of decrease of the average NN distance drops significantly. Fig. 6 also shows the four horizontal lines (shown by different colors) representing for what value of k', the average NN distance reaches $\alpha = \{1, 0.5, 0.2, 0.1\}\%$ of the maximum neighbor distance. Table III lists the mean and standard deviation of the Chi-square distance between the distribution of the n samples and N trajectories, as well as the time taken to find k' distinguished trajectories (as a percentage of the time required to compute $k' = 100$ distinguished trajectories) for different values of α. As α decreases, k' increases and hence the computation time also increases. For $\alpha = 0.5$, the n sample trajectories are representative of the N trajectories with ~98% accuracy and are computed in less than 15% of the time required to compute $k' = 100$.

B. Choosing Optimal k and Finding Clusters of the Samples

The MST built using Prim's algorithm in VAT/iVAT (step 3 of clusiVAT) provides an array that represents 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 trajectory dataset $\mathbf{T}$ by k dark blocks along the diagonal. Having this estimate, we cut the $k-1$ largest edges in the MST, resulting in k connected subtrees (the clusters). 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 most of the $k-1$ longest edges of the MST. A more useful approach in such a scenario is to manually select the dark blocks along the diagonal representing the k clusters.

C. Approximate NPR Labeling

The most time consuming step in the clusiVAT application for the trajectory data using trajDTW distance is step 4: NPR (see Fig. 4). This step consists of computing the distance of each of the $N-n$ non-sampled trajectories from the n sample trajectories and finding the nearest sample trajectory, a total of $(N-n)\times n$ distance computations, hence the high runtime. We take advantage of the reordering of the n samples by the VAT algorithm during step 3 of clusiVAT to reduce the computational complexity of the NPR step in Fast-clusiVAT.

Let $\{T_n^1, T_n^2, \ldots, T_n^n\}$ be the n (reordered) sample trajectories obtained as the output of the VAT/iVAT application to the n samples. Since VAT reordering is essentially building a *minimum spanning tree* (MST), it places nearer trajectories

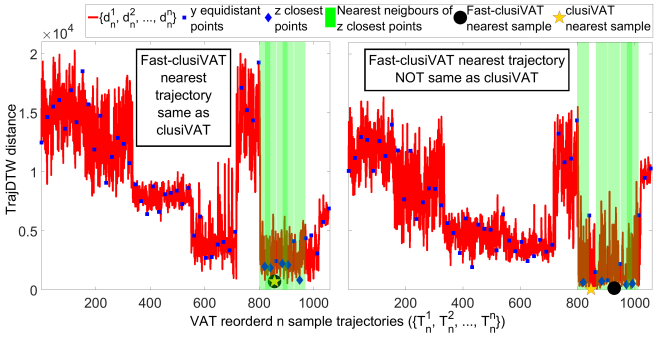


Fig. 7. An example illustrating Fast-clusiVAT NPR approximation.

(in terms of the distance measure), closer to each other in the ordering of $\{T_n^1, T_n^2, \ldots, T_n^n\}$, i.e., the distance between T_n^k and T_n^{k+1} is likely to be smaller than T_n^k and T_n^{k+l}, $l > 1$. In such a scenario, if the distance of a non-sampled trajectory T_N^x from T_n^k is large, then it is likely to be large for T_N^x and T_n^{k+1} as well, and hence it is not necessary to search for the closest of $\{T_n^1, T_n^2, \ldots, T_n^n\}$ from T_N^x.

We propose a novel approach for fast (but approximate) computation of NPR. For this, we first choose y equidistant trajectories (in terms of ordering) from the n VAT reordered samples $\{T_n^1, T_n^2, \ldots, T_n^n\}$. Let these y trajectories be:

$$\{T_n^1, T_n^{\lceil \frac{n}{y-1} \rceil}, T_n^{\lceil \frac{2\times n}{y-1} \rceil}, \ldots, T_n^{\lceil \frac{(y-2)\times n}{y-1} \rceil}, T_n^n\}.$$

Let the distance of a non-sampled trajectory T_N^x from these y equidistant trajectories be:

$$\{d_n^1, d_n^{\lceil \frac{n}{y-1} \rceil}, d_n^{\lceil \frac{2\times n}{y-1} \rceil}, \ldots, d_n^{\lceil \frac{(y-2)\times n}{y-1} \rceil}, d_n^n\}$$

respectively. We choose (say) z trajectories from the y equidistant samples, which are closest to T_N^x. Let these z trajectories be:

$$\{T_n^{z1}, T_n^{z2}, \ldots, T_n^{zz}\}.$$

We then find the distance of T_N^x from all the nearest neighbors of $\{T_n^{z1}, T_n^{z2}, \ldots, T_n^{zz}\}$, which do not belong to the y equidistant trajectories, and declare the nearest trajectory as the nearest prototype.

To illustrate this process with an example, Fig. 7 shows two cases where the Fast-clusiVAT NPR approximation produces same results as clusiVAT (left subfigure) and different (but nearby) results as clusiVAT (right subfigure).

The red curve represents the distances

$$\{d_n^1, d_n^2, \ldots, d_n^n\}$$

of the non-sampled trajectory T_N^x from the VAT reordered sampled trajectories

$$\{T_n^1, T_n^2, \ldots, T_n^n\}.$$

The clusiVAT NPR step requires calculating all these n distances for each of the $N-n$ non sampled trajectories. From this, the nearest prototype is chosen as the trajectory that is closest (shown by the yellow star). For the approximate NPR presented in this paper, we first find the distance of T_N^x from the $y = 50$ (chosen for this illustration) equidistant

TABLE IV
ACCURACY AND RUN TIMES FOR APPROXIMATE NPR FOR DIFFERENT VALUES OF y AND z

	y \ z	2	5	10
Nearest prototype accuracy (%)	50	80.76	91.30	96.25
	100	76.23	90.49	96.22
	200	76.37	87.21	92.17
	y \ z	2	5	10
Ground truth cluster accuracy (%)	50	99.40	99.71	99.88
	100	99.70	99.86	99.90
	200	99.56	99.65	99.68
	y \ z	2	5	10
Run time (% of NPR run time)	50	16.50	24.69	37.15
	100	21.62	25.93	32.30
	200	35.08	37.34	40.09

trajectories shown by blue squares. Of these, we chose $z = 5$ (again, chosen for this illustration) nearest trajectories, which are shown in Fig. 7 by blue diamonds. The nearest neighbors (in terms of ordering) of these z trajectories include all the trajectories inside the green vertical stripes. The nearest of these nearest neighbors (shown by the black circle in Fig. 7) is declared as the nearest prototype for this approximate NPR approach. For the example shown in the left view, the nearest prototype found using NPR and approximate NPR are the same. However, for the right view, they are different (although nearby and possibly belonging to the same ground truth cluster).

Table IV lists the nearest prototype and ground truth label accuracies and run time for the approximate NPR for $y = \{50, 100, 200\}$ equidistant trajectories and $z = \{2, 5, 10\}$ nearest trajectories. The approximate NPR correctly identifies the nearest sampled trajectory in more than 75% of the cases, and in those cases where the approximate NPR trajectory is not same as the NPR one, they belong to the same cluster almost all of the time (the ground truth cluster accuracy being more than 99.40% for all of the cases). This high accuracy is achieved with a run time that can be as low as just 16.5% of the run time required for traditional NPR. Hence, approximate NPR for the Fast-clusiVAT presents a fast and accurate way to label non-sampled trajectories of the big dataset.

D. Two-Stage Fast-clusiVAT Algorithm

We adapt our two-stage clusiVAT algorithm described in [29] to the road network trajectory data to separate trajectory clusters where the vehicles are moving in opposite directions. In the first stage, we use nondirectional_trajDTW as a distance measure to cluster the Singapore taxi trajectory dataset consisting of 3.28 million trajectories using the Fast-clusiVAT algorithm, which ensures that trajectories that have opposite starting and finishing points, but follow similar paths, are clustered in the same group. In the next stage, we use trajDTW (directional) as the distance measure for the sample trajectories of each cluster obtained in the first step to separate the trajectories going in opposite directions using the iVAT algorithm. Later we use approximate NPR with the (directional) trajDTW distance measure to assign non-sampled trajectories to one of the clusters obtained after the second stage of the two-stage Fast-clusiVAT. The schematic flow chart of the application of two-stage Fast-clusiVAT to the trajectory dataset is shown in Fig. 8.

VI. TIME COMPLEXITY

In this section, we discuss the time complexity of our proposed trajDTW distance measure and the two-stage Fast-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 $O(|E|+|V|log(\frac{|E|}{|V|}+|V|))$ [30]. For two trajectories of length l and m, the time complexity of a standard DTW algorithm is $O(l \times m)$ for each distance computation.

For the trajectory dataset T containing N trajectories, the first step in Fast-clusiVAT is the selection of optimal k'. For clusiVAT, this step has a time complexity of $O(k' \times N)$, where k' is a user-defined input for an overestimate of the number of clusters in the data and is usually chosen to be (needlessly) large (usually 50 or 100). However, Fast-clusiVAT reduces the time taken by this step by optimally choosing k' such that samples are still representative. The optimal value of k' obtained from Fast-clusiVAT is usually just 10-20% of the k' chosen for clusiVAT, thus reducing the run-time of this step by as much as 85% as shown in Section V-A and Table III. The next step in Fast-clusiVAT is to randomly select trajectories from the k' partitions to get a total of n sample trajectories. 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). All these steps (after selecting k' distinguished objects) are pretty fast and are just a fraction ($< 1\%$) of the total run-time of Fast-clusiVAT. The final step of Fast-clusiVAT uses an approximate NPR to label the non-sampled trajectories. The time complexity of the NPR step used in clusiVAT requires $(n \times (N - n))$ trajDTW distance computations to find the nearest prototype of each of the N non-sampled trajectories. However, for Fast-clusiVAT, the approximate NPR uses the VAT reordering property and requires only $((y + (\lceil \frac{n}{y-1} \rceil) \times z) \times (N - n))$ trajDTW distance computations, where $\{y, z\} \ll n$. The approximate NPR of Fast-clusiVAT on average takes only one-fifth of the time required for the NPR of clusiVAT, depending on the choice of the parameters y and z as demonstrated in Section V-C and Table IV.

VII. TAXI DATA AND TRIP EXTRACTION

In this paper, we aim to understand urban mobility using the most popular route taken by taxi passengers for their commute and find the time distribution of trips for each route. We use our Singapore taxi data for this purpose and refrain from using

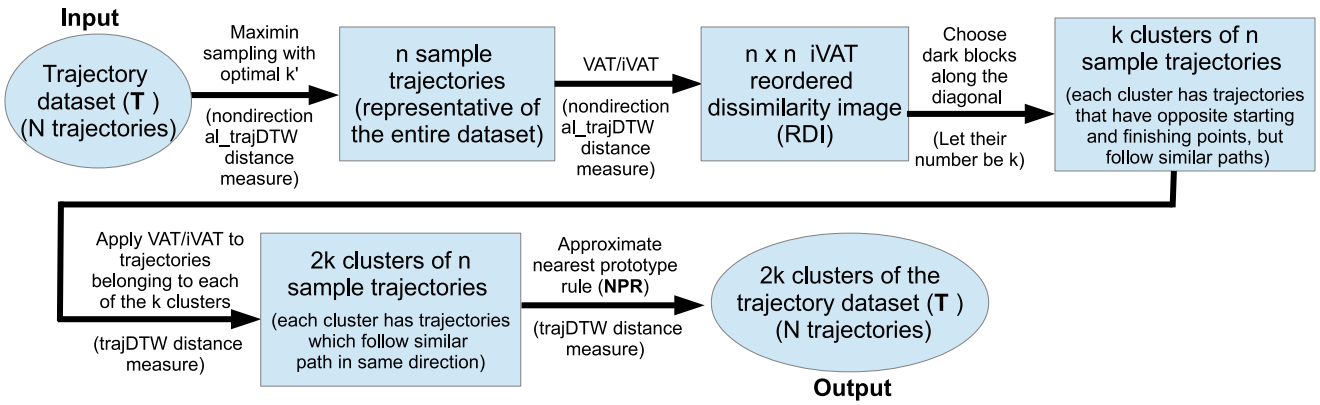


Fig. 8. Flow chart schematic of the two-stage Fast-clusiVAT algorithm.

the publicly available T-drive taxi trajectory data [31] since it does not have information regarding the taxi occupancy state, and hence it is not possible to find if the taxi trip corresponds to a passenger trip or if the taxi is vacant. To find the most popular routes taken by taxi passengers along with their time distribution, we first extract the passenger trips from the raw taxi GPS traces as described below.

The dataset consists of the trajectories of 15,061 taxis collected over a duration of 1 month. The dataset is very dense as it consists of more than 370 million GPS logs. The general format of each datapoint is as follows: {Time Stamp, Taxi Registration, Latitude, Longitude, Speed, Status}. The "Status" field of each datapoint contains information about the occupancy state of the taxi. This GPS log dataset is processed to obtain taxi trip information before applying the clustering framework on the passenger taxi trips. In order to extract each individual taxi's trips from the raw taxi data, we detect the following taxi status sequence: starting from *FREE* to *Passenger on Board (POB)* and ending from *POB* to *FREE*. We have used the same trip extraction framework as presented in [32].

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 subsequent road segment, we first map each GPS point to its nearest road segment (commonly known as the *Map Matching problem*). We use the popular open source map matching tool GraphHopper [33], which provides an implementation of the approach presented in [19].

VIII. Numerical Experiments

A. Experimental Setup

We use OpenStreetMap (OSM) to extract the Singapore city road network graph. We use those roads for which the "highway" key has one of the following values {motorway, motorway_link, trunk, trunk_link, primary, primary_link, secondary, secondary_link, tertiary}. We combine the edges joining the same pair of nodes to remove different lanes of the same road segment to create our road network. After map matching, we are left with the trajectory dataset $\mathbf{T} = \{T_1, T_2, \ldots, T_N\}$ having $N = 3.28$ million trajectories, whose lengths lie in the range of 10 to 250 road segments and have an average of 22 road segments. We divide the trajectories into 8 parts based on the time of day during which the trip is being made. Specifically we divide the 24 hours of a day into 8 intervals of 3 hours each (12am - 3am, 3am - 6am, 6am - 9am, 9am - 12pm, 12pm - 3pm, 3pm - 6pm, 6pm - 9pm and 9pm - 12am). The trajectories belonging to each of the 8 time intervals are clustered using the Fast-clusiVAT algorithm using the parameter values $\alpha = 0.5$ and $n = 1000$ and nondirectional_trajDTW as the distance measure. The 8000 sample trajectories (1000 each from 8 time intervals of 3 hours each) are again clustered using Fast-clusiVAT and the same parameter values of α and n, and again using nondirectional_trajDTW as the distance measure. The trajectories belonging to the entire dataset are mapped to one of the 1000 sample trajectories using the approximate NPR described in Section V-C using the parameters $y = 50$ and $z = 5$. The purpose of this exercise is to obtain a time-based distribution of trajectories belonging to a particular cluster, thus giving insights into how urban mobility changes with the time of day.

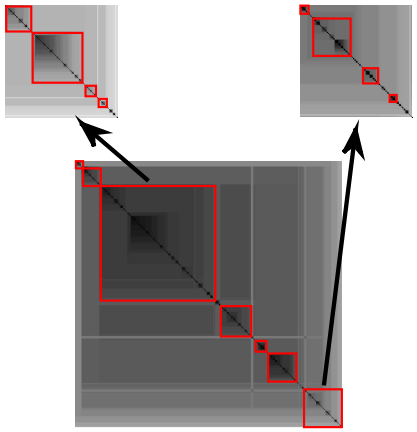


Fig. 9. clusiVAT image of the trajectory dataset.

B. Results

The clusiVAT image of the final 1000 samples is shown in the bottom part of Fig. 9. 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 bottom image in Fig. 9 seems to have seven primary dark

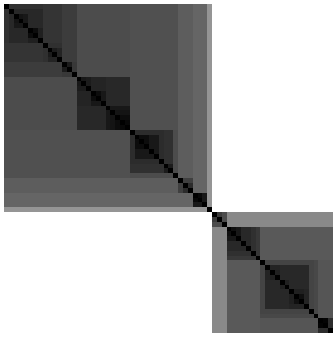


Fig. 10. iVAT image of samples (stage 2).

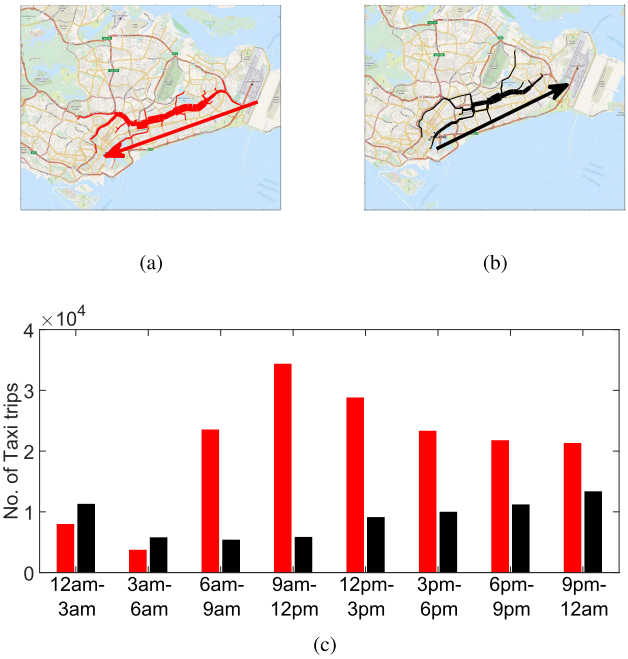


Fig. 11. Subclusters obtained using the two-stage clusiVAT clustering. (a) trajDTW_C1_1. (b) trajDTW_C1_2. (c) Trip distribution with time.

blocks, and embedded in them is a fine structure that has many more. The zoomed images of the third and seventh dark blocks shown in the upper part of Fig. 9 reveal the presence of four dark sub-blocks each representing a total of 13 embedded sub-clusters. These dark blocks are marked by red rectangles for clarity.

In the next stage, we use the iVAT algorithm using (directional) trajDTW as the distance measure for the sample trajectories of each cluster obtained in the previous step in order to separate the trajectories going in opposite directions, producing a total of $13 \times 2 = 26$ clusters. Fig. 10 shows the iVAT image of the samples of one of the clusters obtained by the previous stage using (directional) trajDTW as the distance measure. The iVAT image shows the presence of two clusters by two dark blocks along the diagonal representing the trajectories going in opposite directions.

Fig. 11(a,b) shows the two trajectory sub-clusters (trajDTW_C1_1 and trajDTW_C1_2), which follow similar travel paths but in opposite directions (shown by two colors, red (gray in B/W print) and black), where view (a) shows the taxi trips from the airport and eastern suburbs to the city center, and the return trips are shown in view (b). The distribution of taxi trips in both the directions with respect to time is shown in Fig. 11(c), where the red (gray in B/W print) and black bars show the distribution of taxi trips in view (a) (shown by red (gray)) and view (b) (shown by black) respectively.

The longer red bars with respect to the black ones for most periods of the day except night time (12am - 3am and 3am - 6am) represent more taxi trips towards the city center by visitors and people going to work, whereas on the way back they prefer to use public transport. The longer black bars during the night time may represent people relying more on taxis to catch early morning flights as public transport is not available during the night.

Our algorithm discovered 13 clusters of non-directional taxi routes. Each of them is further divided into two sub-clusters according to the direction of the trip. The largest five clusters (each consisting of two sub-clusters) are shown in Fig. 12 as trajDTW_C2_1, trajDTW_C2_2, trajDTW_C3_1, trajDTW_C3_2, etc., where C2, C3, etc. represent the clusters obtained after the first stage (consisting of trajectories that have opposite starting and finishing points but following similar paths) and the suffix _1 and _2 in the notation separates the trajectories going in opposite directions. Fig. 12 also shows the distribution of trips belonging to each sub-cluster with respect to time. The clusters basically reflect the crowd movement of citizens in different time periods and cover most major roads (i.e., express roads) in Singapore. They can be further classified into two groups, depending on whether each trip cluster passes through the city center or not. Fig. 12(b,e,h) show the trips from the suburban areas in the north, northwest, and west to the city center via three major expressways leading to the city from the three areas respectively, namely CTE, PIE, and AYE. Furthermore, we can see a clear temporal pattern of these three clusters in Fig. 12(c,f,i), which involves a peak from 9 am to 12 pm (the longest black bar among all black bars in each of the views in Fig. 12(c,f,i) corresponds to the 9 am - 12 pm slot). The majority of citizens live in suburban areas and work in/around the city center. This result can quantitatively show the taxi usage by working people and how it impacts the traffic on major roads. As a counterpart, Fig. 12(a,d,g) shows the clusters in opposite directions (city center to suburban areas in the north, northwest, and west). For these three clusters, the temporal patterns (red bars in Fig. 12(c,f,i)) show the higher trip volumes in the afternoon, but there is no obvious peak period. It is consistent with the fact that in Singapore, the daily off-duty time is less concentrated than the busy hours in the morning. Further, approaching or during midnight, a peak emerges. This is probably because of people going home after overtime work when public transport services stop.

Fig. 12(j,k,l) shows a pattern of taxi trips from the north to the east during the morning peak hours (6 am - 9 am for red bars) and back during the evening peak hours (6 pm - 9 pm and 9 pm - 12 am for black bars). There are industries in the east of Singapore, e.g., Changi Business Park, and the north of Singapore is a residential area. Thus there are many working people traveling between the two places daily using a taxi. One of the main reasons is the inconvenience of public

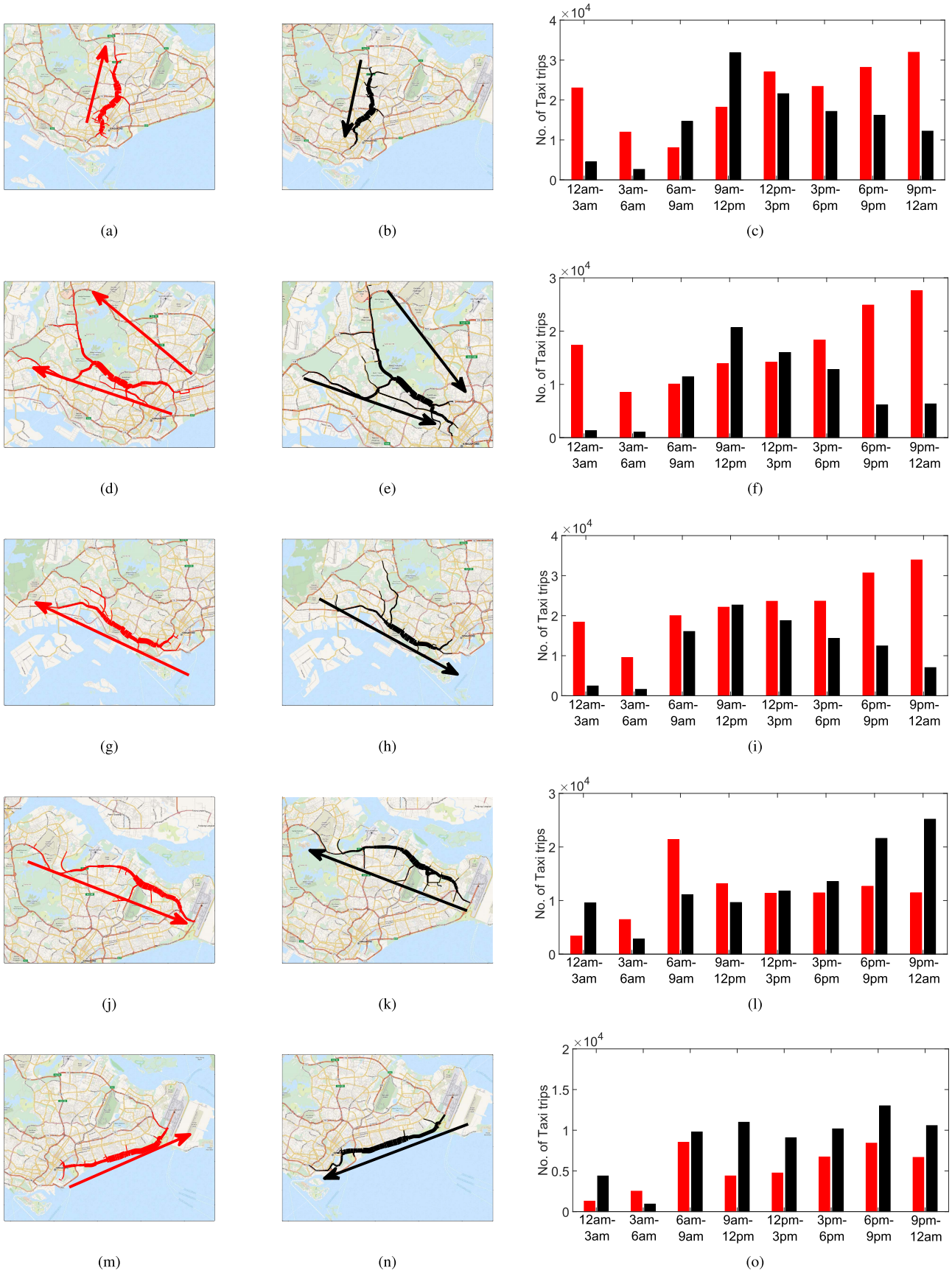


Fig. 12. Trajectories belonging to different clusters for the Singapore taxi trajectory dataset. (a) trajDTW_C2_1. (b) trajDTW_C2_2. (c) Trip distribution with time. (d) trajDTW_C3_1. (e) trajDTW_C3_2. (f) Trip distribution with time. (g) trajDTW_C4_1. (h) trajDTW_C4_2. (i) Trip distribution with time. (j) trajDTW_C5_1. (k) trajDTW_C5_2. (l) Trip distribution with time. (m) trajDTW_C6_1. (n) trajDTW_C6_2. (o) Trip distribution with time.

TABLE V
CLUSTER VALIDITY INDICES AND RUN TIME FOR THE TRAJECTORY CLUSTERS OBTAINED USING DIFFERENT CLUSTERING SCHEMES

	Two-stage Fast-clusiVAT	Two-stage clusiVAT	DBSCAN	OPTICS	NETSCAN	NEAT
Dunn's index	0.012	0.012	0.002	0.004	0	0
Silhouette index	0.250	0.256	0.138	0.154	-0.101	0.006
Run time (minutes)	110	750	400	587	84	75

transportation between these two areas. If a passenger would like to take the metro from the north to the east, he/she has to take North-South Line to the city and then transfer to East-West Line.[1] Also, the expressway connecting these regions, i.e., SLE and TPE shown in the clusters, are not as heavily used as expressways connecting the city center to suburban areas in the north, northwest, and west during the peak hours. Thus many travelers choose taxis for commuting.

The cluster in Fig. 12(m,n) shows passenger movement from the airport to the city center. There are many tourists coming to Singapore daily, who travel between the airport and the city center (since it has many tourist places and hotels). This group of people may be the major contributor to these two sub-clusters. In addition, locals that live in the east and work in the city may also contribute to these two clusters. Because of the likely majority of tourist travelers in these sub-clusters, the temporal patterns of these two clusters (Fig. 12(o)) are not very obvious.

We compare the trajectory clusters obtained using our novel two-stage Fast-clusiVAT clustering framework with five baseline approaches: the first one being clusiVAT, two density-based clustering algorithm: DBSCAN [16] and OPTICS [17] (using trajDTW as distance measure), and the remaining two are vehicular trajectory specific clustering approaches: NETSCAN [11] and NEAT [4]. The DBSCAN and OPTICS algorithms cannot be applied to the entire dataset that consists of 3.28 million trajectories since it would require the computation of a $(3.28 \times 10^6) \times (3.28 \times 10^6)$ distance matrix D_N, which is computationally prohibitive. Hence, we apply DBSCAN/OPTICS to the 8000 sample trajectories obtained after the first stage of the two-stage clusiVAT algorithm and later extend the k-partition of these samples to the remaining trajectories using approximate NPR. Since NEAT outputs just the flow clusters and does not assign the original trajectories in the dataset to these flow clusters, we use the last step of the NETSCAN algorithm to group the trajectories according to their similarity to each NEAT-generated flow cluster.

We compare the proposed algorithm with the baselines in terms of the quality of the obtained clusters as well as the run times. Since there are no ground truth labels available for the Singapore taxi trajectory dataset, we assess the quality of the clusters obtained from various algorithms using two popular internal cluster validity indices: Dunn's index [34] and Silhouette index [35], which aim to judge the quality of clusters based on measures such as the within-cluster variance and the inter-cluster separation. A higher value of these indices implies clusters are well separated and hence more meaningful. Table V lists the values of the Dunn's and Silhouette indices for the six schemes employed to cluster the Singapore taxi trajectory dataset: our novel two-stage Fast-clusiVAT, two-stage clusiVAT, DBSCAN, OPTICS, NETSCAN, and NEAT. Two-stage Fast-clusiVAT and two-stage clusiVAT produce the highest values of both the indices among all the comparable algorithms (although the run time of Fast-clusiVAT is approximately one-sixth that of clusiVAT), demonstrating that the clusters obtained from the novel framework proposed in this paper are better (clearly separated from each other) and more meaningful as compared to the other algorithms. The flow-based schemes, NETSCAN and NEAT, produce poor quality clusters (i.e., the clusters are not well separated from each other) as shown by the very small values of the Dunn's and Silhouette indices, and hence are not useful in planning public transport routes and frequencies. The run-time of Fast-clusiVAT is only marginally higher than NETSCAN and NEAT, however, the cluster quality is far superior. DBSCAN and OPTICS produce better clusters than the flow based schemes NETSCAN and NEAT, but the DBSCAN/OPTICS clusters are not as well separated compared to those obtained using the novel two-stage Fast-clusiVAT proposed in this paper.

[1]Circle Line commenced the full service in recent years, which can help to link the two places. However, the period of the data used in our experiments is before the Circle Line commencement.

IX. CONCLUSION

We presented a novel Dijkstra-based DTW distance measure, trajDTW, between two trajectories, which is suitable for large numbers of overlapping trajectories in a dense road network. We applied our new efficient clustering algorithm: two-stage Fast-clusiVAT for a road network containing a large number of vehicle trajectories. We performed our experiments on 3.28 million trajectories of passenger trips obtained from the real-life Singapore Taxi Trajectory dataset, which contains the GPS traces of 15,061 taxis within Singapore over a period of one month. For the clusters found using Fast-clusiVAT, we provide a time-based distribution of the trajectories to provide insights into how urban mobility changes with the time of day.

We compare the clusters obtained using our novel trajDTW distance measure based two-stage Fast-clusiVAT with that obtained using the popular general and trajectory specific clustering algorithms: DBSCAN, NETSCAN, and NEAT using two internal cluster validity measures, Dunn's and Silhouette indices. The two-stage clusiVAT produces the highest value of both the indices among all the comparable algorithms, demonstrating that the clusters obtained from the novel framework proposed in this paper are clearly separated from each other and more meaningful as compared to other algorithms. We conclude that while general clustering schemes such as DBSCAN using a specialized distance measure for trajectories are not scalable, flow and density based schemes: NETSCAN and NEAT, although scalable, are not suitable for clustering a

dense dataset consisting of millions of trajectories spread over a typical urban road network consisting of a criss-cross array of parallel and perpendicular road segments.

This analysis can effectively facilitate the understanding of spatial patterns in trajectories and has great significance for decision-makers to understand road traffic conditions and to propose metro bus corridors and light rail systems for better public transport. In the future, we would like to experiment on real-time route prediction for taxi passengers based on the clusters obtained using historical data. Such predictions can then be used for personalized location-based services that direct advertising material, special offers and discounts from businesses to the commuters most likely to pass near those business outlets and stores based on their travel trajectories.

REFERENCES

[1] E. Kamar and E. Horvitz, "Collaboration and shared plans in the open world: Studies of ridesharing," in *Proc. 21st Int. Joint Conf. Artif. Intell. (IJCAI)*, 2009, pp. 187–194.

[2] K. Zheng, Y. Zheng, N. J. Yuan, S. Shang, and X. Zhou, "Online discovery of gathering patterns over trajectories," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1974–1988, Aug. 2014.

[3] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1393–1402, Sep. 2013.

[4] B. Han, L. Liu, and E. Omiecinski, "Road-network aware trajectory clustering: Integrating locality, flow, and density," *IEEE Trans. Mobile Comput.*, vol. 14, no. 2, pp. 416–429, Feb. 2015.

[5] Y. Zheng, "Trajectory data mining: An overview," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 3, p. 29, 2015.

[6] J.-I. Won, S.-W. Kim, J.-H. Baek, and J. Lee, "Trajectory clustering in road network environment," in *Proc. IEEE Symp. Comput. Intell. Data Mining (CIDM)*, Mar./Apr. 2009, pp. 299–305.

[7] Y. Wang, Q. Han, and H. Pan, "A clustering scheme for trajectories in road networks," in *Proc. Int. Conf. Teach. Comput. Sci. (WTCS)*, vol. 117, 2012, pp. 11–18.

[8] G.-P. Roh and S.-W. Hwang, "NNCluster: An efficient clustering algorithm for road network trajectories," in *Database Systems for Advanced Applications*, H. Kitagawa, Y. Ishikawa, Q. Li, and C. Watanabe, Eds. Berlin, Germany: Springer, 2010, pp. 47–61. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-642-12098-5_4

[9] J. Kim and H. S. Mahmassani, "Spatial and temporal characterization of travel patterns in a traffic network using vehicle trajectories," *Transp. Res. Procedia*, vol. 9, pp. 164–184, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352146515001702, doi: 10.1016/j.trpro.2015.07.010.

[10] M. Nanni and D. Pedreschi, "Time-focused clustering of trajectories of moving objects," *J. Intell. Inf. Syst.*, vol. 27, no. 3, pp. 267–289, 2006.

[11] A. Kharrat, I. Popa, K. Zeitouni, and S. Faiz, "Clustering algorithm for network constraint trajectories," in *Headway in Spatial Data Handling*. Berlin, Germany: Springer, 2008, pp. 631–647.

[12] Z. Hong, Y. Chen, and H. S. Mahmassani, "Recognizing network trip patterns using a spatio-temporal vehicle trajectory clustering algorithm," *IEEE Trans. Intell. Transp. Syst.*, to be published. [Online]. Available: https://ieeexplore.ieee.org/document/8082114/, doi: 10.1109/TITS.2017.2754401.

[13] D. Kumar, M. Palaniswami, S. Rajasegarar, C. Leckie, J. C. Bezdek, and T. C. Havens, "clusiVAT: A mixed visual/numerical clustering algorithm for big data," in *Proc. IEEE Int. Conf. Big Data*, Oct. 2013, pp. 112–117.

[14] D. Kumar, S. Rajasegarar, M. Palaniswami, X. Wang, and C. Leckie, "A scalable framework for clustering vehicle trajectories in a dense road network," in *Proc. ACM SIGKDD Int. Workshop Urban Comput.*, 2015.

[15] D. Kumar, J. C. Bezdek, M. Palaniswami, S. Rajasegarar, C. Leckie, and T. C. Havens, "A hybrid approach to clustering in big data," *IEEE Trans. Cybern.*, vol. 46, no. 10, pp. 2372–2385, Oct. 2016.

[16] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 226–231.

[17] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," in *Proc. Int. Conf. Manage. Data (SIGMOD)*, 1999, pp. 49–60.

[18] 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 *Proc. SIGKDD Int. Workshop Urban Comput.*, 2013, Art. no. 9.

[19] P. Newson and J. Krumm, "Hidden Markov map matching through noise and sparseness," in *Proc. ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2009, pp. 336–343.

[20] J. Yuan, Y. Zheng, C. Zhang, X. Xie, and G.-Z. Sun, "An interactive-voting based map matching algorithm," in *Proc. 11th Int. IEEE Conf. Mobile Data Manage. (MDM)*, May 2010, pp. 43–52.

[21] P. C. Besse, B. Guillouet, J. M. Loubes, and F. Royer, "Review and perspective for distance-based clustering of vehicle trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3306–3317, Nov. 2016.

[22] X. Wang, X. Ma, and E. Grimson, "Unsupervised activity perception by hierarchical Bayesian models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2007, pp. 1–8.

[23] D. Guo, S. Liu, and H. Jin, "A graph-based approach to vehicle trajectory analysis," *J. Location Based Services*, vol. 4, nos. 3–4, pp. 183–199, 2010.

[24] J. C. Bezdek and R. Hathaway, "VAT: A tool for visual assessment of (cluster) tendency," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2002, pp. 2225–2230.

[25] T. C. Havens and J. C. Bezdek, "An efficient formulation of the improved visual assessment of cluster tendency (iVAT) algorithm," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 813–822, May 2012.

[26] J. Bouttier, P. D. Francesco, and E. Guitter, "Geodesic distance in planar graphs," *Nucl. Phys. B*, vol. 663, no. 3, pp. 535–567, 2003.

[27] M. E. Johnson, L. M. Moore, and D. Ylvisaker, "Minimax and maximin distance designs," *J. Stat. Planning Inference*, vol. 26, no. 2, pp. 131–148, 1990.

[28] O. Pele and M. Werman, "The Quadratic-Chi histogram distance family," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 749–762.

[29] D. Kumar, J. C. Bezdek, S. Rajasegarar, C. Leckie, and M. Palaniswami, "A visual-numeric approach to clustering and anomaly detection for trajectory data," *Vis. Comput.*, vol. 33, no. 3, pp. 265–281, 2017.

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

[31] J. Yuan, Y. Zheng, X. Xie, and G. Sun, "Driving with knowledge from the physical world," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2011, pp. 316–324.

[32] D. Kumar, H. Wu, Y. Lu, S. Krishnaswamy, and M. Palaniswami, "Understanding urban mobility via taxi trip clustering," in *Proc. 17th IEEE Int. Conf. Mobile Data Manage. (MDM)*, vol. 1, Jun. 2016, pp. 318–324.

[33] Graphhopper. (2017). *Map-Matching*. [Online]. Available: http://www.unhabitat.org/pmss/listItemDetails.aspx?publicationID=3387

[34] J. C. Dunn, "A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters," *J. Cybern.*, vol. 3, no. 3, pp. 32–57, 1973.

[35] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, 1987.

Dheeraj Kumar received the B.Tech. and M.Tech. dual degree in electrical engineering from IIT Kanpur, India, in 2010, and the Ph.D. degree in electrical and electronic engineering from The University of Melbourne, Australia, in 2017. He is currently a Post-Doctoral Researcher with the Lyles School of Civil Engineering, Purdue University, USA. His research interests include big data mining for the smart city and intelligent transportation applications, Internet of Things, enhancing urban mobility via unconventional data-driven analytics, and leveraging social media data for understanding emergency evacuations.

Huayu Wu received the Ph.D. degree in computer science from the School of Computing, National University of Singapore, in 2011. He was a Research Scientist with the Data Analytics Department, Institute for Infocomm Research, A*STAR, Singapore. He is currently a Data Scientist in the financial services industry. He is also an Instructor with Nanyang Technological University, Singapore. He is involved in applied research. His research interests are in general database and data analytics topics.

Sutharshan Rajasegarar received the Ph.D. degree from The University of Melbourne, Melbourne, VIC, Australia, in 2009. He is currently a Lecturer with the School of Information Technology, Deakin University, Geelong, Australia. His current research interests include anomaly/outlier detection, distributed machine learning, pattern recognition, signal processing, health analytics, cyber security, and Internet of Things.

Christopher Leckie received the B.Sc. degree, the B.E. degree in electrical and computer systems engineering (with first class honors), and the Ph.D. degree in computer science from Monash University, Australia, in 1985, 1987, and 1992, respectively. In 1988, he joined Telstra Research Laboratories, where he conducted research and development into artificial intelligence techniques for various telecommunication applications. In 2000, he joined The University of Melbourne, Australia, where he is currently a Professor with the Department of Computing and Information Systems. His research interests include scalable data mining, network intrusion detection, bioinformatics, and wireless sensor networks.

Shonali Krishnaswamy received the master's degree in computing and the Ph.D. degree in computer science from Monash University, Melbourne, Australia, in 1998 and 2003, respectively. She was the Director of the Centre for Distributed Systems and Software Engineering, Faculty of Information Technology, Monash University, where she is currently an Associate Professor. She is also the Head of the Data Analytics Department, Institute for Infocomm Research, A*STAR, Singapore. Her current research interests include the areas of mobile, ubiquitous, distributed data mining, and data stream mining.

Marimuthu Palaniswami (F'12) received the M.Eng.Sc. degree from The University of Melbourne and the Ph.D. degree from The University of Newcastle, Australia. He is currently a Professor with the Electrical and Electronic Engineering Department, The University of Melbourne, Melbourne, Australia. He leads one of the largest funded ARC Research Network on Intelligent Sensors, Sensor Networks and Information Processing Program to conduct research in the sensor network, Internet of Things (IoT), health, environmental, machine learning, and control areas. His research interests include SVMs, sensors and sensor networks, IoT, machine learning, neural network, pattern recognition, and signal processing and control. He has been a Grants Panel Member for NSF, an Advisory Board Member for European FP6 Grant Center, a Steering Committee Member for NCRIS GBROOS and SEMAT, and a Board Member for IT and SCADA companies. He is representing Australia as a core partner in EU FP7 projects such as SENSEI, SmartSantander, IOT Initiative, and SocIoTal.