Machine Learning for Identifying Group Trajectory Outliers

ASMA BELHADI, Dept. of Technology, Kristiania University College, Oslo, Norway
YOUCEF DJENOURI, Dept. of Mathematics and Cybernetics, SINTEF Digital, Oslo, Norway
DJAMEL DJENOURI, Computer Science Research Centre, Department of Computer Science & Creative Technologies, University of the West of England, Bristol, UK
TOMASZ MICHALAK, Dept. of Computer Science, Warsaw University, Warsaw, Poland
JERRY CHUN-WEI LIN, Dept. of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences, Bergen, Norway

Prior works on the trajectory outlier detection problem solely consider *individual* outliers. However, in real-world scenarios, trajectory outliers can often appear in groups, e.g., a group of bikes that deviates to the usual trajectory due to the maintenance of streets in the context of intelligent transportation. The current paper considers the *Group Trajectory Outlier* (GTO) problem and proposes three algorithms. The first and the second algorithms are extensions of the well-known DBSCAN and *k*NN algorithms, while the third one models the GTO problem as a feature selection problem. Furthermore, two different enhancements for the proposed algorithms are proposed. The first one is based on ensemble learning and computational intelligence, which allows for merging algorithms' outputs to possibly improve the final result. The second is a general high-performance computing framework that deals with big trajectory databases, which we used for a GPU-based implementation. Experimental results on different real trajectory databases show the scalability of the proposed approaches.

CCS Concepts: • **Information systems → Data mining**; • **Security and privacy → Intrusion/anomaly detection and malware mitigation**; • **Computing methodologies → Machine learning**;

Additional Key Words and Phrases: Group trajectory outliers, machine learning, data mining

Authors' addresses: A. Belhadi, Dept. of Technology, Kristiania University College, Oslo, Norway; email: asma.belhadi@kristiania.no; Y. Djenouri, Dept. of Mathematics and Cybernetics, SINTEF Digital, Oslo, Norway; email: youcef.djenouri@sintef.no; D. Djenouri, Computer Science Research Centre, Department of Computer Science & Creative Technologies, University of the West of England, Bristol, UK; email: djamel.djenouri@uwe.ac.uk; T. Michalak, Dept. of Computer Science, Warsaw University, Warsaw, Poland; email: tpm@mimuw.edu.pl; J. C.-W. Lin, Dept. of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences, Bergen, Norway; email: jerrylin@ieee.org.

1 INTRODUCTION

Outlier detection problem involves detecting and, when appropriate, removing anomalous observations from data. This problem emerges in numerous applications [7, 25, 75]. The *group outlier* (or *anomaly*) *detection* is general form of this problem, in which the aim is to identify sets of anomalous observations rather than only individual ones [6, 38]. Typical applications include locating unusual clusters of celestial objects from image processing [6], astronomical data [68], machine monitoring [23], article and web site management [28], etc. This work considers *trajectory outlier detection*— the outlier detection problem for spatial data that involves trajectories. This includes, for instance, vehicle positioning data, hurricane tracking data, and animal movement data. The aim is to identify trajectories that do not conform with the others [44].

The interest in this problem has been especially boosted by the proliferation of GPS-enabled devices and ubiquitous wireless sensor technologies [27, 29] that produce countless trajectories.

The identification of outliers in such data is important to optimize routes in the short term, e.g. in car navigation systems, or to make long-term decisions, such as improving the organization of an urban area [39, 72]. Prior works on the trajectory outlier detection problem consider solely *individual* outliers (see the next section for an overview). However, in real-world scenarios, trajectory outliers may appear in groups. For instance, a group of vehicles that deviates from a usual trajectory due to the maintenance of streets in the context of intelligent transportation, or a group of hurricane trajectories that deviates from the normal ones in the context of climate change consequences [32]. Considering the example of taxi trajectories, where each trajectory is mapped to the road map network in the smart city. Traditional trajectory outlier detection algorithms [44, 69, 71] cannot identify outliers presented by group of taxis deviating from the usual trajectory. Detecting these trajectory outliers could help the planners to study the different correlations between these trajectories to refer useful information. For instance, group of taxi trajectory outliers may indicate that the taxis are partners in a taxi fraud.

This paper deals with the *Group Trajectory Outlier detection* (GTO) problem and explores different solutions based on data mining, machine learning and optimization. First, the use of data mining is investigated by adopting the clustering, and the neighborhood computation techniques in identifying the group of trajectory outliers. Machine learning techniques (including ensemble learning) are then used to treat GTO as a feature selection problem. Further optimization methods are explored to enhance the solutions, including computational intelligence, and high performance computing,.This helps to deal with big trajectory databases in realtime, while ensuring the accuracy. Its main contributions can be summarized as follows:

- An adaptation of the *DBSCAN* algorithm [15] and *k*NN [52] are developed for solving the *GTO* problem (*DBSCAN-GTO* and *k*NN-GTO, respectively). In particular, the DBSCAN-GTO algorithm starts by applying the same method as the *DBSCAN* algorithm to derive the *micro clusters*. These micro clusters are then considered as potential candidates for which a pruning strategy based on density computation measure is proposed. The pruning produces eventual groups trajectory outliers. Similarly, *k*NN-GTO algorithm starts by recursively deriving the trajectory candidates from the individual trajectory outliers, and then prunes these candidates using the proposed density computation measure;
- A novel algorithm, *FS-GTO* that models the GTO problem as feature selection problem is proposed. In particular, the set of individual trajectory outliers are considered as the set of all features, and the feature selection process is adopted to identify the group of trajectory outliers;
- Two different enhancements are proposed for these algorithms by incorporating ensemble learning, computational intelligence, and high performance computing (HPC). The ensemble learning aims the accuracy, and the HPC the computational processing time, which will

enable solving big trajectory databases. An instantiation of the HPC framework is developed using GPU architecture.

- The performance of the proposed algorithms is analyzed using different real trajectory databases. To the best of our knowledge, this is the first work that explores group trajectory outlier detection in the current literature. The only option we have is thus to compare the proposed solutions with the existing general group outlier detection solutions (see Section 2 for an overview) when applied on trajectory data. The results of the experiments demonstrate that the proposed algorithms clearly outperform the baseline algorithms. The experiments show the scalability of the three approaches, as well as the ability of the HPC implementation to efficiently deal with large trajectory databases.

The remainder of the paper is organized as follows. Section 2 reviews the main existing trajectory and group outlier detection algorithms. The problem is formally defined in Section 3. Section 4 presents the proposed algorithms for tackling the GTO problem, *DBSCAN-GTOD*, *k*NN-GTOD and *FS-GTOD*. Section 5 presents the performance evaluation, while Section 6 discusses the learned lessons and draw directions from this work. Finally, Section 7 concludes the paper,

2 RELATED WORK

In this section we briefly discuss two main bodies of related literature: 1) on trajectory outlier detection and, 2) on group outlier detection.

Trajectory outlier detection: Lee et al. [33] dealt with the *angular* sub-trajectory outlier detection problem, where the direction of anomalous sub-trajectories differ from those of neighboring sub-trajectories. Their algorithm uses the *partition and detect* strategy, i.e., each trajectory in the set of all trajectories is partitioned into different line segments called t-partitions. Each detection step is performed by computing the adjusting coefficient of each t-partition for a given trajectory. If this value is greater than 1, i.e., more adjustment is needed, and therefore the t-partition is considered an outlier. Ge et al. [19] introduced a direction-based trajectory outliers in a robotic environment. The monitoring area is partitioned into grids, each of which is further divided into eight directions.The outlying score of each trajectory is calculated by summing the density values of each direction in each grid passed by this trajectory. Only trajectories having score less than a given threshold are considered as outliers. Liu et al. [41] proposed the exploitation of a stochastic model for context-aware anomaly detection in indoor location traces by using the hospital work flow, and considering the movements of medical devices as transitions in finite state machines. Kong et al. [31] proposed a long-term public transport traffic anomaly detection approach. This method first creates a database from both the bus trajectory and the bus station line data. It then calculates the anomaly index score of each road segment, based on its traffic density. Zhu et al. [80] proposed Time-dependent Popular Routes based trajectory Outlier detection. Their solution first identifies the popular routes for each time interval. The trajectories database is then divided into groups so that each group contains trajectories with the same source points and the same destination points in a given time interval. The score of the representative trajectory of each group is determined by comparing its similarity with the top popular roads. Yu et al. [69, 71] find sub-trajectory outliers within a given time window by considering the neighbours of individual points and sub-trajectories. In the work by Wu et al. [65], the set of historical trajectories is first matched to the road network of the city according to the source and destination points. The probabilistic learning model described by the maximum entropy inverse reinforcement [81] is then used to transform the mapped trajectories into historical action trajectories. Thus, each road segment is regarded as a state, the different road decisions, such as turning left, turning right, or moving straight forward, are regarded as actions, and the drivers are considered agents. Afterwards, the learning model is launched to estimate the cost of historical trajectories. For a new sub-trajectory, its probability is computed based on the set of action historical trajectories. If the probability value is greater than a probability threshold then it is an outlier. In the work by Mao et al. [45], the set of trajectory fragments are derived. Each fragment of the given trajectory is composed by a line segment of two consecutive points. The local outlier factor algorithm is used to determine the fragment outliers, where the local difference density is used rather than the local reachability density. This process is repeated for all fragments in all trajectories.

In the domain of mobility analysis, trajectories have been largely studied. Fu et al. [18] explored trajectory abnormalities, and developed a general collective learning approach to understand the heterogeneous human mobility data towards identifying and quantifying the urban forms of residential communities. Liu et al. [42] first provided an integrated mobility pattern framework among the taxicabs and the bus transactions, and then developed a localized transportation model to predict the bus travel demand in a smart city environment. Wang et al. [63, 64] proposed a hybrid mixture of hawkes process and hierarchical topic model to identify abnormal trips. The origin and destination regions of trajectories are augmented by connecting to the neighborhood points of interests. A latent probabilistic based model is then performed to represent the mixed functionality of the trajectories.

Group outlier detection: Some solutions employ statistical models to derive the group of outliers [38, 58, 68, 70]. In particular, Chalapathy et al. [6] considered the use of a deep generative model and test it on various image applications. The outlierness for each group in the input data is estimated by group reference function using a standard back-propogation algorithm. Liang et al. [68] used flexible genre model to find group outliers, where their approach is based on topic modeling, where the inference is performed by gibbs sampling [20], and the learning is done by monte carlo [5]. Das et al. [9] considered the different correlation between the data outliers to detect pattern anomalous by investigating bayesian network anomaly detection [48], and conditional anomaly detection [8]. Thus, the correlation score between the individual outliers is determined by the probability of possible values of these outliers in the training data. Tang et al. [56] defined contextual outlier detection as small group of points that share similarity, on some attributes, with a significantly larger reference group of data, but deviates dramatically on some other attributes. In order to avoid enumerating all contextual outliers, they only maintain the closure context outliers. In addition, only contextual outliers with a statistical significance test greater than a given threshold are retrieved. Other works which explore contextual outlier detection include [37, 76]. [37] assigns feature weights on each group outlier, and compute chain rule entropy to determine correlation between different feature groups. [76] designs a parallel computing solution to deal with contextual outlier detection in high and sparse dimensional space. Xiong et al. [67] studied detecting two kinds of group anomalies: a group of individual anomalous points, and a set of normal points the distribution of whom as a group is abnormal. The authors define a mixture of gaussian mixture model by adopting the likelihood of each group, the marginal likelihood of each observation within a group, and the maximum likelihood estimation to learn the hyperparameters of the mixture model. An application of this algorithm in social media analysis was investigated in [70] by taking into account the dynamic properties of the social media data. Masumoto et al. [46] proposed a new algorithm to detect both local and global outliers from fashion data. It incorporates the local outlier factor algorithm in identifying the top data slices that generate local outlier results of automatically generated OLAP queries. Camacho et al. [4] developed an intrusion detection, which adopts the principal component analysis, and introduce the group-wise concept. The proposed framework is a simple and easy to understand by security professionals not trained in multivariate tools. Fan et al. [16] suggested a hybrid unsupervised model based on convolutional auto-encoder and gaussian process regression to idnetify group outlier detection from high dimensional data. Wu et al. [66] considered the group anomaly detection as a one-class classification problem, and introduce a fault-attention generative probabilistic adversarial autoencoder to automatically retrieve the low-dimensional manifold embedded in high-dimensional space of the signal. Other approaches use clustering strategies on the individual of outliers to group these outliers into similar clusters [9, 54, 56]. Each cluster is then considered as a group of outliers. Soleimani et al. [54] proposed supervised learning approach that groups anomalous patterns when memberships are previously unknown. The salient features are extracted from an appropriate training set with discrete data inputs. It implements a nonparametric bootstrap sampling procedure to evaluate the statistical significance of a detected anomalous behavior for a single object as well as a cluster of objects. The approach is applied on topic documents modeling and it is able to discover irregular topic mixtures from a collection of documents. Sun et al. [55] proposed abnormal group-based joint medical fraud approach. The abnormal group problem was converted to the maximal clique enumeration problem [47] by considering the set of patients as the set of vertices, and each edge indicates that the two connected patients are similar. Note that, the similarity between patients is determined by computing their identical joint behaviors. Maximal clique enumeration is NP-hard problem, to do such task efficiently different partition strategies [13] have been investigated to reduce the graph size. As a result, each maximal clique is considered as abnormal group of patients.

Discussion: As can be seen from the above short literature overview, existing solutions for trajectory data focus on discovering individual outliers. Typically, off-line processing is used to discover whole trajectories while on-line one for sub-trajectories. As for the existing group outlier detection algorithms they are not dedicated to trajectory data. Furthermore, they focus on finding a group outliers from the set of candidate groups, and not from the individual outliers. In contrast, this work is the first dedicated to detect group trajectory outliers from individual trajectory outliers.

3 PROBLEM STATEMENT

Some preliminary definitions are needed to introduce the group trajectory outlier problem. A trajectory is a sequence of location points in space. In the remaining of the paper pt denotes a single spatial location point, where each pt is a tuple of two values—the latitude and the longitude of this location.

Definition 3.1 (Trajectory Database). A trajectory database $T = \{T_1, T_2, \ldots, T_m\}$ is a database in which each raw trajectory T_i is a sequence of spatial location points $\{pt_{i1}, pt_{i2}, \ldots, pt_{in}\}$.

As common in the literature [12], the location points which are similar enough are aggregated into regions. Let us denote by R a location *region* in space.

Definition 3.2 (Mapped Trajectory Database). In a mapped trajectory database $\Lambda = \{\Lambda_1, \Lambda_2 \ldots \Lambda_m\}$, each mapped trajectory Λ_i is a sequence of spatial location regions $\{R_{i1}, R_{i2} \ldots R_{in}\}$, obtained by replacing each point pt_{ik} in T_i with its region R_{ik}.

In the literature, several similarity measures have been used for trajectories data such as longest common subsequence, dynamic time warping, and edit distance [57]. However, all these measures are point-based, it means that the computation of the similarity is based on the points of the trajectories. In real applications, trajectories have different points but may belong to the same region. In the following, a new measure to determine the similarity between trajectories is introduced.

Definition 3.3 (Trajectory Similarity). The similarity between two trajectories Λ_i, and Λ_j, denoted $d(\Lambda_i, \Lambda_j)$, is defined by the number of all regions in the two trajectories minus the number of shared regions in the two trajectories. Formally:

$$d(\Lambda_i, \Lambda_j) = n - \{|(R_{il}, R_{jl})| R_{il} = R_{jl}, \forall l \in [1..n]\} \tag{1}$$

Note that the trajectories do not necessary need to have the same length, it means trajectories may contain different number of regions. To deal with this issue, trajectories with less number of regions are completed by an empty regions called *null*.

Trajectory candidates form the set of potential trajectories belong to a group of trajectory outliers. These trajectory candidates are retrieved from the individual trajectory outliers.

Definition 3.4 (Trajectory Candidates). The set of the first p individual trajectory outliers, i.e., $\mathcal{G}^+_{\mathcal{A}} = \{\Lambda^+_1, \Lambda^+_2 ... \Lambda^+_p\}$ (where $\mathcal{A}$ is a given trajectory outlier detection algorithm) is defined as follows:

$$\mathcal{G}^+_{\mathcal{A}} = \{\Lambda^+_i | \forall j \in \Lambda \setminus \mathcal{G}^+_{\mathcal{A}}, Score(\Lambda_i, \mathcal{A}) \geq Score(\Lambda_j, \mathcal{A})\}, \tag{2}$$

where Score($\bullet$, $\mathcal{A}$) is the ranking function used by $\mathcal{A}$.

The density of a group is an important concept in our analysis. Intuitively, it is defined as the number of shared regions between all the trajectories of the group.

Definition 3.5 (Group Density). We define the density of the candidate group of trajectory outliers, $\mathcal{G}$, as

$$Density(\mathcal{G}) = |\{R | \forall \Lambda_i \in \mathcal{G}, R \in \Lambda_i\}| \tag{3}$$

The group of trajectory outliers is defined as individual of trajectory outliers which share a certain number of regions. An intuition behind this definition is that these trajectories are individual outliers, it means that these trajectories deviate from the normal trajectories. If these trajectories are close from each other, they can form a group of trajectory outliers. In the following, the concept of a group of trajectory outliers is formally defined.

Definition 3.6 (Group of Trajectory Outliers). Set of trajectories $\mathcal{G}$ is said to be a group of trajectory outliers iff

$$\begin{cases} \mathcal{G} \subseteq \mathcal{G}^+_{\mathcal{A}} \\ Density(\mathcal{G}) \geq \gamma, \end{cases} \tag{4}$$

where that $\gamma \in [1, \dots, n]$ is the density threshold.

Definition 3.7 (Group Trajectory Outlier Problem). Group Trajectory Outlier Problem aims to discover from the set of all individual trajectory outliers, the set of all groups of trajectory outliers, denoted by $\mathcal{G}^*$.

Example 3.8. Let us assume that the plane in Figure 1 is a map with 16 different regions and the lines represent 15 mapped trajectories. The map may be viewed as matrix where the element in i^{th} row and j^{th} column represents the region $R_{((i-1)*j)+j}$. Let us denote by Λ_1, Λ_2, and Λ_3 trajectories represented by dashed lines. This is the set of the trajectory candidates, $\mathcal{G}^+_{\mathcal{A}} = \{\Lambda^+_1, \Lambda^+_2, \Lambda^+_3\}$. If we consider two candidate groups $\mathcal{G}_1 = \{\Lambda_2, \Lambda_3\}$, and $\mathcal{G}_2 = \{\Lambda_1, \Lambda_2, \Lambda_3\}$, $Density(\mathcal{G}_1) = 4$ by sharing $\{R_5, R_6, R_7, R_8\}$, and $Density(\mathcal{G}_2) = 3$, by sharing $\{R_6, R_7, R_8\}$. If $\gamma > 3$ only $\mathcal{G}_1$ is a group of trajectory outliers, otherwise, both $\mathcal{G}_1$ and $\mathcal{G}_2$ are group of trajectory outliers.

A straightforward process of detecting groups of trajectory outliers would include the following two activities:

- to consider all possible combinations between the individual trajectory outliers; and
- to evaluate each subset separately using Definition 3.5.

Obviously, this method does not scale (its complexity is $O(2^{|\mathcal{G}^+_{\mathcal{A}}|})$). Therefore, we propose in the next section a general framework and within a number of different dedicated solutions that significantly improve upon the above process of detecting groups of trajectory outliers.

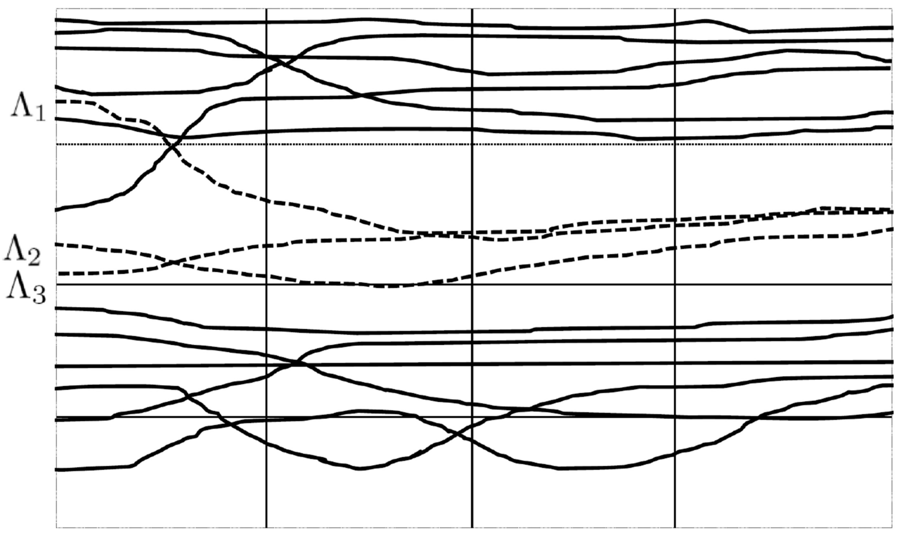


Fig. 1. An example of a group of trajectory outliers: Λ_1, Λ_2, and Λ_3 are represented by the dashed lines.

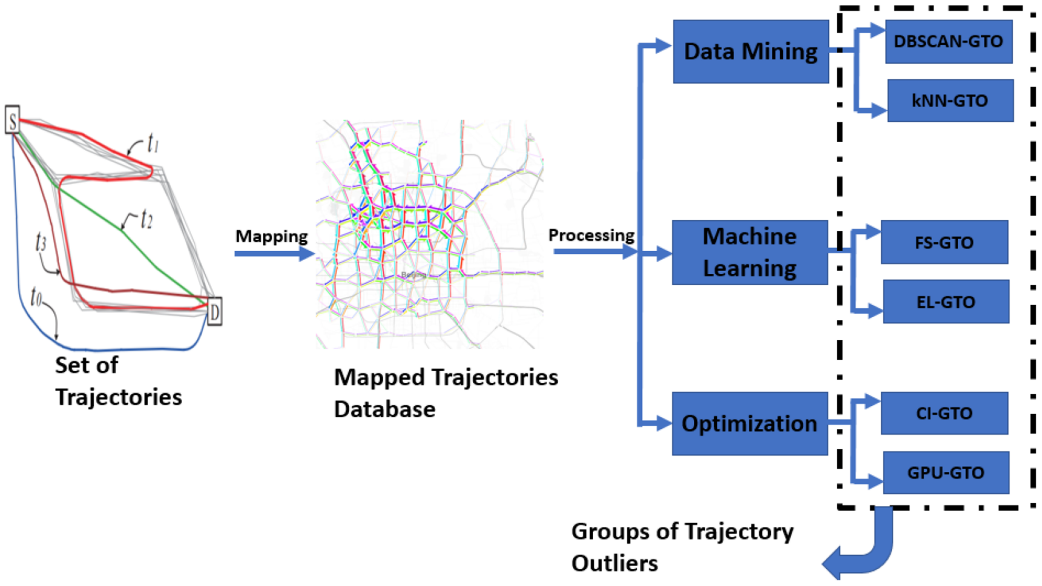


Fig. 2. The solution schema for the GTO problem.

4 PROPOSED SOLUTION & ALGORITHMS

Let us begin by describing the key element of the solution to the GTO problem (see Figure 2 for an overview). Generally speaking, our approach builds upon machine learning and high performance computing. In particular, several machine learning techniques are used when developing the proposed algorithms, including clustering, feature selection, neighborhood computation. The algorithms' performance is further improved by applying ensemble learning, and computational intelligence. Finally, HPC is used to further boost the runtime performance and deal with big trajectory databases in a reasonable time.

As shown in Figure 2, the approach can be divided into:

1. **Mapping (Pre-processing):** Typically, trajectories in most applications consist of noisy GPS data points where errors can exceed several meters. This can negatively influence the final output of many algorithms. Hence, first a map-matching stepshould be used to project GPS data points of each trajectory onto a road network. Several approaches have been developed to achieve this [3, 21, 40]. Since in this work considers sparse trajectory databases, probabilistic model based on a Hidden Markov Model [21, 51] are used. In particular, we represent each road segment as hidden state in the Markov chain—it has an emission probability which is the likelihood of observing the GPS point conditional on the candidate road segment being the true match. A higher probability to a road segment is assigned if the observed trajectory points are close to it. The maximum likelihood path over the Markov chain that has highest probability is then determined, and the corresponding road segment is associated to the observed trajectory point.This way the mapped trajectory database is created, in which every observed trajectory is assigned to the associated road segment.
2. **Processing:** After constructing the mapped trajectory database, a processing step is performed to find out the group of trajectory outliers. In this context, we propose two ways to find group of trajectory outliers and investigate a few approaches including clustering, neighborhood computation, and feature selection. The first way is to start by determining the individual trajectory outliers and then find out the group of trajectory outliers. Another way is to derive directly the group of trajectory outliers from the mapped trajectory database. Further improvements are presented, using incorporating ensemble learning and HPC.

In the remainder of this section describes in details the proposed algorithms (Sections 4.1-4.3), ensemble learning (Section 4.4), and the HPC framework (Section 4.6).

4.1 The DBSCAN-GTO Algorithm

To present the adaptation of the DBSCAN algorithm [15], the following three definitions are needed:

Definition 4.1 (Trajectory Neighborhoods). The neighborhoods of a trajectory Λ_i, $\mathcal{N}_{\Lambda_i}$, for a given threshold ϵ is defined by

$$\mathcal{N}_{\Lambda_i} = \{\Lambda_j | d(\Lambda_i \Lambda_j) \leq \epsilon \vee j \neq i\}. \tag{5}$$

Definition 4.2 (Core T rajectory). A trajectory Λ_i is defined as a core trajectory if there is at least a minimum number of trajectories *MinPts* such that $|\mathcal{N}_{\Lambda_i}| \leq MinPts$.

Definition 4.3 (Micro C luster). A cluster of trajectories C_i is defined as a micro cluster if and only if $0 < |C_i| \leq \mu$, where μ is a user threshold.

In general, solutions to trajectory clustering [34, 35] are able to derive clusters with different densities. However, these algorithms do not explore the *micro clusters* property for anomaly detection. This section presents the proposed approach for identifying group of trajectory outliers, *DBSCAN-GTO*, that uses the DBSCAN algorithm to search for clusters by checking the ϵ-neighborhood of each trajectory (See Definition 4.1). The core trajectories are determined using Definition 4.2. DBSCAN-GTO then iteratively collects density-reachable trajectories from these core trajectories directly, which may involve merging a few density-reachable clusters. The process terminates when no new trajectories can be added to any cluster. Initially, the set of trajectories are grouped (as in DBSCAN). This generates several clusters with different s izes. E ach m icro c luster (See Definition 4.3) is considered as group candidates. For each group, the density of each group is determined using Definition 3.5, if the density exceeds γ threshold, then the group is selected as outlier.

4.2 The kNN-GTO Algorithm

Let us begin the presentation of the adaptation of the kNN algorithm [52], with the following definition:

Definition 4.4 (kNN). kNN of a trajectory Λ_i, denoted by $k\text{NN}(\Lambda_i)$, is defined as

$$k\text{NN}(\Lambda_i) = \{\Lambda_j \in \Lambda \setminus \{\Lambda_i\} | d(\Lambda_i, \Lambda_j) \leq k_{dist}(\Lambda_i)\}, \tag{6}$$

where $k_{dist}(\Lambda_i) = d(\Lambda_i, \Lambda_l)$ is the k-distance of the trajectory Λ_i defined such as it exists k trajectories $\Lambda' \in \Lambda$, it holds that $d(\Lambda_i, \Lambda_l) \geq d(\Lambda_i, \Lambda')$.

The following proposition holds:

Proposition 4.5. *Let us consider two trajectories Λ' and Λ''. Let $\mathcal{G}^*(t)$ be a group of trajectory outliers at the iteration* t *such that:*

$$\Lambda' \in \bigcup_{\Lambda_i^* \in \mathcal{G}^*(t)} kNN(\Lambda_i^*) \wedge \Lambda'' \notin \bigcup_{\Lambda_i^* \in \mathcal{G}^*(t)} kNN(\Lambda_i^*).$$

Then, the following holds:

$$\Lambda' \notin \mathcal{G}^*(t+1) \Rightarrow \Lambda'' \notin \mathcal{G}^*(t+1).$$

Proof. Given that:

$$\Lambda' \in \bigcup_{\Lambda_i^* \in \mathcal{G}^*(t)} k\text{NN}(\Lambda_i^*) \wedge \Lambda'' \notin \bigcup_{\Lambda_i^* \in \mathcal{G}^*(t)} k\text{NN}(\Lambda_i^*)$$

$$\Rightarrow Density(\mathcal{G}^*(t) \cup \{\lambda''\}) \leq Density(\mathcal{G}^*(t) \cup \{\lambda'\}) \ldots \tag{7}$$

$$\Lambda' \notin \mathcal{G}^*(t+1) \Rightarrow Density(\mathcal{G}^*(t) \cup \{\lambda'\}) \leq \gamma \ldots \tag{8}$$

From (7) and (8) it yields: $Density(\mathcal{G}^*(t) \cup \{\lambda''\}) \leq \gamma \Rightarrow \Lambda'' \notin \mathcal{G}^*(t+1)$. □

It follows from the above proposition that if a trajectory Λ_i belongs to the k *nearest neighbors* of at least one trajectory in the current group of trajectory outliers, and Λ_i is not in the group of trajectory outliers of the next iteration, then, any trajectory that belongs to the k *nearest neighbors* of Λ_i could not be in the group of trajectory outliers of the next iteration. Consequently, it is judicious to prune the search into k *nearest neighbors* of the individual trajectory outliers.

kNN-GTO algorithm, an adapted kNN algorithm for identifying group trajectory outliers, considers as input the set of the first p individual trajectory outliers $\mathcal{G}^+ = \{\Lambda_1^+, \Lambda_2^+ \ldots \Lambda_p^+\}$, ranked according to the kNN value, i,e, $\forall i \geq j$, $k\text{NN}(\Lambda_i^+) \geq k\text{NN}(\Lambda_j^+)$. The process aims to enumerate the sets of group trajectory outliers, $\mathcal{G}^*$, by exploring a search tree of $\mathcal{G}^+$. It starts by adding the individual trajectory outlier ranked first, Λ_1^+, to the group trajectory outliers, denoted by $\mathcal{G}_1^*$. It then generates all potential candidates from Λ_1^+. A trajectory t is a potential candidate from Λ_1^+, if and only if, $t \in \mathcal{G}^+ \vee t \in k\text{NN}(\Lambda_1^+)$. The density of $\mathcal{G}_1^*$ is updated by adding the potential candidates to $\mathcal{G}_1^*$, one by one. Only the potential candidates respecting the density threshold are saved, and the remaining ones are removed. Once the potential candidate is added to $\mathcal{G}_1^*$, it is removed from $\mathcal{G}^+$. If $\mathcal{G}_1^*$ contains less than two elements, it is removed from $\mathcal{G}^*$. The same process is recursively applied to all potential candidates added to $\mathcal{G}_1^*$, and the overall process is repeated for all trajectory outliers in $\mathcal{G}^+$.

4.3 FS-GTO

Definition 4.6 (Transformation to FS Problem). Consider GTO problem $\langle \mathcal{G}^{+}_{\mathcal{A}}, \mathcal{G}^{*} \rangle$. This problem is transfomed to the feature selection problem, represented by the set of all features F and the subset of selected features F^{*}, as follows: $F = \mathcal{G}^{+}_{\mathcal{A}}$, and F^{*}=$\mathcal{G}^{*}$. F^{*} is evaluates as follows:

$$Eval(F^{*}) = Quality(F^{*}) - \frac{|F^{*}|}{|F|}, \tag{9}$$

where Quality(F^{*}) is computed as per Definition 3.5.

We consider each individual trajectory outlier as one feature, while the aim is to select the most relevant features from the set of all features. This set then becomes the group of trajectory outliers (see Definition 4.6). The evaluation of the selected set of features (trajectories) is computed using the group density measure, see Equation (9). According to the recent work of Li et al. [36], feature selection algorithms have been categorized into similarity based, information theoretical based, and statistical based methods. In particular, information theoretical based methods explore different heuristics such as information gain [30], Minimum Redundancy Maximum Relevance [50], and Joint Mutual Information [22], These algorithms can only work for supervised learning; while the ground truth is not always available in the problem treated herin. Conversely, statistical based methods calculate different statistical measures such as low variance [53], T-score [11], and Chi-Square Score [79]. These algorithms work only on discrete data, where preprocessing step is required for numerical and continuous data. Finally, similarity based methods determine similarities between features to associate weight importance on each feature candidate. These algorithms of this kind, such as Laplacian Score [24], SPEC [77], Fisher Score [49], and Trace Ratio Criterion [26], proved excellent performance in both supervised and unsupervised scenarios. They are also easy to implement, as the main process focuses on designing an affinity matrix from which the scores of features can be computed. Thus, we chose similarity based methods, and SPEC in particular, as it considers unsupervised learning. In more detail, in SPEC, a feature that is consistent with the data manifold structure should assign similar values to instances that are near each other. The process starts by applying the SPEC algorithm on the set of individual trajectory outliers, The output of this step is a ranking of individual outliers in the descending order in terms of score feature relevance. Using the SPEC ranking vector of individual outliers, a search enumeration tree is generated in the breadth-first-search (BFS) manner. If the quality of the current group candidate does not reach the criteria from Definition 3.5, a backtracking procedure is launched by taking the next trajectory in the SPEC ranking vector. While exploring the enumeration tree of individual outliers, the aim is to maximize the function reported in Equation (9).

4.4 Ensemble Learning

Each of the previously proposed techniques—clustering (*DBSCAN-GTO*), neighborhood computation (*KNN-GTO*) and feature selection (*FS-GTO*)—returns potential groups of trajectory outliers. Out of these groups, some groups are usually good while others may not be useful. In order to improve the accuracy of the detected group of trajectory outliers, we propose the use of ensemble learning [61, 83]. In particular, the proposed algorithms, DBSCAN-GTO, kNN-GTO, and FS-GTO, are launched in this context called *learners*. The three outputs—sets G^{DBSCAN}, G^{kNN}, and G^{FS}—are then merged to derive the final set of groups of trajectory outliers. Hence, the main challenge of this approach is to find an efficient merging strategy. We propose the following one. For each group of trajectory outliers, the number of the occurrences of the three learners is determined, by looking for the groups that are highly frequent. For instance, if there are two groups: the first group $\{\Lambda_1, \Lambda_2, \Lambda_4\}$ appears two times—ones in the output of the G^{DBSCAN} algorithm, and another time in the output of the G^{kNN} algorithm—and the second group $\{\Lambda_1, \Lambda_2, \Lambda_3\}$ appears only one time in the output of G^{FS}, then the first group is better ranked than the second one.

4.5 Computational Intelligence

Evolutionary computing draws ideas from natural evolution such as survival of the fittest, natural selection, reproduction, mutation, competition and symbiosis [14]. The aim of the computational intelligence methods, such as genetic algorithms, genetic programming, mimetic algorithms, immune and swarm intelligence algorithms, is to make an accurate solution for the given optimization problems. In this work, evolutionary computation, and in particular, the genetic algorithm [10] is used to improve the quality of the returned group of trajectory outliers. The initial population is first generated by randomly selecting the group of trajectory outliers from the results obtained by DBSCAN-GTO, kNN-GTO, or FS-GTO. Each chromosome in the initial population will be one potential group of trajectory outliers. Two operators are then used to refine the initial solution: i) The first one is crossover, it takes two chromosomes from the population, and generates two new chromosomes by making intersection operator between the groups selected, and ii) The second one is mutation, it takes two generated chromosomes on the crossover step, and generates two more chromosomes by making the union operator between the groups selected. All generated chromosomes are evaluated by the fitness function using Definition 3.5. The best chromosomes are selected for the next iteration. This process is repeated until a maximum number of iterations are reached.

4.6 High Performance Computing

In this section, a generic approach is first proposed to implement the proposed solutions on parallel architectures. A particular instantiation on GPU architecture of this generic approach is then presented.

Generic Approach to Parallelize GTO Solutions: To run the proposed GTO algorithms on any parallel architecture, the following sequential steps need to be performed:

(1) **Map Partitioning:** In this step, the map is divided into several grids, whereby each grid contains a set of similar trajectories. This step is performed in CPU.
(2) **Computing and storing the local results:** In this step, each parallel node applies one of the GTO solutions on each cluster, generates all group trajectory outliers from the grid that is assigned to it, and stores them in the set of all groups of trajectory outliers. The latter is built following the same logic of building the list of the group of trajectory outliers in the serial implementation of GTO solutions. Once the local group trajectory outliers are calculated, they will be sent to CPU for further processing.
(3) **Merging the local results:** The local group of trajectory outliers are merged into a global one on the CPU side. This can be done using a simple concatenation of all the local results.

GPU-GTO: The instantiation of the three steps defined above must be carefully designed to fit the hardware in use. Here, we present an instantiation of this generic approach using GPU hardware. GPUs (Graphic Processing Units) are graphical cards initially developed for efficient generation of images intended for a display device, but their use as a powerful computing tool has gained popularity in many domains during the last decade [59, 60, 62]. The hardware is composed of two hosts, the CPU one and the GPU one. The former contains processor(s) and the main memory. The latter is a multi-threading system that consists of multiple computing *cores*, where each core executes a block of threads. The threads of a block in the same core communicate with one another using a shared memory, whereas the communication between blocks relies on a global memory. The CPU/GPU communication is made possible by hardware buses. GPU-GTO is our adaptation of GTO for deployment on GPU architectures. In GPU-GTO, the map is first partitioned on k grids $\{g_1, g_2 \dots g_k\}$ using the map partitioning step. The set of designed grids are then sent to GPU. Each block of threads is mapped onto one grid, where the GTO solutions are applied on each block in parallel. Let the size of the shared memory of each block to be denoted sm. The first sm trajectories of the grid g_i are allocated to the shared memory of the block, and the remaining trajectories of the grid g_i is allocated to the global memory of the GPU host. GPU-GTO defines a *local table*, $table_i$, to store the group of trajectory outliers of the grid g_i. The local table of each grid is sent to CPU for further processing. In this context, CPU host performs a merging step to find the global group of trajectory outliers, where the union of all sets of group of trajectory outliers in the local tables is computed. From a theoretical standpoint, GPU-GTO improves the GTO solutions by exploiting the massively threaded computing of GPUs while mining the grids of trajectories. GPU-GTO also minimizes the CPU/GPU communication, by defining only two points of CPU/GPU communication. The first one takes place when the grids are loaded into the GPU host, and the second one when the local tables are returned to the CPU. GPU-GTO also provides an efficient memory management by using different levels of memories including global and shared memories. However, GPU-GTO may suffer from the synchronization between the GPU blocks. This takes place when the GPU blocks process grids with different number of trajectories. This issue degrades the performance of the GPU-based implementation of the GTO solutions. In real scenarios, different number of trajectories per grid may be obtained, this depends to the way of the trajectories are placed into the map, as the size of the grids are different, as the synchronization cost of the GPU-based implementation will be high. All these statements will be clearly explained in the performance evaluation section below.

5 EXPERIMENTAL EVALUATION

In this section, the GTO framework is experimentally evaluated and its different components. In particular, the serial implementations of the different GTO solutions are compared with the state-of-the art group outlier detection algorithms using standard trajectory databases. In addition, the scalability performance of the GPU-based implementation is carried out on big trajectory databases.

Experimental Setup: The implementation of the different components of the GTO framework[1] has been integrated on the SPMF data mining library [17]. The experimental evaluation of the serial implementations has been performed on a computer with $64bit$ core i7 processor running Windows 10 and $16GB$ of RAM. The evaluation of the the GPU-based implementation has been carried out on a CPU host coupled with a GPU device. The CPU host is a 64-bit quad-core Intel Xeon E5520 with a clock speed of $2.27GHz$. The GPU device is an Nvidia Tesla C2075 with 448 CUDA cores (14 multiprocessors with 32 cores each) and a clock speed of $1.15GHz$. It has $2.8GB$ of global memory, $49.15KB$ of shared memory, and a warp size of 32. Both the CPU and GPU are used in single precision.

In general, a common problem of outlier detection techniques is the evaluation procedure. This is particularly the case for new applications such as the GTO problem, where a ground truth is typically unknown. To facilitate a quantitative evaluation, for group trajectory outlier detection techniques, the process of Zhang et al. [74] is adapted to inject synthetic group trajectory outliers. In particular:

- **Injecting individual trajectory outliers:** Individual trajectory outliers are generated by adding noise *several times* with a certain probability $p \sim \mathcal{U}(0.0, 1.0)$ and a given threshold μ.

[1]https://github.com/YousIA/GTOD.

- **Injecting group trajectory outliers:** Group of trajectory outliers are generated by adding noise to the set of individual of trajectory outliers, but now only a *few times* with a certain probability $p \sim \mathcal{U}(0.0, 1.0)$ and a given μ.

Note that $\mathcal{U}(0.0, 1.0)$ is the continuous uniform distribution with lower, and upper bounds set to 0, and 1, respectively. For both injections, each point p_{il} in the trajectory Λ_i is changed as follows:

$$p_{il} = \begin{cases} p_{il} + n \sim \mathcal{N}(0,1) & \text{if } p \geq \mu \\ p_{il} & \text{otherwise.} \end{cases} \tag{10}$$

The evaluation is performed using F-measure, and ROCAUC, which are common measures for the evaluation of outlier detection methods [52].

Data Description: Benchmark trajectory databases from different domains are used, i.e.:

(1) **Intelligent Transportation:** A database from the *ECML PKDD 2015* databases competition[2] is used. The database contains real trajectories retrieved from 01/07/2013 to 30/06/2014 of *442* taxis in the city of Porto, in Portugal. This allows to recuperate more than *3 GB* of data stored in one single CSV file. Each row contains information related to one trip including: *TripID*, *CallType* and *TaxiID*. The last component of the row contains a list of GPS coordinates. This list contains one pair of coordinates for each *15* seconds of trip. The last list item corresponds to the trip's destination while the first one represents its start.
(2) **Climate Change:** The*hurricane track data set* is used, which contains latitude, longitude, maximum sustained surface wind, and minimum sea-level pressure of hurricane trajectories in USA at 64 hourly intervals. The Atlantic hurricanes [32] is used, which was retrieved from the years 1851 to 2018. This data contains 52775 hurricane trajectories.
(3) **Environment:** A database of the *Starkey Project*[3] is used. We consider the animal movement data illustrated by the radio-telemetry locations of *elk*, *deer*, and *cattle* retrieved from 1989 to 1999. The locations are recorded at 30 minute intervals. This data is considered sparse one with 100 trajectories, and more than 40,000 different points.

Moreover, two additional big databases are used in the experiments: i) *taxi 13-1* containing 1.89 million trajectories, and ii) *taxi 13-2* containing 3.69 million trajectories [44].

The results of the experiments are presented in the following.

5.1 Parameters Setting

The first part of this experiment focuses on tuning the parameters of different proposed GTO solutions. As shown in Figures 3, 4, and 5, several tests have been performed by varying the user threshold (from 1 to 10) for DBSCAN-GTO, the number of neighborhood (from 1 to 10) for kNN-GTO, and the tree depth (from 1 to 10) for FS-GTO. In all the trajectory database used as input (Intelligent Transportation, Climate Change or Environment), the accuracy (determined by F-measure and ROC-AUC) of DBSCAN-GTO and kNN-GTO increases as the value of the corresponding parameter grows up to reaching an optimal point. After this, the accuracy starts decreasing except for FS-GTO that converges at this point. The best parameter values obtained in this step are used in the remaining of the experiments. The best values of the proposed solutions for different trajectory databases are as summarized for each dataset as follows:

[2]http://www.geolink.pt/ecmlpkdd2015-challenge/dataset.html.
[3]https://www.fs.fed.us/pnw/starkey/introduction.shtml.

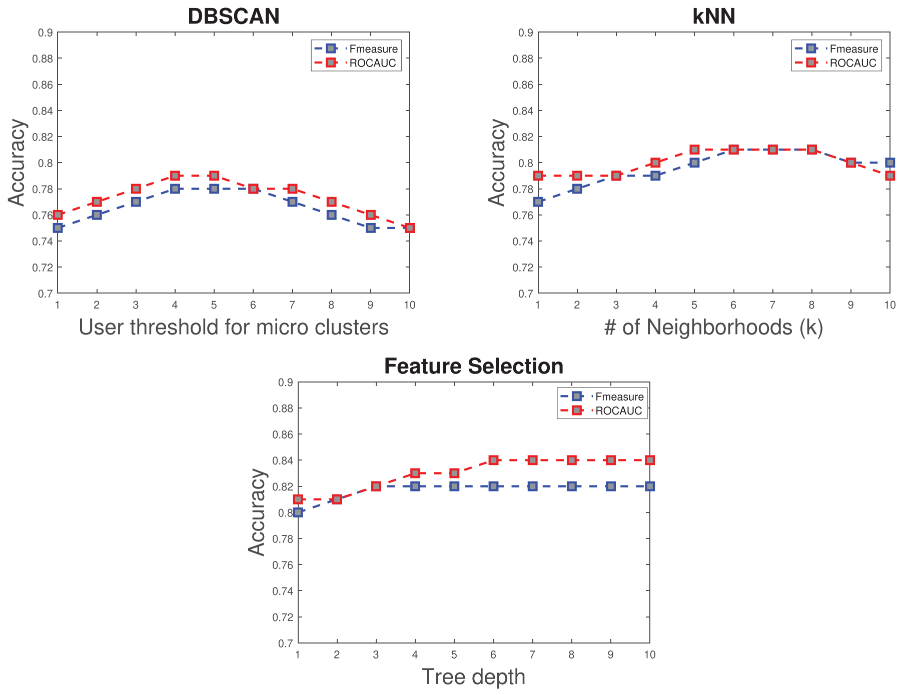


Fig. 3. The parameter setting of the proposed algorithms on Intelligent Transportation.

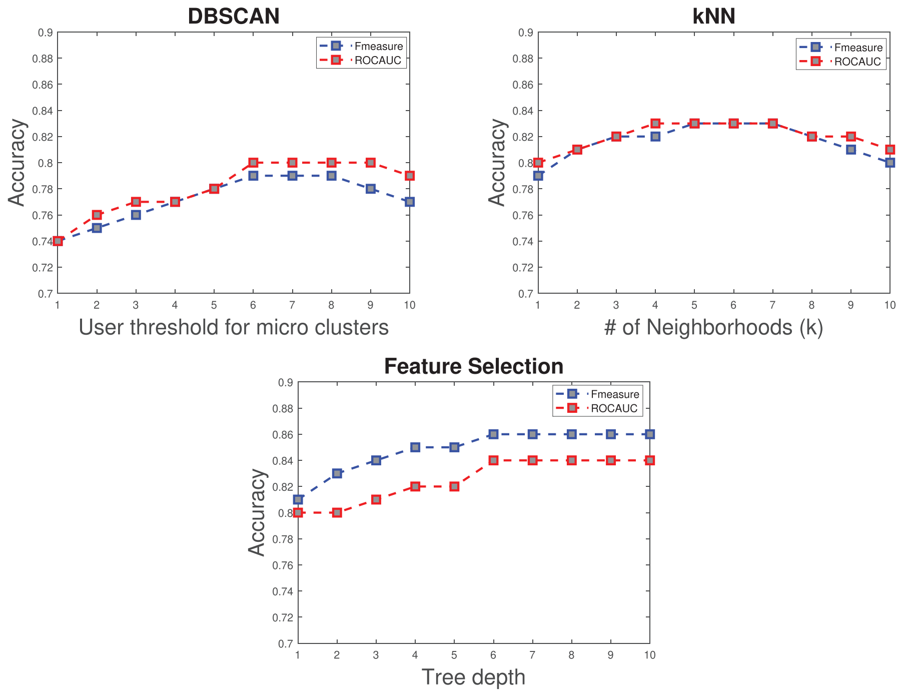


Fig. 4. The parameter setting of the proposed algorithms on Climate Change.

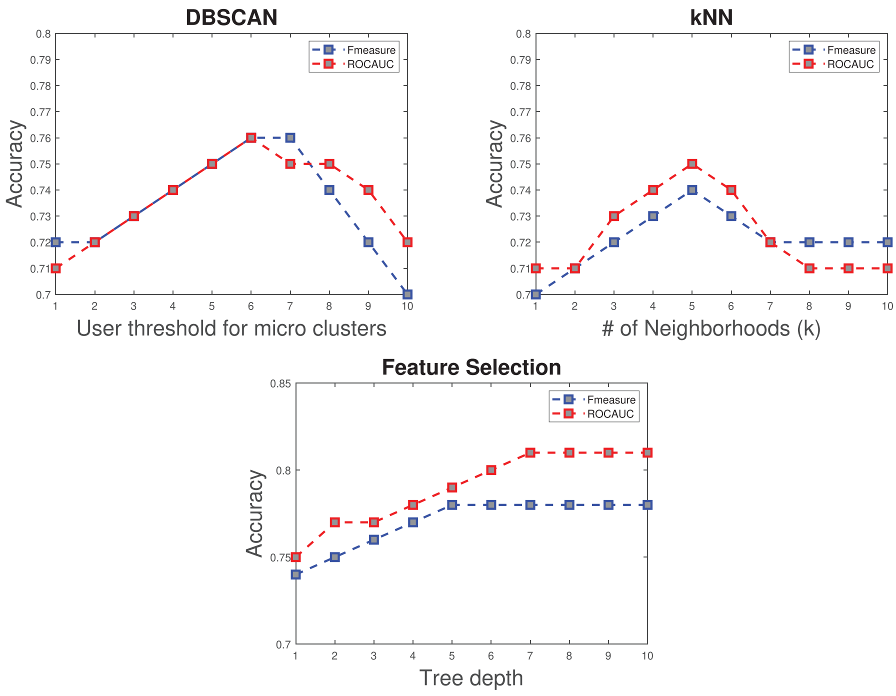
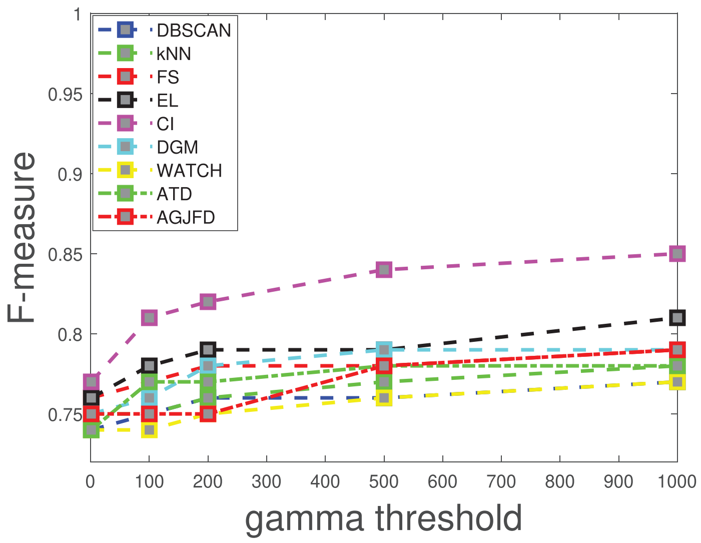


Fig. 5. The parameter setting of the proposed algorithms on Environment.

(1) Intelligent Transportation: user threshold is set to 4 for DBSCAN-GTO, k is set to 6 for kNN-GTO, tree depth is set to 6 for FS-GTO.
(2) Climate Change: user threshold is set to 6 for DBSCAN-GTO, k is set to 5 for kNN-GTO, tree depth is set to 6 for FS-GTO.
(3) Intelligent Transportation: user threshold is set to 6 for DBSCAN-GTO, k is set to 5 for kNN-GTO, tree depth is set to 7 for FS-GTO.

5.2 Performance of Serial GTO Solutions

The aim of this experiment is to compare the proposed solutions with the state-of-the art algorithms in terms of accuracy and processing time. Since, to the best of our knowledge, this is the first work that explores group trajectory outlier detection, the proposed solutions are compared with general group outlier detection solutions (see Section 2). We adopted four well-known algorithms to trajectory data, i.e., DGM [6], WATCH [37], ATD [54], and AGJFD [55]. DGM considers the use of a deep generative model, in which the outlierness is estimated using a standard back-propogation algorithm. WATCH deals with contextual outlier detection in high and sparse dimensional space using feature grouping. ATD is a clustering-based solution that uses a nonparametric bootstrap sampling and topic modeling to identify group outliers. AGJFD uses the maximal clique enumeration problem to identify the abnormal group of patients. Figures 6, 7, 8, and 9 present both accuracy and runtime of the proposed solutions (DBSCAN, kNN, Feature Selection, Ensemble Learning, and Computational Intelligence), and the baseline algorithms (DGM, WATCH, ATD, and AGJFD). For any gamma threshold from 1 to 1000, the solutions based on feature selection, ensemble learning, and computational intelligence methods outperform the baseline algorithms in terms of accuracy. However, solutions based on neighborhood computation and DBSCAN are less competitive. This comes from the fact that the former solutions use more advanced and recent strategies, while the



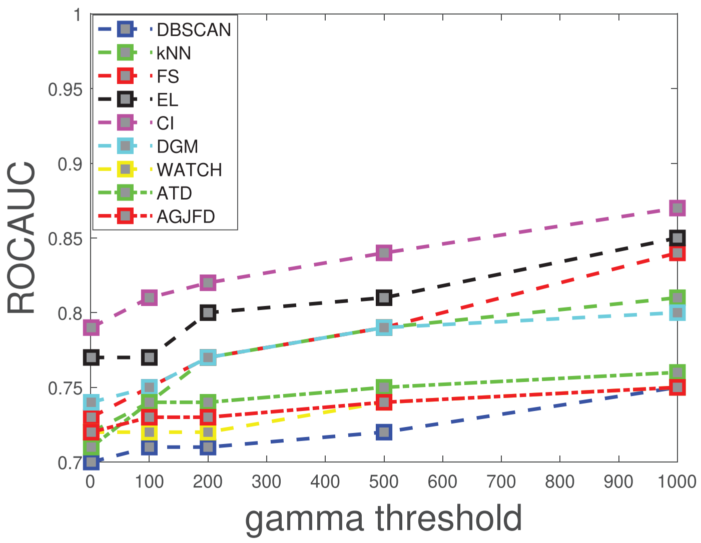
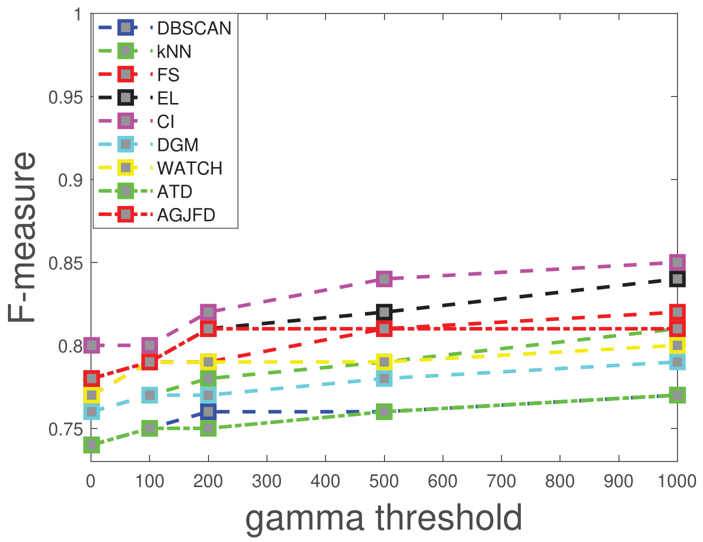


Fig. 6. Accuracy: The Proposed Solutions Vs. State-of-the art Group Detection on Intelligent Transportation.

latter solutions use less advanced concepts. Regarding the time processing, the advanced solutions require more time than the less advanced ones, but they are still competitive in this respect to the baseline algorithms.

5.3 Performance of GPU-GTO Solutions

In this section, the GPU-based solutions are evaluated on big trajectory databases. Figure 10 presents the speedup obtained by the proposed algorithms. The number of blocks has been varied from 128 to 1,024, and the number of threads per block from 1 to 512. The figure show that the speedup increases considerably for the proposed solutions. This is the case, in particular, for computational intelligence method. While, for small number of blocks and small number of threads



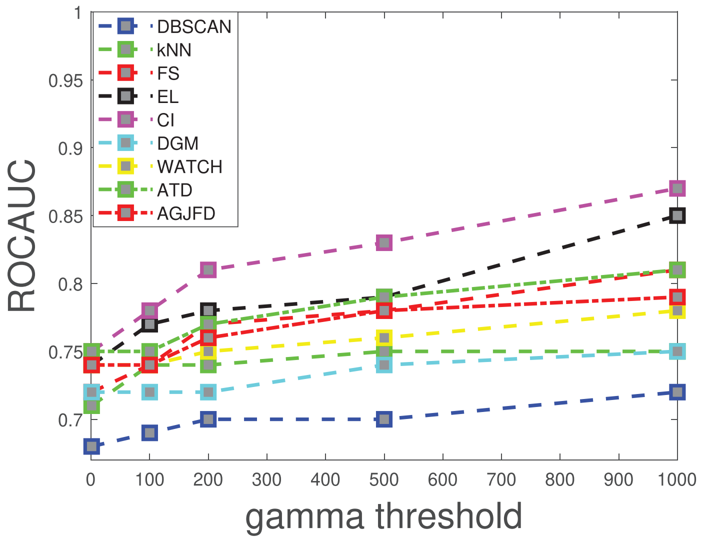
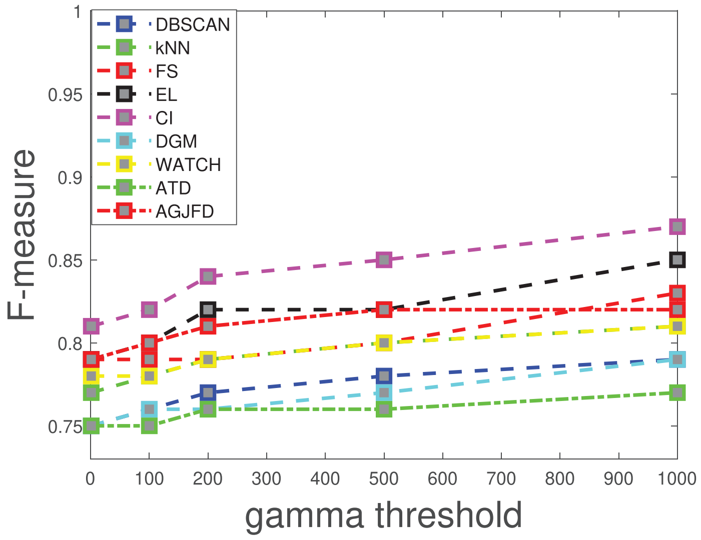


Fig. 7. Accuracy: The Proposed Solutions Vs. State-of-the art Group Detection on Climate Change.

per block, the speedup of the proposed solutions is a bit less than 100, for large number of blocks and large number of threads per block, the GPU-based solutions are 200 times faster than the serial solutions. This means that computational overhead is relatively small, and this is among others due to an efficient partition strategy to map the trajectories to different GPU blocks.

Figure 11 compares the proposed GPU-solution using the computational intelligence with the baseline GPU-based outlier detection algorithms (SolvingSet [?], SolvingSet+ [1], and MoNavGPU [73]), using big trajectory databases. SolvingSet, and SolvingSet+ are GPU-based implementations of neighborhood-based outlier detection algorithm, whereas MoNavGPU is the GPU implementation of MoNav algorithm [43]. By increasing with the percentage of trajectory sizes from 10% to 100%, our solution outperforms the baseline approaches in terms of processing time.



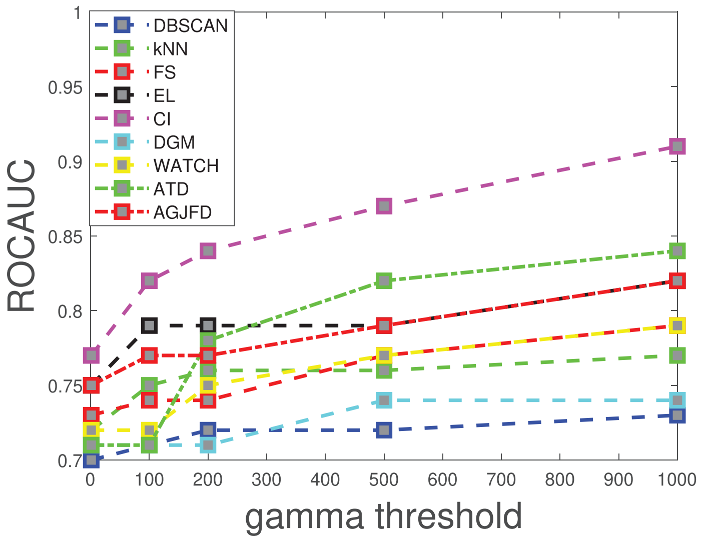


Fig. 8. Accuracy: The Proposed Solutions Vs. State-of-the art Group Detection on Environment.

Its runtime does not exceed 300 seconds to deal with the whole trajectory *taxi 13-2* database, while the other solutions need $x500$ seconds for processing such trajectory database. These speedup is obtained due to the intelligent mapping between trajectories and the GPU blocks, the parallelism by nature of computational intelligence approach.

6 DISCUSSIONS AND FUTURE PERSPECTIVES

The findings from the application of the group of outlier detection to the trajectory data are:

(1) The efficient combination of several concepts from different fields in detecting group of trajectory outliers improves the overall performance (for both quality and runtime) compared to the baseline approaches for group detection. Notably by exploring machine

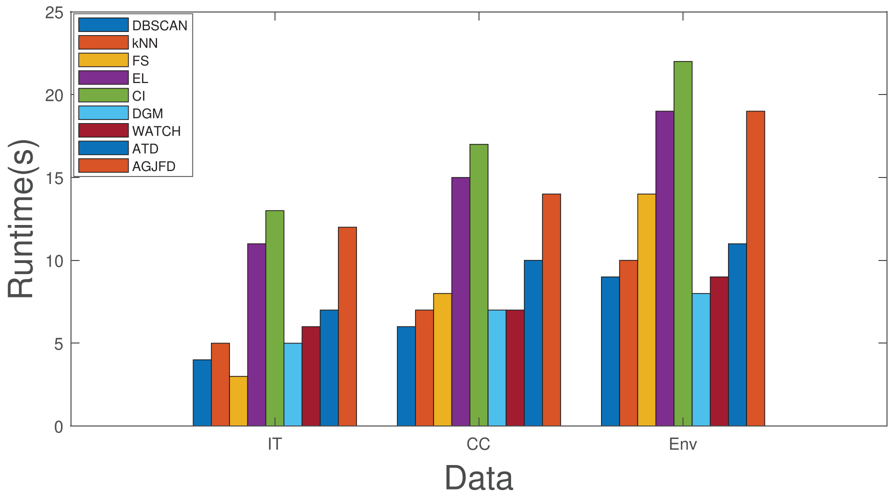


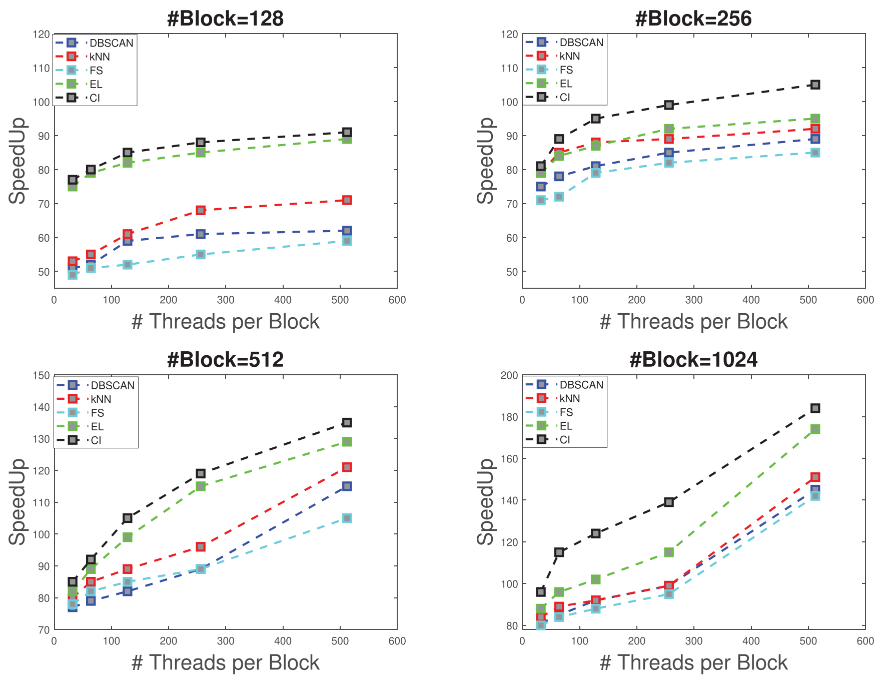
Fig. 9. Runtime: The Proposed Solutions Vs. State-of-the art Group Detection.

learning algorithms (micro clusters, nearest neighbors, feature selection), as well as ensemble learning, computational intelligence and the high performance computing.

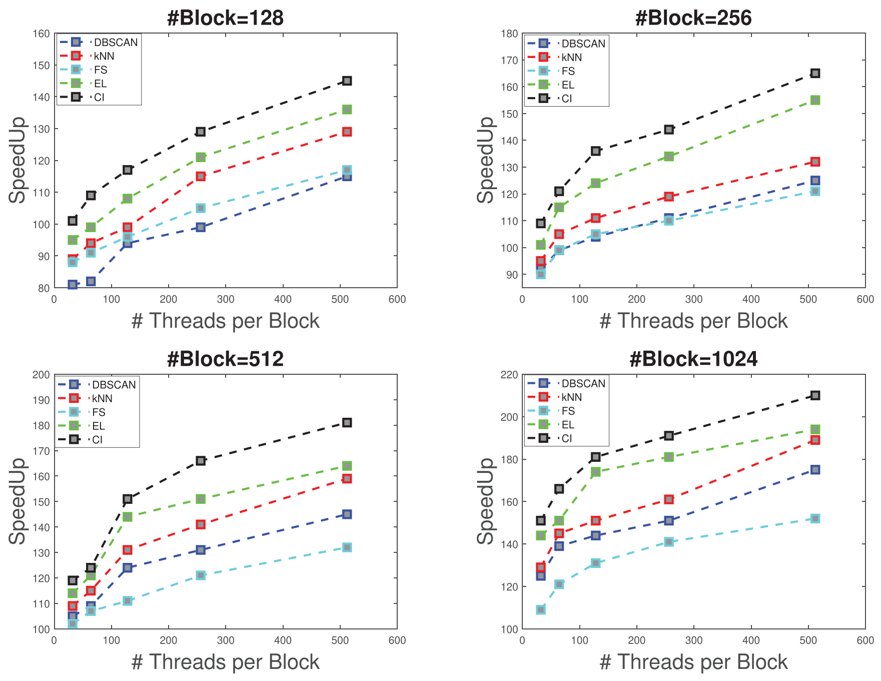
(2) From a machine learning research standpoint, the GTO solutions proposed in this work are examples of the adaptation of generic algorithms to a specific context of trajectory outliers. As in many other cases, porting a pure machine learning technique into a specific application domain requires methodological refinement and adaptation [39, 54]. In our specific context, this adaptation has been implemented by integrating a new model called *group of trajectory outliers*.

This work is just the first milestone of a long path, while much investigation by the machine learning community is required before reaching mature solutions ready to be exploited by city planners in smart city environments. The directions of future work include:

(1) *Techniques for GTO*: other more sophisticated techniques can be developed for *GTO* problem. For instance, other traditional outlier detection techniques may be adopted such as Local Outlier Factor (LOF) [2]. This can be done by developing new concepts of density and local reachability density, designed for the *GTO* problem.
(2) **Missing of the ground truth** It is a common problem in evaluating of trajectory outlier detection algorithms [78, 82]. The following issues and research questions represent challenges for future research on the aspect of quality assessment of group of trajectory outlier detection results:
    (1) Defining useful, publicly available benchmark data for group trajectory outlier detection problem would allow for a more objective research study for analyzing the group of trajectory outlier detection algorithms.
    (2) It would be very useful to identify meaningful criteria of to run an internal evaluation of group trajectory outlier detection. One way to address this challenging issue is to provide unified ranking function scores to rank the group trajectory outliers. These functions should be independent from the whole process of retrieving the group trajectory outliers.
(3) *GTO applications*: more effort is needed to investigating and targeting new applications of *GTO*, such as climate change analysis, for example, finding a group of hurricane



taxi 13-1



taxi 13-2

Fig. 10. The speed of the proposed solutions on the GPU architecture.

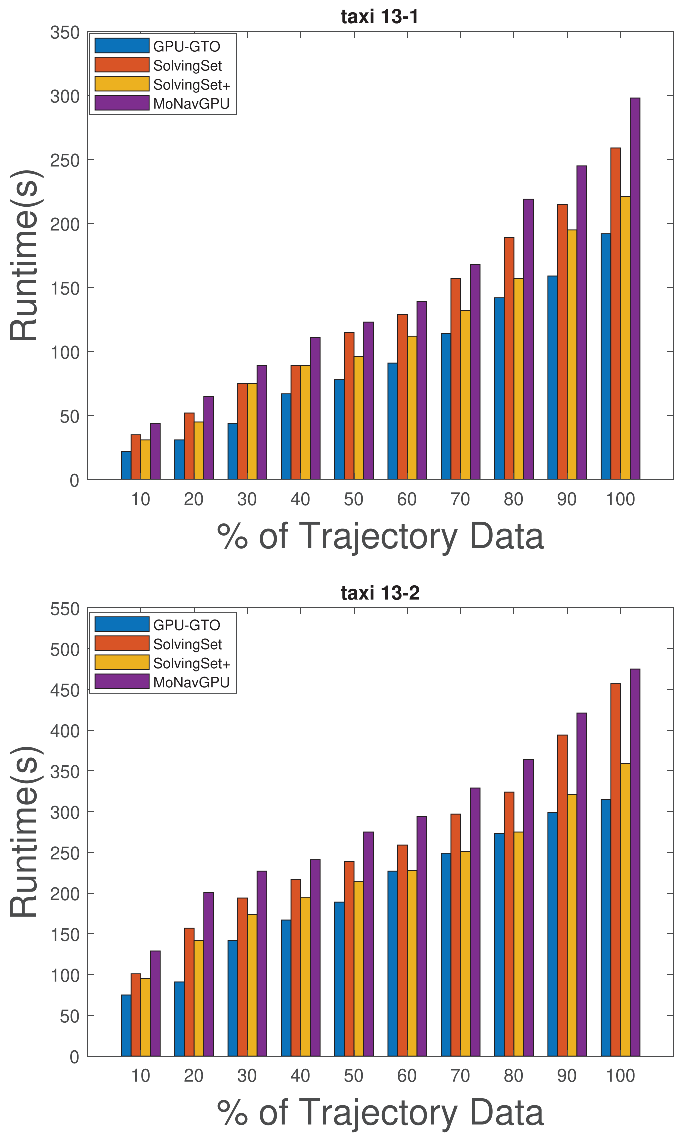


Fig. 11. The Proposed Solutions Vs. State-of-the art GPU-based Outlier Detection.

trajectories that deviates from the normal hurricane ones. This allows to identify other cities that could be affected by the hurricanes. The last hurricanes observed in the *United States* during the period of 2018-2019 is a real example of such study. In addition, new visualization techniques can be developed for *GTO*, in order to present in an accessible way groups of trajectory outliers to the city planners.

7 CONCLUSIONS

The problem of Group Trajectory Outlier (GTO) detection has been dealt with in this paper, which focuses on discovering group of trajectory outliers. This is different from previous trajectory outlier detection approaches, which are only able to derive individual trajectory outliers. The

GTO problem is particularly relevant to smart city applications, where a large volume of data on trajectories is collected daily. In order to solve the new problem efficiently, we have proposed three algorithms, *DBSCAN-GTO*, *k*NN-GTO, and *FS-GTO*. All approaches have been tested on real trajectory databases, and the results demonstrate the usefulness of exploring ensemble learning, computational intelligence, and high performance computing in identifying group trajectory outliers. The advantages of the approaches proposed in this paper over to the baseline group detection has been validated through extensive experimental study. Moreover, the results demonstrate that GPU-parallel approach outperforms the existing HPC approaches when dealing with big trajectory databases.

REFERENCES

[1] Fabrizio Angiulli, Stefano Basta, Stefano Lodi, and Claudio Sartori. 2016. GPU strategies for distance-based outlier detection. *IEEE Transactions on Parallel and Distributed Systems* 27, 11 (2016), 3256–3268.

[2] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. LOF: Identifying density-based local outliers. In *ACM SIGMOD Record*, Vol. 29. 93–104.

[3] Marcus A. Brubaker, Andreas Geiger, and Raquel Urtasun. 2015. Map-based probabilistic visual self-localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38, 4 (2015), 652–665.

[4] José Camacho, Roberto Therón, José M García-Giménez, Gabriel Maciá-Fernández, and Pedro García-Teodoro. 2019. Group-wise principal component analysis for exploratory intrusion detection. *IEEE Access* 7 (2019), 113081–113093.

[5] Gilles Celeux, Didier Chauveau, and Jean Diebolt. 1996. Stochastic versions of the EM algorithm: An experimental study in the mixture case. *Journal of Statistical Computation and Simulation* 55, 4 (1996), 287–314.

[6] Raghavendra Chalapathy, Edward Toth, and Sanjay Chawla. 2018. Group anomaly detection using deep generative models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 173–189.

[7] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)* 41, 3 (2009), 15.

[8] Kaustav Das and Jeff Schneider. 2007. Detecting anomalous records in categorical datasets. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 220–229.

[9] Kaustav Das, Jeff Schneider, and Daniel B. Neill. 2008. Anomaly pattern detection in categorical datasets. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 169–176.

[10] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation* 6, 2 (2002), 182–197.

[11] David Dernoncourt, Blaise Hanczar, and Jean-Daniel Zucker. 2014. Analysis of feature selection stability on high dimension and small sample data. *Computational Statistics & Data Analysis* 71 (2014), 681–693.

[12] Youcef Djenouri, Asma Belhadi, Jerry Chun-Wei Lin, Djamel Djenouri, and Alberto Cano. 2019. A survey on urban traffic anomalies detection algorithms. *IEEE Access* 7 (2019), 12192–12205.

[13] John D. Eblen, Charles A. Phillips, Gary L. Rogers, and Michael A. Langston. 2012. The maximum clique enumeration problem: Algorithms, applications, and implementations. In *BMC Bioinformatics*, Vol. 13. BioMed Central, S5.

[14] Andries P. Engelbrecht. 2007. *Computational Intelligence: An Introduction*. John Wiley & Sons.

[15] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of KDD*. 226–231.

[16] Jinan Fan, Qianru Zhang, Jialei Zhu, Meng Zhang, Zhou Yang, and Hanxiang Cao. 2020. Robust deep auto-encoding Gaussian process regression for unsupervised anomaly detection. *Neurocomputing* 376 (2020), 180–190.

[17] Philippe Fournier-Viger, Antonio Gomariz, Ted Gueniche, Azadeh Soltani, Cheng-Wei Wu, and Vincent S. Tseng. 2014. SPMF: A Java open-source pattern mining library. *The Journal of Machine Learning Research* 15, 1 (2014), 3389–3393.

[18] Yanjie Fu, Guannan Liu, Yong Ge, Pengyang Wang, Hengshu Zhu, Chunxiao Li, and Hui Xiong. 2018. Representing urban forms: A collective learning model with heterogeneous human mobility data. *IEEE Transactions on Knowledge and Data Engineering* 31, 3 (2018), 535–548.

[19] Yong Ge, Hui Xiong, Zhi-hua Zhou, Hasan Ozdemir, Jannite Yu, and Kuo Chu Lee. 2010. Top-eye: Top-k evolving trajectory outlier detection. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. 1733–1736.

[20] Stuart Geman and Donald Geman. 1987. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. In *Readings in Computer Vision*. Elsevier, 564–584.

[21] Chong Yang Goh, Justin Dauwels, Nikola Mitrovic, Muhammad Tayyab Asif, Ali Oran, and Patrick Jaillet. 2012. Online map-matching based on hidden markov model for real-time traffic sensing applications. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 776–781.

[22] Baofeng Guo and Mark S. Nixon. 2008. Gait feature subset selection by mutual information. *IEEE Transactions on Systems, MAN, and Cybernetics-part a: Systems and Humans* 39, 1 (2008), 36–46.

[23] Tom Harris. 1993. A Kohonen SOM based, machine health monitoring system which enables diagnosis of faults not seen in the training set. In *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*, Vol. 1. IEEE, 947–950.

[24] Xiaofei He, Deng Cai, and Partha Niyogi. 2006. Laplacian score for feature selection. In *Advances in Neural Information Processing Systems*. 507–514.

[25] Victoria Hodge and Jim Austin. 2004. A survey of outlier detection methodologies. *Artificial Intelligence Review* 22, 2 (2004), 85–126.

[26] Yi Huang, Dong Xu, and Feiping Nie. 2012. Semi-supervised dimension reduction using trace ratio criterion. *IEEE Transactions on Neural Networks and Learning Systems* 23, 3 (2012), 519–526.

[27] Mohamed Amine Kafi, Yacine Challal, Djamel Djenouri, Abdelmadjid Bouabdallah, Lyes Khelladi, and Nadjib Badache. 2012. A study of wireless sensor network architectures and projects for traffic light monitoring. In *Proceedings of the 3rd International Conference on Ambient Systems, Networks and Technologies (ANT 2012), the 9th International Conference on Mobile Web Information Systems (MobiWIS-2012), Niagara Falls, Ontario, Canada, August 27-29, 2012 (Procedia Computer Science)*, Elhadi M. Shakshuki and Muhammad Younas (Eds.), Vol. 10. Elsevier, 543–552.

[28] Ramakrishnan Kannan, Hyenkyun Woo, Charu C. Aggarwal, and Haesun Park. 2017. Outlier detection for text data. In *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 489–497.

[29] Elmouatezbillah Karbab, Djamel Djenouri, Sahar Boulkaboul, and Antoine B. Bagula. 2015. Car park management with networked wireless sensors and active RFID. In *IEEE International Conference on Electro/Information Technology, EIT 2015, Dekalb, IL, USA, May 21-23, 2015*. IEEE, 373–378. DOI: https://doi.org/10.1109/EIT.2015.7293372

[30] Kenji Kira and Larry A. Rendell. 1992. A practical approach to feature selection. In *Machine Learning Proceedings 1992*. Elsevier, 249–256.

[31] Xiangjie Kong, Ximeng Song, Feng Xia, Haochen Guo, Jinzhong Wang, and Amr Tolba. 2017. LoTAD: Long-term traffic anomaly detection based on crowdsourced bus trajectory data. *World Wide Web* (2017), 1–23.

[32] Christopher W. Landsea and James L. Franklin. 2013. Atlantic hurricane database uncertainty and presentation of a new database format. *Monthly Weather Review* 141, 10 (2013), 3576–3592.

[33] Jae-Gil Lee, Jiawei Han, and Xiaolei Li. 2008. Trajectory outlier detection: A partition-and-detect framework. In *Proc. of ICDE*. 140–149.

[34] Jae-Gil Lee, Jiawei Han, and Kyu-Young Whang. 2007. Trajectory clustering: A partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*. 593–604.

[35] Huanhuan Li, Jingxian Liu, Kefeng Wu, Zaili Yang, Ryan Wen Liu, and Naixue Xiong. 2018. Spatio-temporal vessel trajectory clustering based on data mapping and density. *IEEE Access* (2018).

[36] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. 2018. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)* 50, 6 (2018), 94.

[37] Junli Li, Jifu Zhang, Ning Pang, and Xiao Qin. 2018. Weighted outlier detection of high-dimensional categorical data using feature grouping. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*99 (2018), 1–14.

[38] Sheng Li, Ming Shao, and Yun Fu. 2018. Multi-view low-rank analysis with applications to outlier detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 12, 3 (2018), 32.

[39] Wenjia Li, Houbing Song, and Feng Zeng. 2018. Policy-based secure and trustworthy sensing for internet of things in smart cities. *IEEE Internet of Things Journal* 5, 2 (2018), 716–723.

[40] Yang Li, Qixing Huang, Michael Kerber, Lin Zhang, and Leonidas Guibas. 2013. Large-scale joint map matching of GPS traces. In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 214–223.

[41] Chuanren Liu, Hui Xiong, Yong Ge, Wei Geng, and Matt Perkins. 2012. A stochastic model for context-aware anomaly detection in indoor location traces. In *2012 IEEE 12th International Conference on Data Mining*. IEEE, 449–458.

[42] Yanchi Liu, Chuanren Liu, Nicholas Jing Yuan, Lian Duan, Yanjie Fu, Hui Xiong, Songhua Xu, and Junjie Wu. 2017. Intelligent bus routing with heterogeneous human mobility patterns. *Knowledge and Information Systems* 50, 2 (2017), 383–415.

[43] Dennis Luxen and Christian Vetter. 2011. Real-time routing with OpenStreetMap data. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 513–516.

[44] Jiali Mao, Pengda Sun, Cheqing Jin, and Aoying Zhou. 2018. Outlier detection over distributed trajectory streams. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM, 64–72.

[45] Jiali Mao, Tao Wang, Cheqing Jin, and Aoying Zhou. 2017. Feature grouping-based outlier detection upon streaming trajectories. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2696–2709.

[46] Takumi Matsumoto, Yuya Sasaki, and Makoto Onizuka. 2019. Data slice search for local outlier view detection: A case study in fashion EC. In *EDBT/ICDT Workshops*.

[47] Natwar Modani and Kuntal Dey. 2008. Large maximal cliques enumeration in sparse graphs. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. ACM, 1377–1378.

[48] Andrew Moore and Weng-Keen Wong. 2003. Optimal reinsertion: A new search operator for accelerated and more accurate Bayesian network structure learning. In *ICML*, Vol. 3. 552–559.

[49] Feiping Nie, Shiming Xiang, Yangqing Jia, Changshui Zhang, and Shuicheng Yan. 2008. Trace ratio criterion for feature selection. In *AAAI*, Vol. 2. 671–676.

[50] Hanchuan Peng, Fuhui Long, and Chris Ding. 2005. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis & Machine Intelligence*8 (2005), 1226–1238.

[51] Anatolii Prokhorchuk, Justin Dauwels, and Patrick Jaillet. 2019. Estimating travel time distributions by bayesian network inference. *IEEE Transactions on Intelligent Transportation Systems* (2019).

[52] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. 2000. Efficient algorithms for mining outliers from large data sets. In *ACM SIGMOD Record*, Vol. 29. 427–438.

[53] Kai-Quan Shen, Chong-Jin Ong, Xiao-Ping Li, Zheng Hui, and Einar PV Wilder-Smith. 2007. A feature selection method for multilevel mental fatigue EEG classification. *IEEE Transactions on Biomedical Engineering* 54, 7 (2007), 1231–1237.

[54] Hossein Soleimani and David J. Miller. 2016. ATD: Anomalous topic discovery in high dimensional discrete data. *IEEE Transactions on Knowledge and Data Engineering* 28, 9 (2016), 2267–2280.

[55] Chenfei Sun, Zhongmin Yan, Qingzhong Li, Yongqing Zheng, Xudong Lu, and Lizhen Cui. 2019. Abnormal group-based joint medical fraud detection. *IEEE Access* 7 (2019), 13589–13596.

[56] Guanting Tang, Jian Pei, James Bailey, and Guozhu Dong. 2015. Mining multidimensional contextual outliers from categorical relational data. *Intelligent Data Analysis* 19, 5 (2015), 1171–1192.

[57] Kevin Toohey and Matt Duckham. 2015. Trajectory similarity measures. *Sigspatial Special* 7, 1 (2015), 43–50.

[58] Edward Toth and Sanjay Chawla. 2018. Group deviation detection methods: A survey. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 77.

[59] Md Zia Uddin. 2019. A wearable sensor-based activity prediction system to facilitate edge computing in smart health-care system. *J. Parallel and Distrib. Comput.* 123 (2019), 46–53.

[60] Thé Van Luong, Nouredine Melab, and El-Ghazali Talbi. 2013. GPU computing for parallel local search metaheuristic algorithms. *IEEE Trans. Comput.* 62, 1 (2013), 173–185.

[61] Jan N. van Rijn, Geoffrey Holmes, Bernhard Pfahringer, and Joaquin Vanschoren. 2018. The online performance estimation framework: Heterogeneous ensemble learning for data streams. *Machine Learning* 107, 1 (2018), 149–176.

[62] José R. Vázquez-Canteli, Stepan Ulyanin, Jérôme Kämpf, and Zoltán Nagy. 2019. Fusing TensorFlow with building energy simulation for intelligent energy management in smart cities. *Sustainable Cities and Society* 45 (2019), 243–257.

[63] Pengfei Wang, Yanjie Fu, Guannan Liu, Wenqing Hu, and Charu Aggarwal. 2017. Human mobility synchronization and trip purpose detection with mixture of hawkes processes. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 495–503.

[64] Pengfei Wang, Guannan Liu, Yanjie Fu, Yuanchun Zhou, and Jianhui Li. 2017. Spotting trip purposes from taxi trajectories: A general probabilistic model. *ACM Transactions on Intelligent Systems and Technology (TIST)* 9, 3 (2017), 1–26.

[65] Hao Wu, Weiwei Sun, and Baihua Zheng. 2017. A fast trajectory outlier detection approach via driving behavior modeling. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 837–846.

[66] Jingyao Wu, Zhibin Zhao, Chuang Sun, Ruqiang Yan, and Xuefeng Chen. 2020. Fault-attention generative probabilistic adversarial autoencoder for machine anomaly detection. *IEEE Transactions on Industrial Informatics* (2020).

[67] Liang Xiong, Barnabás Póczos, Jeff Schneider, Andrew Connolly, and Jake VanderPlas. 2011. Hierarchical probabilistic models for group anomaly detection. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 789–797.

[68] Liang Xiong, Barnabás Póczos, and Jeff G. Schneider. 2011. Group anomaly detection using flexible genre models. In *Advances in Neural Information Processing Systems*. 1071–1079.

[69] Cao Lei Yu, Yanwei, Elke A Rundensteiner, and Qin Wang. 2017. Outlier detection over massive-scale trajectory streams. *ACM Transactions on Database Systems (TODS)* 42, 2 (2017), 10.

[70] Rose Yu, Xinran He, and Yan Liu. 2015. Glad: Group anomaly detection in social media analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 10, 2 (2015), 18.

[71] Yanwei Yu, Lei Cao, Elke A Rundensteiner, and Qin Wang. 2014. Detecting moving object outliers in massive-scale trajectory streams. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 422–431.

[72] Nicholas Jing Yuan, Yu Zheng, Xing Xie, Yingzi Wang, Kai Zheng, and Hui Xiong. 2014. Discovering urban functional zones using latent activity trajectories. *IEEE Transactions on Knowledge and Data Engineering* 27, 3 (2014), 712–725.

[73] Jianting Zhang. 2012. Smarter outlier detection and deeper understanding of large-scale taxi trip records: A case study of NYC. In *Proceedings of the ACM SIGKDD International Workshop on Urban Computing*. ACM, 157–162.

[74] Jiong Zhang, Mohammad Zulkernine, and Anwar Haque. 2008. Random-forests-based network intrusion detection systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 5 (2008), 649–659.

[75] Yang Zhang, Nirvana Meratnia, and Paul JM Havinga. 2010. Outlier detection techniques for wireless sensor networks: A survey. *IEEE Communications Surveys and Tutorials* 12, 2 (2010), 159–170.

[76] Xujun Zhao, Jifu Zhang, Xiao Qin, Jianghui Cai, and Yang Ma. 2019. Parallel mining of contextual outlier using sparse subspace. *Expert Systems with Applications* 126 (2019), 158–170.

[77] Zheng Zhao and Huan Liu. 2007. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th International Conference on Machine Learning*. ACM, 1151–1157.

[78] Zhong Zheng, Soora Rasouli, and Harry Timmermans. 2014. Evaluating the accuracy of GPS-based taxi trajectory records. *Procedia Environmental Sciences* 22, 2014 (2014), 186–198.

[79] Zhaohui Zheng, Xiaoyun Wu, and Rohini Srihari. 2004. Feature selection for text categorization on imbalanced data. *ACM Sigkdd Explorations Newsletter* 6, 1 (2004), 80–89.

[80] Jie Zhu, Wei Jiang, An Liu, Guanfeng Liu, and Lei Zhao. 2015. Time-dependent popular routes based trajectory outlier detection. In *International Conference on Web Information Systems Engineering*. Springer, 16–30.

[81] Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. 2008. Maximum entropy inverse reinforcement learning. In *AAAI*, Vol. 8. Chicago, IL, USA, 1433–1438.

[82] Arthur Zimek, Ricardo JGB Campello, and Jörg Sander. 2014. Ensembles for unsupervised outlier detection: Challenges and research questions a position paper. *Acm Sigkdd Explorations Newsletter* 15, 1 (2014), 11–22.

[83] Arthur Zimek, Matthew Gaudet, Ricardo JGB Campello, and Jörg Sander. 2013. Subsampling for efficient and effective unsupervised outlier detection ensembles. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 428–436.