Hybrid Group Anomaly Detection for Sequence Data: Application to Trajectory Data Analytics

Asma Belhadi, Youcef Djenouri, Gautam Srivastava, Alberto Cano, and Jerry Chun-Wei Lin*

***Abstract*—Many research areas depend on group anomaly detection. The use of group anomaly detection can maintain and provide security and privacy to the data involved. This research attempts to solve the deficiency of the existing literature in outlier detection thus a novel hybrid framework to identify group anomaly detection from sequence data is proposed in this paper. It proposes two approaches for efficiently solving this problem: i) *Hybrid Data Mining-based algorithm*, consists of three main phases: first, the clustering algorithm is applied to derive the micro-clusters. Second, the kNN algorithm is applied to each micro-cluster to calculate the candidates of the group's outliers. Third, a pattern mining framework gets applied to the candidates of the group's outliers as a pruning strategy, to generate the groups of outliers, and ii) a *GPU-based* approach is presented, which benefits from the massively GPU computing to boost the runtime of the hybrid data mining-based algorithm. Extensive experiments were conducted to show the advantages of different sequence databases of our proposed model. Results clearly show the efficiency of a GPU direction when directly compared to a sequential approach by reaching a speedup of 451. In addition, both approaches outperform the baseline methods for group detection.**

***Index Terms*—Sequence Databases, Anomaly Detection, Data Mining, GPU Computing.**

I. Introduction

Sequence data analysis is a challenging research area from data mining because it can find correlations of ordered events, which have real-world implications [1]. DNA sequencing [2], Weblog [3],smart manufacturing [4], and trajectory databases [5] are all examples of areas that actively use sequential data mining. With regards to intelligent transportation [6], analysts encounter a multitude of sequence data represented by a trajectories' set that is derived using people's mobility, taxis, motorcycles, buses, cars, etc. Existing approaches to solving the outlier detection problem for sequential data have solely considered simple basic outliers [7]–[9]. However, in real-world applications, outliers in sequence data frequently occur in groups, for example, when a group of taxis deviates from the anticipated and usual trajectory due to road work, or groups of time series that deviate from the normal data sensors in a given period. This paper introduces a new framework for detecting outliers of the sequence data, which is to find groups of anomalous behaviors existing in sequence data.

A. Motivation

Consider the problem of taxi frauds, where taxi trajectories begin from some source location and end at a destination location. Usually, known trajectory outlier detection (TOD) methodologies [10], [11] may detect individual taxi frauds, represented by simple trajectory outliers, where individual taxi frauds follow the expected path from a given source location to a destination point, where they deviate a lot from normal taxi routes. However, existing TOD algorithms are unable to determine collective taxi frauds, where groups of taxis deviate from a normal trajectory using the same deviation point or deviation time. Detecting the collective outliers in the taxi fraud problem could help the city officials to analyze and discover patterns with relevant information. An example may include installing surveillance cameras at hot spots. One approach to deal with this challenging issue is to first consider the taxi trajectories as a set of sequence data and then solve the group sequence data outlier detection problem.

Climate change analysis is another example of the motivation of this paper. Meteorologists attempt to determine the reasons for sudden changes in the path of a hurricane. Being able to predict these sudden changes is the main research interest in the outlier detection community [11]. Hurricanes in the United States are recently notorious for their unexpected path since they were expected to hit the lands, where many citizens were unprepared. Detecting sudden changes in a hurricane could be easily fit to the trajectory outlier detection problem, where hurricanes are represented by the temporal trajectories. Therefore, detecting outlying in hurricanes helps to quickly figure out the sudden hurricane track changes. Besides, studying the different correlations among hurricane trajectory outliers can identify useful patterns to help governments in predicting future hurricanes.

B. Contribution

This research work presents a new framework and methodology that can identify a group of sequence data outliers. The approach explores different data mining techniques in different stages, which allows accurately be able to identify groups for sequence data outliers. GPU-based computing is also investigated to boost the runtime of the

A. Belhadi is with the Department of Technology, Kristiania University College, Oslo, Norway. Email: Asma.Belhadi@kristiania.no

Y. Djenouri is with the SINTEF Digital, Forskningsveien 1, 0314, Oslo, Norway. Email: Youcef.Djenouri@sintef.no

G. Srivastava is with the Department of Mathematics & Computer Science, Brandon University, Manitoba, Canada. Email: SRIVASTAVAG@brandonu.ca and with the Research Centre for Interneural Computing, China Medical University, Taichung, Taiwan.

A. Cano is with the Department of Computer Science, Virginia Commonwealth University, Richmond, VA, USA. Email: acano@vcu.edu

J. C. W Lin is with the Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, Bergen, Norway. Email: jerrylin@ieee.org (*Corresponding author)

This is the author accepted version of an article published in
IEEE Transactions on intelligent transportation systems
http://dx.doi.org/10.1109/TITS.2021.3114064

proposed approach in dealing with big sequence databases. We can summarize our primary contributions in the following list:

1) We propose a novel technique, named HDM-GAD (**H**ybrid **D**ata **M**ining for **G**roup **A**nomaly **D**etection), which firstly exploits the *DBSCAN* (Density-Based Spatial Clustering of Applications with Noise) clustering method to identify potential outliers represented as micro-clusters, and secondly employs kNN to prune micro-clusters, and finally uses a methodology for pattern analytics to discover groups sequence data outliers.
2) We propose a GPU-based solution, called GHDM-GAD (**G**PU based **H**ybrid **D**ata **M**ining for **G**roup **A**nomaly **D**etection), which exploits the GPU massively threaded computing to boost the runtime of HDM-GAD in identifying the group of sequence data outliers from the big sequence databases. Besides, the optimization of our GPU-based solution is developed by minimizing wrap divergence among GPU blocks.
3) We show the great performance of the approaches using two different use cases (trajectory and sequence databases). The experimental results reveal that HDM-GAD and GHDM-GAD outperform current state-of-the-art algorithms used for outlier detection.

II. Related Work

Singh *et al.* [12] investigated several outlier identification approaches in various data formats, including sequence data. It also distinguished between several types of outliers, including simple outliers, contextual outliers, and collective outliers. The authors in [13] presented a new model that can be used as a forecasting model in training sequence data, as well as predicting future values. Anomalies are identified when observed values are outside some prediction interval. This interval is found using predicted value as well as confidence coefficient. The authors in [14] created an incremental-based probabilistic model for learning outliers from sequence data, in which the score for each sequence data point is used to determine the deviation from the actual model. Yamanishi *et al.* [15] advocated the use of online algorithms for discounting learning that may be used to learn a probabilistic algorithm created in [14]. It discovered anomalies in an online process by analyzing the sequence data source using a finite mixture model, where a high score for the sequence data indicated a very high probability that it is a statistical outlier. Xie *et al.* [16] used existing concepts from chaos theory to transform sequence data moving into a multi-dimensional space. Following that, several types of anomalous are deduced using a decision tree method. The authors in [17] involved an anomaly detection method. It presented a non-parametric-based learning method that takes into account malfunctioning sensors, unexpected phenomena, and a variable probability distribution of sequence data. The authors in [18] used a local outlier factor method to summarize sequence data in compact and various layers of IoT architecture by creating a novel density-based sampling approach. This method avoids the enormous amount of memory space required by the local outlier factor while identifying a long series of outliers that are not recognized by conventional sequence data outlier detection approaches. The authors in [19] presented two recurrent-based methods for detecting sequence-based anomalies. Both methods used autoencoders in conjunction with sparsely connected recurrent neural networks to build several models with varying neural network connection topologies. The authors in [20] developed a multi-scale convolutional recurrent encoder-decoder to detect aberrant behaviours in multivariate sequence data. It initially created multi-resolution matrices with various layers of distinct time increments. The convolutional encoder was then utilized to encode each series of data and capture the temporal patterns. Finally, a convolutional decoder is utilized to rebuild, identify, and diagnose abnormalities. The authors in [21] established a method for detecting anomalies in mixed-type sequencing data. It investigated the various correlations between the sequence data and identified the frequent patterns to create and train the isolation forest structure. In the intelligent transportation context, Belhadi *et al.* [22] suggested a two-phase secure-based method for identifying abnormalities from ride-hailing trajectories. The first phase seeks to identify taxi fraud by computing the distance between each stop point in each taxi route, while the second seeks to enhance the mining process via the use of both feature selection and sliding windows techniques. Javed *et al.* [23] utilizes a combination of long short-term memory, convolutional neural networks, and a multi-attention method to identify outliers in-vehicle data. Multiple classifiers are also incorporated by developing a novel strategy based on the principle of average predicted probabilities. Wang *et al.* [24] developed a novel method for enhancing the safety of the autonomous vehicle system. The extended Kalman filter is first used for smoothing the sensor reading. The support vector machine is then performed to identify the sensor anomalies. Table I presents the merit and the limitation of the relevant works to this research study.

TABLE I
Related Work Summary.

Work	**Merit**	**Limitation**
Singh *et al.* [12]	Simple, contextual, collective outliers	Use traditional techniques
Yu *et al.* [13]	Both anomaly detection and forecasting	Hard to estimate the time window
Yamanishi *et al.* [15]	Use incremental learning	Hard to build the training data
Xie *et al.* [16]	Identify different kind of outliers	Use traditional techniques
Nesa *et al.* [17]	Detecting Outliers from IoT data	Hard to build the training data
Na *et al.* [18]	Less memory consumption	Use tradition techniques
Zhang *et al.* [20]	Use hybrid deep learning models	Hard to build the training data
Feremans *et al.* [21]	Study pattern correlation	High time consuming
Belhadi *et al.* [22]	Application to taxi frauds	High time consuming
Javed *et al.*, [23]	Use hybrid deep learning models	Hard to build the training data
Wang *et al.* [24]	Detecting anomaly sensors	Use traditional techniques

In our analysis, there has been no work explores the group outlier detection from the individual outliers, particularly for sequence databases. Therefore, the existing algorithms are unable to be applied to sequence data. They tend to focus on being able to find individual outliers. A hybrid methodology is proposed here to detect group anomaly detection from sequence data by investigating data mining, and high-performance computing in exploring the individual outliers space.

III. Hybrid Data Mining based Framework

A. Problem Statements

Definition 1 (Sequence Databases). Let us clearly define a sequence database $S = \{S_1, S_2, \ldots, S_m\}$, where each sequence data S_i is a sequence of spatial locations points $P = \{p_{i1}, p_{i2}, \ldots, p_{in}\}$.

Definition 2 (Candidate of Group Sequence Data Outliers). Let us define a candidate of group sequence data outliers $\mathcal{G} = \{S_1^{(\mathcal{G})}, S_2^{(\mathcal{G})}, \ldots, S_s^{(\mathcal{G})}\}$, where each sequence data in $\mathcal{G}$ is an outlier.

Definition 3 (Group Density). Let us define density from candidate group sequence data outliers $\mathcal{G}$ for a given user threshold γ using eq. (1):

$$\mathcal{GD}(\mathcal{G}) = \frac{\sum\limits_{S_i, S_j \in \mathcal{G}} distance(S_i, S_j)}{|\mathcal{G}| \times |(\mathcal{G} - 1)|}. \tag{1}$$

Note that $distance(S_i, S_j)$ is the distance between two sequence data and it depends on the data representation used. In this research work, we use two data representation, trajectories and time series. Therefore, we used the DTW (Dynamic Time Wrapping) for computing the distance between two time series, and two trajectories data using eq. (2):

$$D(S_i, S_j) = \begin{cases} 0 & \text{if } |S_i| = |S_j| = 1 \\ \infty & \text{if } |S_i| = 1 \text{ or } |S_j| = 1 \\ S_i^0 - S_j^0 + MIN & \text{otherwise,} \end{cases} \tag{2}$$

where MIN is defined using eq. (3):

$$MIN = min\{D(S_i', S_j'), D(S_i, S_j'), D(S_i', S_j)\}. \tag{3}$$

Note that S_i^0, S_j^0 are the current components of the time series data S_i, and S_j. S_i' and S_j' are the sequences of S_i, and S_j without considering the current components S_i^0 and S_j^0, respectively.

Definition 4 (Group Sequence Data Outlier Problem). The Group Sequence Data Outlier Problem aims to discover all groups of sequence data outliers $\mathcal{G}$, for a given user threshold γ such using eq. (4):

$$\mathcal{GD}(\mathcal{G}) \leq \gamma. \tag{4}$$

A basic procedure for finding clusters of outliers in sequence data entails the following two steps:

- Consider all possible combinations of the outliers in the sequence data.
- Utilize Definition 3 to evaluate each subset independently.

Though our method does not scale (its complexity is $O(2^{|\mathcal{G}|})$), therefore, we propose in the following section a general methodology based on hybrid data mining techniques to accurately retrieve the group sequence data outliers.

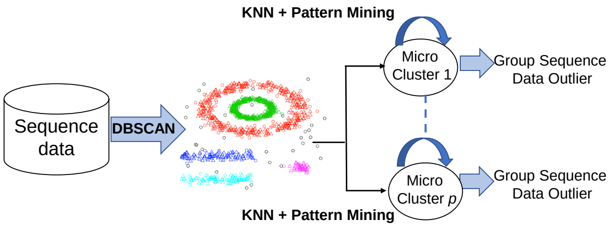

Fig. 1. HDM-GAD Framework.

B. Principle

This section presents our algorithm HDM-GAD (Hybrid Data Mining for Group Anomaly Detection), which explores an efficient manner in the search space of sequence data to determine groups for sequence data outliers. Through our extensive research presented here, the clustering, the k-nearest neighbours, and the pattern mining methods are adopted to reduce the search space exploration, as well as identify groups of sequence data outliers. As explained in Fig. 1, the micro-clusters are first identified making use of the *DBSCAN* algorithm, then next we use a pruning strategy on each micro-cluster using the kNN method. Finally, we employ a pattern mining method to the output subset for known candidates of the group of sequence data to derive the groups of sequence data outliers. Following in the next part of the remainder of this section, an explanation of the main components of the HDM-GAD framework is detailed.

C. Clustering

First, formal definitions and terms of the clustering method used in this research work are given.

Definition 5 (Sequence Data Neighborhoods). *We define the neighborhoods of a sequence data S_i, $\mathcal{N}_{S_i}$, for a given threshold ϵ using eq. (5):*

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

Definition 6 (Core Sequence Data). *A sequence data S_i is known as core sequence data iff there exists at least some minimum number of sequence data S_{min} such that $|\mathcal{N}_{S_i}| \geq S_{min}$*

Definition 7 (Micro-Cluster). *A sequence data cluster C_i is known as a micro-cluster iff $0 < |C_i| \leq \mu$, where μ is defined as the threshold.*

DBSCAN and *OPTICS*(Ordering Points To Identify the Clustering Structure) [25]–[27] aim to find clusters with different sizes, large, small, and micro-clusters. Individual outliers are considered as noises. In this research work, we propose a new definition of outliers which is a group of sequence data outliers. Here we clearly introduce usability of the *DBSCAN* method for finding micro-clusters, and each and every micro-cluster can be considered a candidate of the group for sequence data outliers. The ϵ-neighborhood for each and every given piece of sequence data can be calculated making use of Def. 5. The core sequence data is identified by

Def. 6. *DBSCAN* sequentially identifies the density-reachable sequence data from these core sequence data directly, which comprises merging density-reachable clusters. The algorithm terminates when no new sequence data can be added to any of the existing clusters. In the beginning, the set of sequence data are grouped using *DBSCAN*, producing multiple clusters with varied sizes. Each micro-cluster (see Def. 7) is regarded as a group candidate. Consequently, sets of candidates of groups sequence data called $\{\mathcal{G}_i^+\}$ are generated.

D. Pruning Strategy

Previous clustering outputs micro-clusters, where each and every micro-cluster can form candidate groups for the sequence data. These groups comprise individual sequence data outliers that can be close to one another. That being said, they could also have normal sequence data in it. For pruning the candidate's group sequence data, we propose a cost-effective pruning strategy based on the kNN method.

Definition 8 (kNN of a Sequence Data). *Let us define the* kNN *of the sequence data* S_i*, denoted by* $kNN(S_i)$ *using eq. (6):*

$$kNN(S_i) = \{S_j \in S \setminus \{S_i\} | d(S_i, S_j) \leq k_{dist}(S_i)\}. \quad (6)$$

$k_{dist}(S_i) = d(S_i, S_l)$ *is the k-distance of the sequence data* S_i *defined such as it exists k sequence data* $S' \in S$*, and it holds that* $d(S_i, S_l) \geq d(S_i, S')$*.*

Definition 9 (Outlierness degree of a Sequence Data). *We define the outlierness degree of a given sequence data* S_i*, denoted by* $\delta(S_i)$ *using eq. (7):*

$$\delta(S_i) = |\{S_j | j \neq i \vee S_j \in (kNN(S_i) \cap \mathcal{G}^+)\}|. \quad (7)$$

For example, consider the kNN (with k = 3) of a given sequence data S_1 is $\{S_2, S_3, S_4\}$, and the set of candidate group of sequence data outlier is $\{S_1, S_2, S_4, S_5, S_6\}$, the outlierness degree of S_1 is $|\{S_2, S_4\}|$, which is equal to 2.

Next, we introduce an adaptation of the kNN algorithm for pruning the candidate sequence data outliers. The proposed method takes as input the sets of all sequence data candidate $\mathcal{G}^+$. The aim is to reduce the number of candidate sequence data outliers on each micro-cluster. First, it adds the sequence data outlier with the highest outliers degree, S_1^+, to the set of candidate sequence data labelled outliers using S_1^+, and we can denote this using $\mathcal{G}_1^+$. Secondly, we generate the complete set of potential candidates using S_1^+. We can define sequence data s as a candidate potentially from $S_1^+ \iff s \in \mathcal{G}_1^+ \vee s \in kNN(S_1^+)$. The same methodology we can apply recursively for the complete set of potential candidates which can be added to $\mathcal{G}_1^+$. We repeat the overall procedure for the complete set of micro-clusters.

E. Pattern Mining

Let us consider $< P, \mathcal{G}^+, NormalizedDensity(\bullet), \gamma$ ¿, which itself could fit into a pattern mining problem that can be used to be represented by transactions as a set D, an item set I, a support function $Support$, as well as a minimum support $minsup$ as given here:

1) $D = P$.
2) $I = \mathcal{G}^+$.
3) $Support(\bullet) = NormalizedDensity(\bullet)$.
4) $minsup = \gamma$.

Each data point is treated as a transaction, and each candidate for sequence data is treated as an item. A pattern is a subset of the $\mathcal{G}^+$ that has been pruned. The pattern f has support equal to the density of the group of f sequence data. The γ is then set as the minimum threshold. With the support function, $Density(\bullet)$ and the minimum support set to $gamma$, the pattern analytics and mining process is performed to the set of transactions D and the set of items I. The discovered frequent patterns are seen as a collection of groupings of sequence data outliers. The GAD issue, by definition, seeks to identify a **non-redundant** collection of sequence data outliers. If we use a traditional pattern mining method, we may be able to extract duplicate patterns. To overcome this issue, we seek closed patterns that guarantee the derivation of a non-redundant collection of sequence data outliers. To find closed patterns, we utilized the Closet algorithm [28] in our implementation. The method consists of two main steps. First, we mine all size 1-frequent patterns that are closed. Second, any new patterns that can be generated directly add to the size 1-frequent patterns that are closed, and any need to mine more frequent patterns is alleviated.

Another example is to consider the data points $P = \{P_1, P_2, P_3, P_4, P_5\}$, and 50 sequence data from S_1 to S_{50}, we then assume that the following micro-cluster $\mathcal{G}^+$=$\{S_1, S_2, S_3, S_6, S_7, S_8\}$ is obtained after the pruning step:

- $S_1 = \{P_1, P_2, P_3\}$.
- $S_2 = \{P_2, P_4\}$.
- $S_3 = \{P_1, P_3, P_4\}$.
- $S_6 = \{P_2, P_2, P_4\}$.
- $S_7 = \{P_2, P_4, P_5\}$.
- $S_8 = \{P_1, P_5, P_7\}$.

The transaction database is obtained as follows,

- $P_1 = \{S_1, S_3, S_8\}$.
- $P_2 = \{S_1, S_2, S_6, S_7\}$.
- $P_3 = \{S_1, S_3, S_6\}$.
- $P_4 = \{S_2, S_3, S_6, S_7\}$.
- $P_5 = \{S_7, S_8\}$.

For instance, if we consider $minsup\ leq$ 40%, the group $\{S_1, S_3\}$ is considered as outlier. Thus, the overall performance of our HDM-GAD is lowered through the total number for sequence data that may become too big.

Algorithm 1 presents the pseudo-code of the HDM-GAD algorithm. The process starts by creating the clusters with *DBSCAN* algorithm in line 3. For each micro-cluster, the neighborhood computation with a closed pattern mining algorithm is performed, where the group outliers on each micro-cluster are derived (from lines 5 to 10). The group outliers of all micro-clusters are returned in line 11.

Algorithm 1 HDM-GAD Algorithm

```
**Input**: $S = \{S_1, S_2, \dots, S_m\}$: the sequence database.
**Output**: $\mathcal{G}$: the groups of sequence data outliers of $S$.
$C \leftarrow DBSCAN(S)$
$\mathcal{G} \leftarrow \emptyset$
**for** each micro-cluster $C_i$ **do**
    $G_i \leftarrow kNN(C_i)$
    $P_i \leftarrow PatternMining(G_i, C_i)$
    $G_i \leftarrow Closed(P_i)$
    $\mathcal{G} \leftarrow \mathcal{G} \cup G_i$
**end for**
**return** $\mathcal{G}$
```

F. GHDM-GAD: GPU-based Hybrid Data Mining algorithm

Here, the GPU-based Hybrid Data Mining (GHDM-GAD) approach is developed, which has the goal of fixing the HDM-GAD issues. Graphical Processing Units (GPU) are integrated with HDM-GAD to speed up its performance and deal with large-scale data. GPUs are graphic cards that are often recently used for the solving of complex issues, such as data mining, computer visions, 3D rendering. GPU programming models are composed of several GPU threads, logically grouped into many blocks, and physically organized into different wraps (32 to $1,024$ threads per wrap, depends on the architecture used). All threads of the same block share the same memory space called *shared memory*. Blocks have access to global as well as constant memory. Threads may be grouped into groups of 32, and thread blocks of size are $1,024$, i.e., 2^{10}. The sequence database is first divided into clusters using the *DBSCAN* algorithm. The micro-clusters are then sent to the shared memory of the GPU blocks, where each block b_i is mapped to one micro-cluster C_i. The j^{th} thread in b_i, th_{ij} first computes the kNN from the j^{th} sequence data of the micro-cluster C_i, and then discovers the frequent patterns of the candidates group retrieved from the j^{th} sequence data of the micro-cluster C_i. Afterward, each block finds the local group sequence data outlier at each micro-cluster. A global groups sequence data outlier will be a local group sequence data outliers that maximize a function described in Def. 4.

Looking at the issues theoretically, GHDM-GAD can improve the HDM-GAD algorithm sequentially using a massive threading approach and the heaving computational power of GPUs. Simultaneous, it is forced to be calculated globally as well as individual sequence data outliers. GHDM-GAD can also minimize CPU and GPU communication. It can do this firstly by loading the database itself as a whole onto the GPU. Next, host memory can be used directly to return both global and individual sequence data outliers. The common problem of the GPU-based deployment is the wrap divergence issue. In the next section, we suggest a method for reducing the number of wrap divergences. The number of wrap divergence should be established initially. Every block in the proposed GPU-based approach deals with a distinct quantity of sequence data. Each thread compares the sequence data it is mapped with to find the group of sequence data outliers on GPU. As a result, wrap divergence can be caused by one of two factors: To begin, each thread is responsible for a distinct number of sequence data. Some threads end before others in this situation. Second, when a specific thread's comparison operation fails to detect sequence data outliers in the block it is mapped with, it is terminated. These two factors influence the number of wrap divergence(*WD*), which may be calculated based on the number of comparisons performed by the various threads using Eq. (8):

$$WD = max\{max\{|t_{(r*w)+i}| - |t_{(r*w)+j}|\}\}, \tag{8}$$

where $(i, j, r) \in [1...w]^3$, and $|t_{(r\times k)+i}|$ is the size of the $(r \times k) + i^{th}$ sequence data that is assigned to the i^{th} thread and allocated to the r^{th} grid. It's worth noting that k denotes the number of blocks.

Wrap divergence may also be calculated based on the distribution of sequencing data. As a result, the two situations listed below may be differentiated: **Irregular Distribution of Sequence Data:** when the sequence data are highly different in size, wrap divergence may be estimated as the maximum number of sequence data minus one. This yields Eq. (9):

$$\lim_{k\to+\infty} WD(m) = m - 1. \tag{9}$$

Regular Distribution of Sequence Data: In contrast to the first example, this occurs when there is a minor variation in the amount of the sequence data. Let us consider r_1 the variation among the sequence data. This yields Eq. (10):

$$\lim_{m\to+\infty} WD(m) = r_1. \tag{10}$$

In the next section, we present a method that reduces wrap divergence while aiming to enhance sequence data assignment on various blocks. The assignment of the sequence data is performed according to their size, where sequence data of i points is assigned to the i^{th} block. As a result, the number of blocs equals the number of points. Because the threads in each block have the same quantity of sequence data, the wrap divergence between them is reduced when using this technique. However, if multiple sequence data contain the same amount of points, load balancing across blocks is ignored. Some blocks handle a large amount of sequence data, whereas others handle a little amount of sequence data. This reduces the speed of the GPU-based group sequence data outlier identification method. To address this issue, we suggest capturing sequence data that reduces load balancing and sorting the theme based on the number of points. Each sequenced data is then assigned to a thread, with the i^{th} thread handling the i^{th} sequence data. As a result, all blocks contain the same quantity of sequence data, ensuring load balance across blocks.

IV. Performance Evaluation

Twelve databases have been used in the experiments, six for trajectories, and six for time series. The data are retrieved from the UCI machine learning repository[1]. The pre-processing step depends on the representation used in the mining process. The

[1] https://archive.ics.uci.edu/ml/datasets.php

time-series data is coming from sensors, filtering technique [29] is used to remove errors from the time series. The trajectory data is gathered from GPS data. Mapping strategy is used to map each trajectory to the corresponding grid in the road network [30]. The description of the data is given in the following:

1) Trajectory Data: We used six different datasets, Geolife, Manhattan, and ECMLPKDD 2015 competition is data with medium size, where the number of trajectories varies from $1,000$ to $7,000$. We also consider big trajectory datasets, such as Taxi13-1, Taxi13-2, and Taxi15, where the number of trajectories varied from 1 million to 3 million.
2) Time Series Data: Similarly to the trajectory data, we used six different datasets with various sizes, medium sizes varied from 6,000 to 43,000 time series such as Australian Sign Language signs, Appliances energy prediction, Amazon Access Samples, and Beijing PM2.5 Data. We also consider two big time-series datasets (Buzz in social media, and Beijing Multi-Site Air-Quality Data), which varied from $100,000$ to $400,000$ time series.

In addition, Table II summarizes the data description used in the experiments.

TABLE II
DATA DESCRIPTION.

Database	$\|S\|$	$\|P\|$
Geolife	17,621	3,000
Manhattan	1,000	1,500
ECML PKDD 2015 Competition	7,184	125
Taxi13-1	1,890,000	1,025
Taxi13-2	3,690,000	1,500
Taxi15	3,690,000	12,521
Amazon Access Samples	30,000	20,000
Appliances energy prediction	19,735	19
Australian Sign Language signs	6,650	15
Buzz in social media	140,000	77
Beijing PM2.5 Data	43,824	13
Beijing Multi-Site Air-Quality Data	420,768	18

A. Parameter Settings

HDM-GAD requires several parameters to be well-tuned to reach better performance. Thus, intensive experiments have been carried out at this stage, where the following parameters are fixed:

1) **Clustering:** Both ϵ and S_{min} for *DBSCAN* algorithm, and μ for determining the micro-clusters.
2) **Pruning:** The number of neighbours, k and the density threshold γ for kNN.
3) **Pattern mining:** The pattern mining minimum support threshold, $minsup$, for discovering the relevant patterns among sequence data.

Figs. 2 and 3 show the parameters setting results of *DBSCAN*, ϵ from 0.2 to 1.0, and S_{min} from 2 to 10, the μ parameter for determining the micro-clusters from 2 to 10, the density threshold values from 0.2 to 1.0, the number of neighborhood from 2 to 10, and the minimum support values from 10% to 99%. For all used sequence databases, the accuracy of the ROCAUC is greater than 0.72 but does not exceed 0.75. These findings are explained by the fact that the concept of the micro-clusters can successfully identify the group of sequence data outliers but not optimally. The results by involving the pruning strategies and the pattern mining steps help to improve the accuracy of the designed model. This can be demonstrated by the fact that the kNN technique prunes the search and holds only the closest neighbours of sequence data outliers among the micro-clusters. Furthermore, the pattern mining method further decreases the search space by analyzing the frequent patterns in the micro-clusters within the group of sequence data outliers. Table III collects the best performing parameters of the HDM-GAD algorithm. Those values will then be used in the experimental evaluation.

TABLE III
BEST PARAMETERS OF HDM-GAD.

Database	ϵ	S_{min}	μ	k	γ	minsup
Geolife	0.3	10	5	5	0.4	60
Manhattan	0.6	5	7	7	0.5	70
ECML PKDD 2015 Competition	0.8	5	7	8	0.6	95
Taxi13-1	0.6	7	6	6	0.7	80
Taxi13-2	0.6	7	5	5	0.8	92
Taxi15	0.6	8	8	7	0.8	80
Amazon Access Samples	0.2	5	10	5	0.2	50
Appliances energy prediction	0.5	10	10	5	0.2	50
Australian Sign Language signs	0.5	5	3	10	0.5	75
Buzz in social media	1.0	10	8	10	0.5	99
Beijing PM2.5 Data	0.5	10	3	10	1.0	75
Beijing Multi-Site Air-Quality Data	0.5	10	8	10	0.2	50

B. HDM-GAD vs Baseline Sequential Group Detection Algorithms

In this experiment, we aim to show the performances of HDM-GAD compared to the state-of-the-art algorithms in terms of both quality of returned solutions and computational time. To the best of our knowledge, this is the first work exploring the correlation between the outliers in sequential data. Therefore, two baseline group outlier detection algorithms called (DGM [31] and WATCH [32]) are adopted to process sequence data. Figs. 4 and 5 display the quality of solutions expressed by the mean ROCAUC values in multi-sequence databases for the three algorithms. They also showed the results for a varied number of anomalous sequences. The HDM-GAD has better performance in almost any case by varying the number of anomalous sequences from 10 to $1,000$. For a total of 72 cases, HDM-GAD holds the best performance among 64 cases, DGM goes to have the best performance with 3 cases, and WATCH is with the best performance for 5 cases. In comparison, the accuracy of the HDM-GAD stabilizes with increased sequence outliers and does not drop below 0.87 while the accuracy of the two other algorithms decreases below 0.76. This is demonstrated by the fact that our model uses more complex and new techniques focused on clusters, neighbourhoods and pattern mining, whereas the baseline methods use fewer technical principles of outlier detection based on data distribution. Regarding processing speed, which can be observed in Fig. 6, the designed model works successfully compared to the

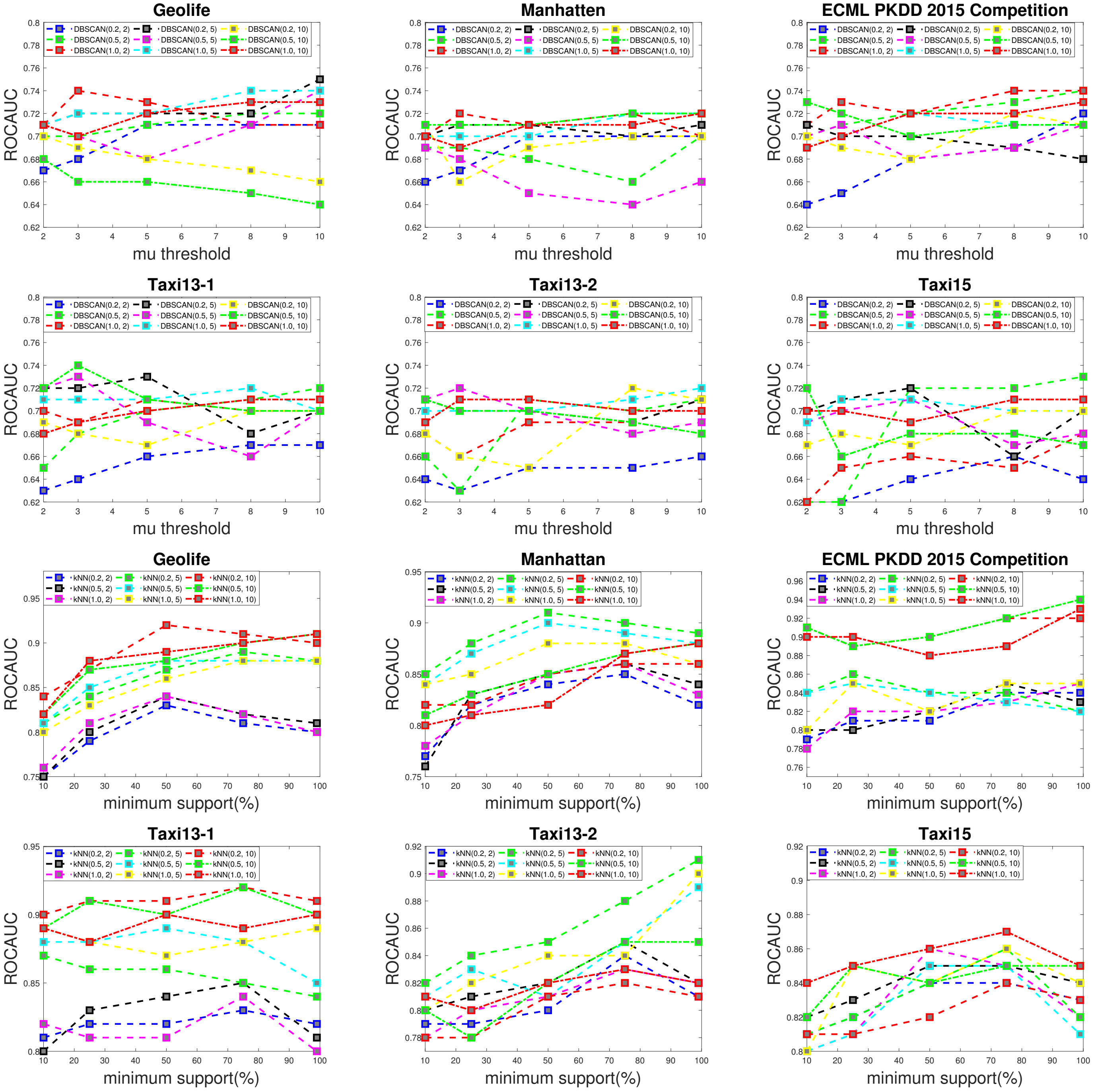

Fig. 2. The parameter setting of the HDM-GAD on trajectory databases.

baseline approaches. The main reason is that the designed model is performed by combining several efficient clustering, kNN, and pattern mining techniques that can successfully identify the candidates' groups of sequence data.

C. GHDM-GAD vs State-of-the-art GPU-based Group Detection Algorithms

The third experiment shows the scalability of GHDM-GAD compared to the state-of-the-art GPU-based group detection algorithms: ($GkNN$ [33] and modified-EFM [34]). Table IV presents both the runtime, the speedup, and the accuracy of the proposed GPU-based solution using big sequence databases. The results confirm that the runtime decreases, and the accuracy increases with the increase of the number of blocks and the number of threads per block. Besides, our GPU-based solution outperforms the other GPU-based baseline algorithms in most cases. This performance is justified by the efficient mapping of the sequence data among the different GPU blocks that take advantage of the massively GPU threaded. In some scenarios, the accuracy of $GkNN$ is better than GHDM-GAD because our algorithm is sensitive to the clustering results, where the micro should be accurately

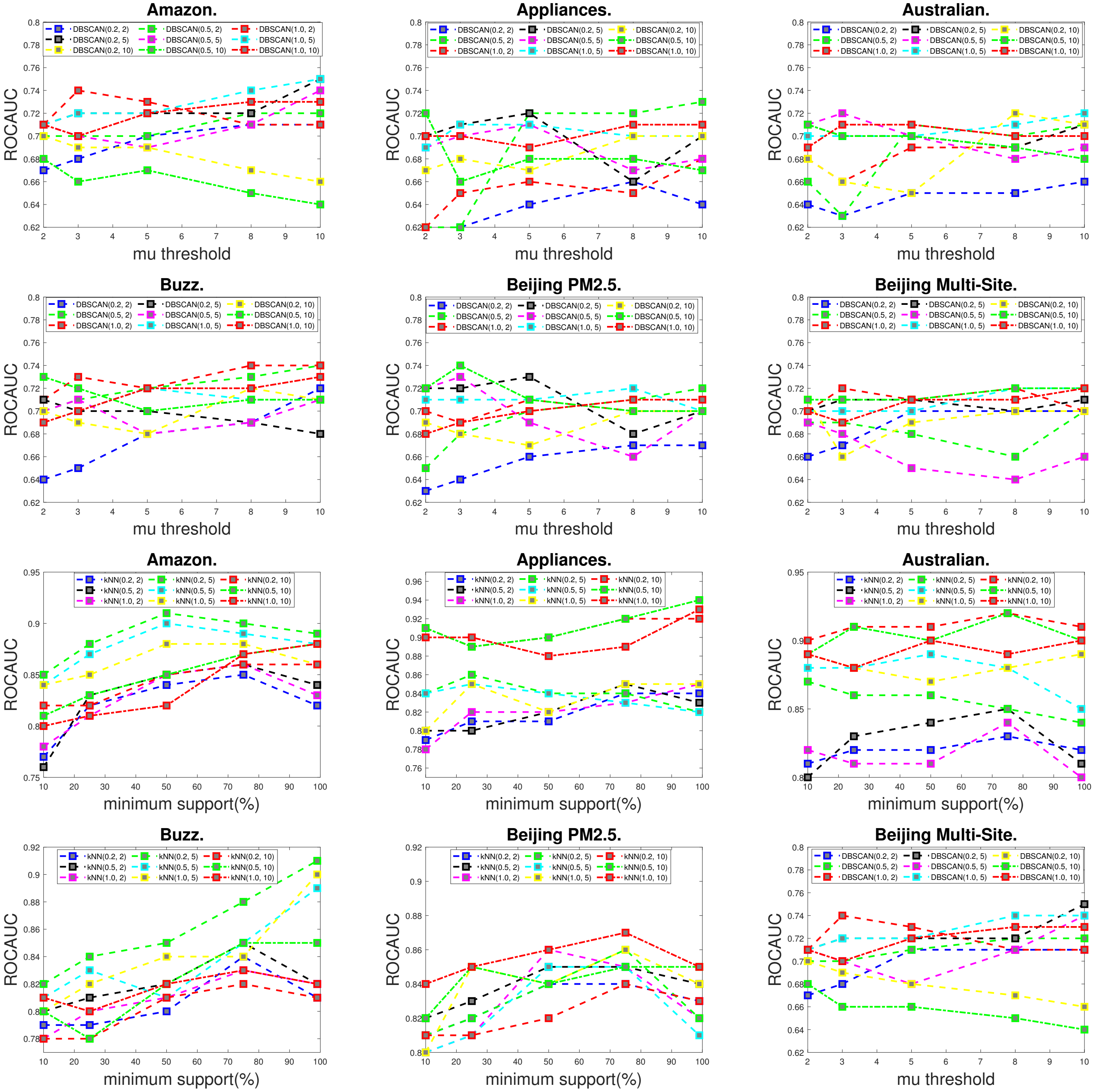

Fig. 3. The parameter setting of the HDM-GAD on time series databases.

retrieved. These results are also due to our efficient intelligent strategy in dealing with the wrap divergence issue. Moreover, Table V presents the speedup of GHDM-GAD with different GPU architectures. The datasets are respectively presented as D1: Geolife, D2: Manhattan, D3: ECML PKDD 2015 Competition, D4: Taxi13-1, D5: Taxi13-2, D6: Taxi15, D7: Amazon Access Samples, D8: Appliances energy prediction, D9: Australian Sign Language signs, D10: Buzz in social media, D11: Beijing PM2.5 Data, D12: Beijing Multi-Site Air-Quality Data. Note the results in Table V are varying under the number of GPU blocks from 256 to $1,024$, with 32 threads per block. We remark from this study that the GHDM-GAD speedup increases with increases the performance of GPU architecture. For instance, with Tesla C2075, the speedup does not exceed 451, where the speedup with K20 and T4 reach 462, and 467, respectively. These results reveal the capability of our GPU-based solution in giving better performances on more advanced GPU-based architectures.

D. Discussion

In this part, we discuss many open research questions concerning group sequence data outlier identification.

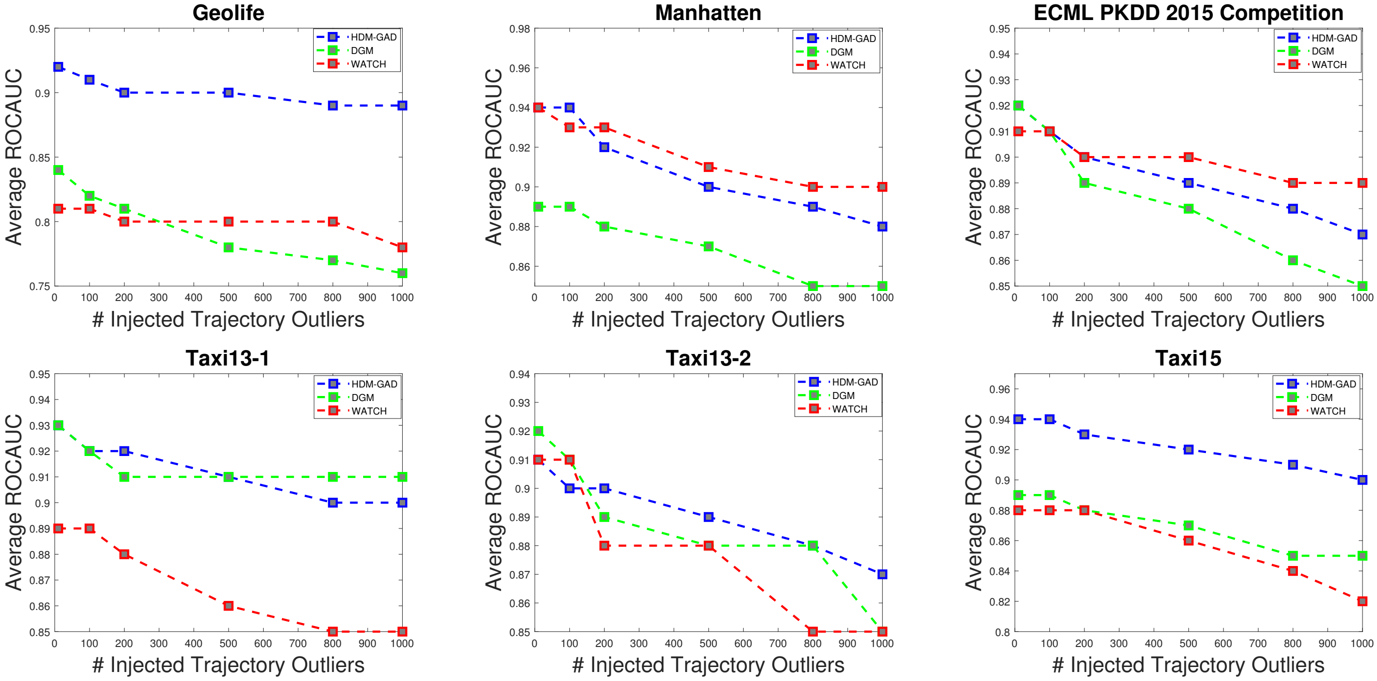

Fig. 4. Accuracy of the HDM-GAD and the baseline group outlier detection algorithms using trajectory databases

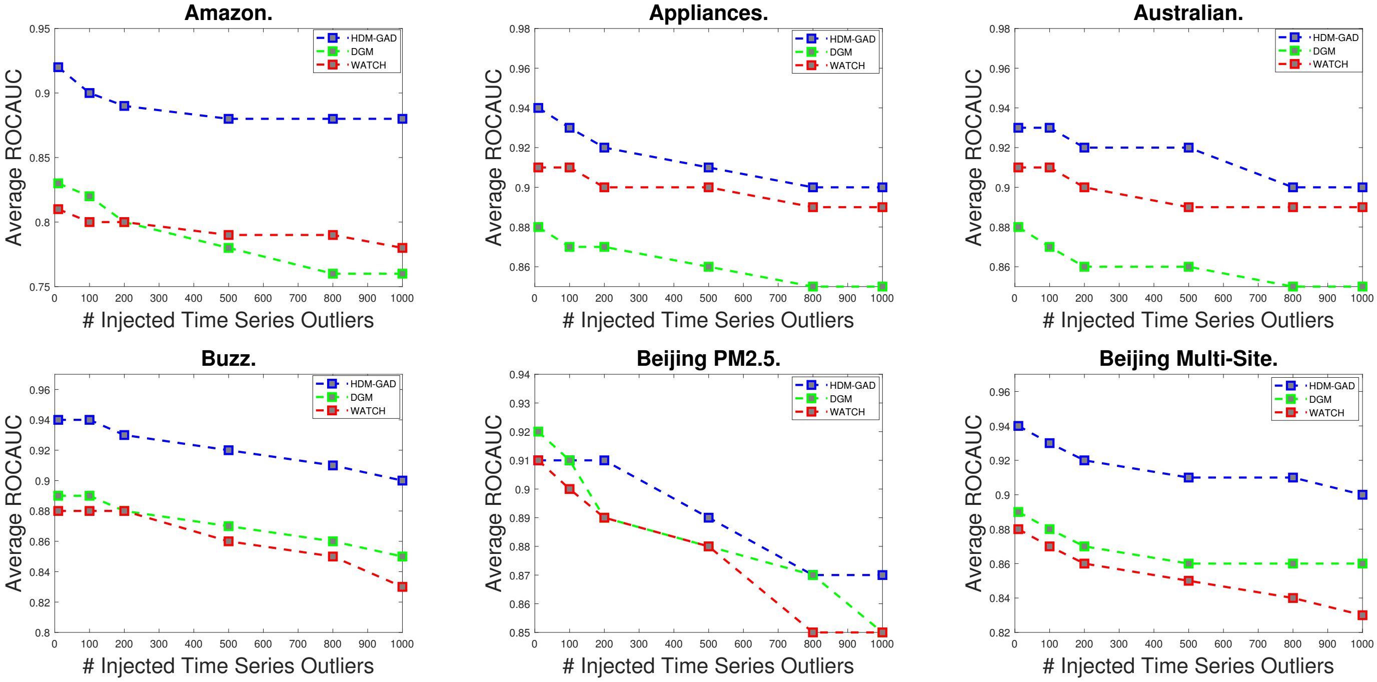

Fig. 5. Accuracy of the HDM-GAD and the baseline group outlier detection algorithms using time series databases.

1) **Sequence pattern mining:** Our study reveals different levels of dependencies among sequence data. High correlated sequence data is sharing a large number of points. Using sequential pattern-mining techniques and examining the identified patterns with sequence data outlier identification is a difficult challenge that may enhance the quality of the returned outliers.
2) **Advanced methods:** Dealing with advanced methods, there have been a lot of adaptations investigated dealing with specific scenarios such as graph data, time series, or trajectory data. All of these user-dependent scenarios deal with working on sequence databases. Therefore, tackling these issues is relevant to an adaptation of group sequence data to tackle different situations at the same time.
3) **High-Performance Computing:** Existing outlier identification approaches for sequence data are computationally costly, especially as the number of

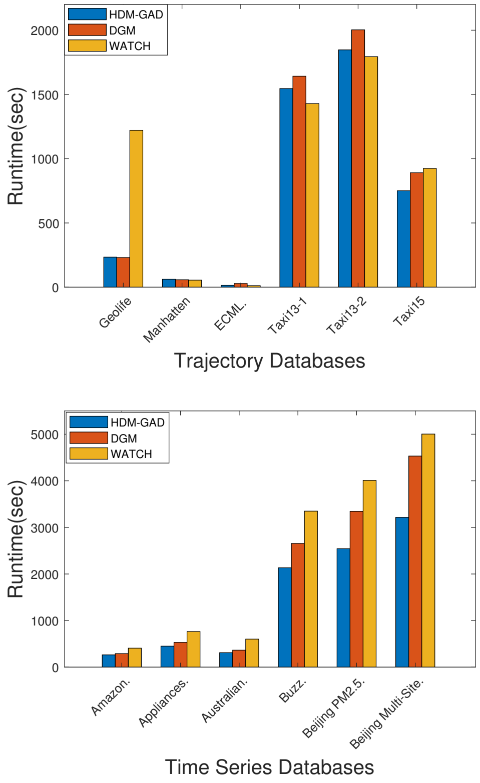

Fig. 6. Runtime of the HDM-GAD and the baseline group outlier detection algorithms using trajectory, and time series databases.

TABLE IV
COMPARISON OF THE RUNTIME(S), SPEEDUP, AND THE ACCURACY OF THE GHDM-GAD, AND THE STATE-OF-THE-ART GPU-BASED GROUP DETECTION ALGORITHMS.

		GHDM-GAD			GkNN			modifiedEFM		
#Blocks	Database	CPU	Speed.	Acc.	CPU	Speed.	Acc.	CPU	Speed.	Acc.
256	D1	25	11	81	35	9	75	39	8	72
	D2	31	12	80	36	11	78	34	7	75
	D3	36	15	79	42	11	77	45	9	74
	D4	52	16	71	55	13	70	58	12	68
	D5	55	18	76	63	10	61	66	10	59
	D6	59	49	71	66	23	68	60	33	64
	D7	11	31	84	19	8	81	18	10	83
	D8	11	39	81	22	15	79	22	12	78
	D9	15	30	71	22	16	70	28	14	70
	D10	14	50	83	21	19	81	32	9	75
	D11	21	56	81	32	29	79	37	16	80
	D12	20	51	79	39	13	80	37	18	77
512	D1	19	20	81	31	11	75	36	10	72
	D2	24	33	80	31	15	78	39	9	75
	D3	28	40	79	35	16	77	39	15	74
	D4	35	59	71	39	42	70	44	48	68
	D5	41	44	76	59	19	61	51	22	59
	D6	39	22	71	77	11	68	76	14	64
	D7	22	5	84	26	3	81	24	3	83
	D8	25	7	81	29	6	79	33	5	78
	D9	29	9	71	33	7	70	35	7	70
	D10	28	11	83	33	9	81	36	6	75
	D11	35	15	81	42	11	79	42	10	80
	D12	41	20	79	45	9	80	46	11	77
1,024	D1	5	72	81	20	43	75	22	40	72
	D2	2	101	80	9	73	78	11	71	75
	D3	1	181	79	11	91	77	16	94	74
	D4	1	204	71	4	111	70	9	81	68
	D5	2	298	76	8	91	61	11	79	59
	D6	1	311	71	7	102	68	5	164	64
	D7	2	151	84	11	80	81	19	52	83
	D8	3	141	81	11	62	79	12	71	78
	D9	4	80	71	13	51	70	11	49	70
	D10	4	218	83	8	148	81	11	150	75
	D11	3	305	81	18	117	79	112	109	80
	D12	2	407	79	15	119	80	22	91	77

TABLE V
SPEEDUP VS GPU ARCHITECTURES.

Database	Tesla C2075	K20	T4
Geolife	152	166	187
Manhattan	164	171	192
ECML PKDD 2015 Competition	186	201	211
Taxi13-1	289	311	325
Taxi13-2	312	325	338
Taxi15	341	348	355
Amazon Access Samples	177	184	192
Appliances energy prediction	151	158	166
Australian Sign Language signs	86	92	98
Buzz in social media	293	315	322
Beijing PM2.5 Data	311	319	330
Beijing Multi-Site Air-Quality Data	451	462	467

points is increased. To deal with large and big sequence databases, high-performance computing tools should be investigated. Several questions, however, must be addressed. Which architectures, for example, should be used? How can we efficiently divide the data across the many jobs? How can we build a parallel method while addressing high-performance computing issues including lowering communication and synchronization costs, enhancing load balancing, and optimizing memory management? In this research work, wrap divergence is taking into account, however, to reach mature solutions, all previous issues should be well studied.

4) **Metaheuristics:** Several metaheuristic-based methods for outlier detection difficulties have been presented. Some works are based on evolutionary algorithms [35]–[37] and other are based on swarm intelligence algorithms [38]–[40]. Adopting these techniques to sequence data for identifying the group of sequence data outliers is an open research issue. Several questions should be addressed in this context, i) How should the solution space be defined? Each candidate group is regarded as a solution; the issue here is to define a suitable representation of the candidate group of sequence data in order to conduct the various metaheuristic operators effectively such as crossover, mutation, local search, and determination of regions. ii) How to explore the candidate's space of group sequence data outliers? It is important to explore the candidate's space of the group sequence data outliers efficiently to find the group of sequence data outliers. Furthermore, the metaheuristics approach consists primarily of two components: the exploitation search, which allows focusing on exploring a local region for good solutions, and the exploration, which aims to generate diverse solutions to explore the entire space on a global scale [41]. The aim here is to offer intelligently specified operators for sequence data while also exploring the sequence data while adhering to both exploitation and exploration requirements. Several metaheuristic-based methods for outlier detection difficulties have been presented. Some works are based on evolutionary algorithms [35]–[37] and other are based on swarm intelligence algorithms [38]–[40]. Adopting these techniques to sequence data

for identifying the group of sequence data outliers is an open research issue. Several questions should be addressed in this context., i) How to define the solution space? Each group candidate is considered as a solution, the challenge here is to define a good representation of the candidate group of sequence data to efficiently perform the different metaheuristic operators such as crossover, mutation, local search, and determination of regions. ii) How to explore the candidate's space of group sequence data outliers? It is important to explore the candidate's space of the group sequence data outliers efficiently to find the group of sequence data outliers. Furthermore, the metaheuristics technique primarily consists of two components: the exploitation search, which enables concentrating on investigating a local region for excellent solutions, and the exploration, which seeks to produce diverse solutions to investigate the whole space on a global scale. The aim here is to offer intelligently specified operators for sequence data while also exploring the sequence data while adhering to both exploitation and exploration requirements.

5) **Missing Ground Truth:** In assessing outlier identification systems, missing the ground truth is a typical issue. The following difficulties and research topics might be identified as obstacles for future study on the element of quality evaluation of outlier identification findings: (i). It is advantageous to define meaningful, publicly available benchmark data for group sequence data outlier detection problems in order to analyze the group sequence data outlier detection methods; (ii). Identifying relevant criteria for an internal review of a group of sequence data outlier identification would be very beneficial. One solution to this difficult problem is to give uniform ranking-function scores to rate the collection of sequence data outliers. These functions should be decoupled from the overall process of locating outliers in a set of sequence data.

V. Future Work

From a future perspective, we plan to propose other efficient approaches for solving the group sequence data outlier detection problem by exploring advanced machine learning techniques such as graph neural networks [42], recurrent neural network [43], and reinforcement learning [44]. Investigating other applications of the group sequence data outlier detection problem such as climate change analysis, and blockchain technology is also on our future agenda. There is also ample room within this research to improve its efficiency. For example, recently we have seen a thrust of papers and research that deal with the sustainable nature of computational works in data mining and anomaly detection. While this work has focused on anomaly detection, doing so in a computationally and energy-efficient manner may be beneficial to push the scope of this research to edge nodes in networks. Moreover, the use of methodologies like Federated Learning combined with the anomaly detection work presented here might pique the interest of researchers looking at having end devices be more active in data mining problems instead of leaving most of the computational efforts to central servers and centrally based databases. A combination of federated learning, anomaly detection, and pattern mining would be a novel future direction that would be worth exploration.

VI. Conclusion

A new hybrid data mining framework is introduced in this paper for retrieving the group outliers from sequence data. The research study starts by proposing the CPU-based version which first determines the micro-clusters using *DBSCAN*, then computes the candidates of groups of sequence data outliers using the kNN. It also studies the different correlations among the candidates of groups of sequence data outliers to retrieve the final groups of sequence data outliers using the pattern mining process. To handle big datasets, the GPU-based version is investigated by exploring the massive computation of the GPU threads. To show the proposed framework's use and efficiency, numerous experiments were conducted on two different types of sequence data (trajectory and time series data). The experimental findings demonstrate the parallel approach's scalability when compared to the sequential version, with a speedup of up to 451 when working with big datasets. The results also reveal the usefulness of exploring hybrid data mining in retrieving the group of sequence data outliers. Thus, our solution outperforms the state-of-the-art group detection algorithms. From a future perspective, we plan to propose other efficient approaches for solving the group sequence data outlier detection problem by exploring advanced machine learning techniques such as graph neural networks [42], recurrent neural network [43], and reinforcement learning [44]. Investigating other applications of the group sequence data outlier detection problem such as climate change analysis, and blockchain technology is also on our future agenda.

Acknowledgment

This paper is partially supported by the Western Norway University of Applied Sciences, Norway, Bergen.

References

[1] Y. Djenouri, D. Djenouri, and J. C. W. Lin, "Trajectory outlier detection: New problems and solutions for smart cities," *ACM Transactions on Knowledge Discovery from Data*, vol. 15, no. 2, pp. 1–28, 2021.

[2] Y. H. Ke, J. W. Huang, W. C. Lin, and B. P. Jaysawal, "Finding possible promoter binding sites in dna sequences by sequential patterns mining with specific numbers of gaps," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2020.

[3] P. O. Prakash and A. Jaya, "Ws-bd-based two-level match: Interesting sequential patterns and bayesian fuzzy clustering for predicting the web pages from weblogs," *The Computer Journal*, vol. 63, no. 2, pp. 322–336, 2020.

[4] X. Zhou, X. Xu, W. Liang, Z. Zeng, S. Shimizu, L. T. Yang, and Q. Jin, "Intelligent small object detection based on digital twinning for smart manufacturing in industrial cps," *IEEE Transactions on Industrial Informatics*, 2021.

[5] S. Peng and A. Yamamoto, "Mining disjoint sequential pattern pairs from tourist trajectory data," in *International Conference on Discovery Science*, 2020, pp. 645–658.

[6] S. Ucar, T. Higuchi, C. H. Wang, D. Deveaux, J. Härri, and O. Altintas, "Vehicular knowledge networking and application to risk reasoning," in *Proceedings of the Twenty-First International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2020, pp. 351–356.

[7] X. Zhou, W. Liang, S. Shimizu, J. Ma, and Q. Jin, "Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5790–5798, 2020.

[8] Y. Zuo, Y. Wu, G. Min, C. Huang, and K. Pei, "An intelligent anomaly detection scheme for micro-services architectures with temporal and spatial data analysis," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 2, pp. 548–561, 2020.

[9] C. Huang, G. Min, Y. Wu, Y. Ying, K. Pei, and Z. Xiang, "Time series anomaly detection for trustworthy services in cloud computing systems," *IEEE Transactions on Big Data*, 2017.

[10] Z. Liu, D. Pi, and J. Jiang, "Density-based trajectory outlier detection algorithm," *Journal of Systems Engineering and Electronics*, vol. 24, no. 2, pp. 335–340, 2013.

[11] Y. Zheng, "Trajectory data mining: an overview," *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 3, pp. 29:1–29:41, 2015.

[12] K. Singh and S. Upadhyaya, "Outlier detection: applications and techniques," *International Journal of Computer Science Issues*, vol. 9, no. 1, p. 307, 2012.

[13] Y. Yu, Y. Zhu, S. Li, and D. Wan, "Time series outlier detection based on sliding window prediction," *Mathematical problems in Engineering*, vol. 2014, p. 879736, 2014.

[14] K. Yamanishi and J. i. Takeuchi, "A unifying framework for detecting outliers and change points from non-stationary time series data," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2002, pp. 676–681.

[15] K. Yamanishi, J. I. Takeuchi, G. Williams, and P. Milne, "On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms," *Data Mining and Knowledge Discovery*, vol. 8, no. 3, pp. 275–300, 2004.

[16] C. Xie, Z. Chen, and X. Yu, "Sequence outlier detection based on chaos theory and its application on stock market," in *International Conference on Fuzzy Systems and Knowledge Discovery*, 2006, pp. 1221–1228.

[17] N. Nesa, T. Ghosh, and I. Banerjee, "Non-parametric sequence-based learning approach for outlier detection in iot," *Future Generation Computer Systems*, vol. 82, pp. 412–421, 2018.

[18] G. S. Na, D. Kim, and H. Yu, "Dilof: Effective and memory efficient local outlier detection in data streams," in *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1993–2002.

[19] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Outlier detection for time series with recurrent autoencoder ensembles," in *International Joint Conferences on Artificial Intelligence*, 2019, pp. 2725–2732.

[20] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. V. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1409–1416.

[21] L. Feremans, V. Vercruyssen, B. Cule, W. Meert, and B. Goethals, "Pattern-based anomaly detection in mixed-type time series," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2019, pp. 240–256.

[22] A. Belhadi, Y. Djenouri, G. Srivastava, D. Djenouri, A. Cano, and J. C.-W. Lin, "A two-phase anomaly detection model for secure intelligent transportation ride-hailing trajectories," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[23] A. R. Javed, M. Usman, S. U. Rehman, M. U. Khan, and M. S. Haghighi, "Anomaly detection in automated vehicles using multistage attention-based convolutional neural network," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[24] Y. Wang, N. Masoud, and A. Khojandi, "Real-time sensor anomaly detection and recovery in connected automated vehicle sensors," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1411–1421, 2020.

[25] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Pacific-Asia conference on knowledge discovery and data mining*, 2013, pp. 160–172.

[26] G. Gupta, A. Liu, and J. Ghosh, "Automated hierarchical density shaving: A robust automated clustering and visualization framework for large biological data sets," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 7, no. 2, pp. 223–237, 2008.

[27] Y. Xie and S. Shekhar, "Significant dbscan towards statistically robust clustering," in *The International Symposium on Spatial and Temporal Databases*, 2019, pp. 31–40.

[28] J. Pei, J. Han, and R. Mao, "Closet: An efficient algorithm for mining frequent closed itemsets," in *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, vol. 4, no. 2, 2000, pp. 21–30.

[29] S. M. Qaisar, "Efficient mobile systems based on adaptive rate signal processing," *Computers & Electrical Engineering*, vol. 79, p. 106462, 2019.

[30] A. Prokhorchuk, J. Dauwels, and P. Jaillet, "Estimating travel time distributions by bayesian network inference," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 5, pp. 1867–1876, 2020.

[31] R. Chalapathy, E. Toth, and S. Chawla, "Group anomaly detection using deep generative models," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2018, pp. 173–189.

[32] J. Li, J. Zhang, N. Pang, and X. Qin, "Weighted outlier detection of high-dimensional categorical data using feature grouping," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 11, pp. 4295–4308, 2020.

[33] F. Wang, Y. Lei, Z. Liu, X. Wang, S. Ji, and A. K. Tung, "Fast and parameter-light rare behavior detection in maritime trajectories," *Information Processing & Management*, vol. 57, no. 5, p. 102268, 2020.

[34] N. Upasani and H. Om, "A modified neuro-fuzzy classifier and its parallel implementation on modern gpus for real time intrusion detection," *Applied Soft Computing*, vol. 82, p. 105595, 2019.

[35] M. Gupta, J. Gao, Y. Sun, and J. Han, "Integrating community matching and outlier detection for mining evolutionary community outliers," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012, pp. 859–867.

[36] H. Wang, J. Wu, W. Hu, and X. Wu, "Detecting and assessing anomalous evolutionary behaviors of nodes in evolving social networks," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 1, pp. 1–24, 2019.

[37] A. Thakkar and R. Lohiya, "Role of swarm and evolutionary algorithms for intrusion detection system: A survey," *Swarm and Evolutionary Computation*, vol. 53, p. 100631, 2020.

[38] M. Asadi, M. A. J. Jamali, S. Parsa, and V. Majidnezhad, "Detecting botnet by using particle swarm optimization algorithm based on voting system," *Future Generation Computer Systems*, vol. 107, pp. 95–111, 2020.

[39] J. Ge, S. Wang, H. Dong, H. Liu, D. Zhou, S. Wu, W. Luo, J. Zhu, Z. Yuan, and H. Zhang, "Real-time detection of moving magnetic target using distributed scalar sensor based on hybrid algorithm of particle swarm optimization and gauss-newton method," *IEEE Sensors Journal*, vol. 20, no. 18, pp. 10 717–10 723, 2020.

[40] L. Guo, "Research on anomaly detection in massive multimedia data transmission network based on improved pso algorithm," *IEEE Access*, vol. 8, pp. 95 368–95 377, 2020.

[41] Y. Djenouri and M. Comuzzi, "Combining apriori heuristic and bio-inspired algorithms for solving the frequent itemsets mining problem," *Information Sciences*, vol. 420, pp. 1–15, 2017.

[42] Y. Wu, H. N. Dai, and H. Tang, "Graph neural networks for anomaly detection in industrial internet of things," *IEEE Internet of Things Journal*, 2021.

[43] J. C. W. Lin, Y. Shao, Y. Djenouri, and U. Yun, "Asrnn: a recurrent neural network with an attention model for sequence labeling," *Knowledge-Based Systems*, vol. 212, p. 106548, 2021.

[44] C. Huang, Y. Wu, Y. Zuo, K. Pei, and G. Min, "Towards experienced anomaly detector through reinforcement learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.