Anomaly Detection on Roads Using C-ITS Messages

Juliet Chebet Moso, Ramzi Boutahala, Brice Leblanc, Hacene Fouchal, Cyril de Runz, Stephane Cormier, John Wandeto

Anomaly Detection On Roads Using C-ITS Messages

Juliet Chebet Moso[1,3], Ramzi Boutahala[1], Brice Leblanc[1], Hacène Fouchal [1], Cyril de Runz [2], Stephane Cormier [1], and John Wandeto[3]

(1) CReSTIC EA 3804, 51097, Université de Reims Champagne-Ardenne, Reims, France
(2) BDTLN, LIFAT, University of Tours, Tours, France
(3) Dedan Kimathi University of Technology, Nyeri, Kenya

Abstract. Cooperative Intelligent Transport Network is one of the most challenging issue in networking and computer science. In this area, huge amount of data are exchanged. Smart analysis of this collected data could be achieved for many purposes: traffic prediction, driver profile detection, anomaly detection, etc. Anomaly detection is an important issue for road operators. An anomaly on roads could be caused by various reasons: potholes, obstacles, weather conditions, etc. An early detection of such anomalies will reduce incident risks such as traffic jams, accidents. The aim of this paper is to collect message exchanges between vehicles and analyze trajectories. This analysis becomes difficult since a privacy principle is applied in the case of C-ITS. Indeed, each message sent is generated with an identifier of the sender. This identifier is kept only over a specified time interval thus one vehicle will have multiple identifiers. We first have to solve Trajectory-User Linking problem by chaining anonymous trajectories to potential vehicles by considering similarity in movement patterns. After that we apply various methods to check variations of trajectories from normal ones. When we observe some differences, we can raise an alarm about a potential anomaly. In order to check the validity of this work, we generated a large amount of messages exchanges by many vehicles using the Omnet ++simulator together with the Artery, Sumo plug-in. We applied various variations on some obtained trajectories. Finally, we ran our detection algorithm on the obtained trajectories using different parameters (angles, speed, acceleration) and obtained very interesting results in terms of detection rate.
keywords: Trajectory-User Linking, Moving objects, Similarity measure, Anomaly detection, Data analysis.

1 Introduction

C-ITS eco-system generates a very huge amount of data. The collection of mobility data is done by online or offline means through devices attached/carried by the moving objects, road side units among other techniques. Usually, such data includes details that explain the movement of vehicles. Each trajectory of a

moving entity is considered a multi-attribute, time-ordered sequence of locations traversed by the entity.

Trajectory mining is a process which entails the analysis of movement traces with the main goal being the extraction of spatial, spatial-temporal and behavioral patterns [1]. The main techniques used for this analysis perform classification, clustering, point of interest detection and anomaly detection. We can look at trajectory data mining as a three phased process which includes [2]: data preprocessing, data management (indexing and data storage) and pattern mining. The key drivers can be "economic (logistical optimization, customer behavior analysis, targeted advertising), scientific (animal behavior analysis, healthcare), administrative (urban planning, criminal investigation), or private" [3]. However, there remains a challenge on how to obtain knowledge and information from these data which can assist in mobility improvement [4].

The paths of moving objects on road networks are affected by the environmental and traffic conditions. To gain a better understanding of the movement patterns one needs to incorporate the environmental information in the analysis [5]. Further, to characterize the behavioral and lifestyle aspects of an entity, an analysis of daily trajectories is imperative. Trajectory pattern mining comes in handy in public security systems, recommender systems and path planning in emergency evacuations [6].

To obtain meaningful information from trajectories, the raw points need to be enriched with semantic attributes, which is basically a daunting process. To solve this issue, semantic annotations can be done by experts or users can add semantic labels to their trajectories. We can also label trajectories with points of interest (POIs) using their location information [2], [7]. Working with semantically enriched trajectories enhances querying and pattern identification [8] which simplifies behaviour analysis of the moving object. Trip recommender systems, life experience sharing and context-aware computing are some of the applications which benefit from semantic trajectory analysis [2].

The advances in battery technology and availability of low cost storage devices have facilitated the capture of highly sampled trajectory data over an extended period of time. With the increased data, it is now possible to discover more interesting patterns during pattern mining. Nevertheless, the analysis of raw un-simplified trajectories can be virtually impossible and computationally resource intensive. This can be alleviated by the use of compression and pruning techniques during pattern mining [6].

When reporting their locations to a central repository, moving objects can have various strategies such as time-based, distance-based, and prediction-based strategies. Communication with a central server may also be interrupted for a while and restored later. This results in segmented trajectories with gaps due to missing readings and also variation in trajectory lengths. Also, for privacy reasons, the device identification(ID) numbers which uniquely identify a trajectory may be changed periodically. In order to reconstruct the movement of a vehicle over a long period of time, the device IDs from the consecutive trips must be identified through a linking process and the missing gaps filled.

We propose to solve the Trajectory-User Linking (TUL) problem by chaining anonymous trajectories to potential vehicles by considering similarity in movement patterns. This will be performed as a pre-processing step for the characterization and semantic analysis of moving objects though behavior analysis. Occurrence of obstacles on the road causes the vehicles passing the affected section to exhibit an avoidance behavior which can be viewed as a drift. We investigate the avoidance behavior through observation of movement variations on the obtained trajectories. When some threshold is reached, an anomaly is possible and is detected.

We make the following contributions: (a) we present a detailed state of the art on trajectory linking, trajectory mining, anomaly detection, and identify the open research issues; (b) we investigate trajectory linking problem using a real dataset of messages generated in Cooperative Intelligent Transportation System (C-ITS); (c) we perform anomaly detection based on concept drift on C-ITS messages.

The rest of this paper is structured as follows: Section 2 presents the state of the art investigation on Trajectory-User Linking, trajectory mining and anomaly detection. Section 3 presents the problem statement, methodology and description of the dataset. Section 4 presents the experiments and results, and Section 5 presents the conclusion and future work.

2 Related Works

This section introduces works on Trajectory-User Linking, trajectory mining and anomaly detection.

2.1 Trajectory-User Linking (TUL)

A recent area of research in location-based social network applications (LBSNs) is Trajectory-User Linking [9]. It is driven by the huge volume of data generated in these applications. To preserve privacy in LBSNs user identifiers are removed from the data during anonymization. Conversely, the ability to link the trajectories to the real users through analysis of check-in data and phone signals can be very useful in recommender and criminal identification systems. Due to the abundance of user classes and the sparsity of data, solving TUL is a challenging task. In [9], a semi-supervised learning model based on Recurrent Neural Networks (RNN), called TULER (TUL via Embedding and RNN) is proposed. TULER learns the semantic mobility patterns of spatio-temporal data by correlating trajectories to the users who generated them. It identifies the dependencies inherent in check-in data and infers hidden user patterns.

Additionally, TULVAE (TUL via Variational AutoEncoder), a semi-supervised learning technique is presented in [10]. TULVAE applies a neural generative architecture with stochastic latent variables in the analysis of geo-tagged social media data. It considers the fact that human trajectories exhibit a hierarchical

semantic structure with high-dimensionality and data sparsity. Through processing vast quantities of unlabeled data, TULVAE tackles the data sparsity issue, thus generating useful knowledge and distinct mobility patterns.

The proliferation of location based services has resulted in the availability of heterogeneous mobility data from the various service providers. There is also a growing need for a better understanding of user behaviour across multiple services. To deal with the data heterogeneity issue, DPLink an end-to-end deep learning based framework for performing user identity linkage is proposed in [11]. It extracts representative features from the trajectory using a feature extractor, a location encoder and a trajectory encoder. The decision to link two trajectories as the same user is made using a comparator. The low-quality problem of mobility data is handled by a multi-modal embedding network and a co-attention mechanism in DPLink.

2.2 Trajectory mining

Moving objects can be categorized into various classes based on their trajectories through trajectory classification. The aim of classification is to identify modes of transport, vessel types or user classes based on trajectory patterns [12]. The classification process is a three step process [2]: (i) Trajectory segmentation, (ii) Feature extraction from the segments, and (iii) Building of the classification model. The process requires as input a sequence of spatio-temporal points.

Clustering is one of the classification techniques applicable to trajectories where the clusters formed should have a low inter-class similarity and a high intra-class similarity. The output of clustering, especially in relation to behavior prediction can be applied in destination prediction, urban planning, market research and location recommendation [13]. The open research issues include: finding appropriate features for trajectory representation, similarity measures and development of algorithms for spatial data clustering [14]. The key challenge is how to identify relevant class distinguishing features and how to select the most discriminate features to be used in building the classification model[15]. One of the frequently used discriminant features is the distance between two trajectories which is computed using a distance measure or metric.

In evaluating user similarity, several literature studies focus on the geometric or sequential features of trajectories. Trajectory similarity is measured based on the co-location frequency (feature-based representations), which is the number of times two moving objects appear spatially close to one another. Other measures include subsequence similarity metrics such as the length of the Longest common subsequence (LCSS) [16], Edit Distance on Real Sequences (EDR) [17], Common Visit Time Interval (CVTI) [18], Maximal Semantic Trajectory Pattern (MSTP) [19], Multidimensional Similarity Measure (MSM) [20], and Stops and Moves Similarity Measure (SMSM) [21].

By defining distance and matching thresholds, LCSS reduces the effect of noisy data. When distance in LCSS is less than a given threshold in all dimensions, two points match. However, LCSS ignores gaps in sequences, resulting in the same similarity value for different pairs of trajectories for some problems.

EDR uses an edit distance to calculate the similarity between elements where all dimensions are taken into account for a match to occur. Penalties are assigned based on the length of the gaps between two matched sub-sequences resulting in more precise results than LCSS. The semantic dimension of stops is integrated by CVTI with the temporal dimension. It does not allow heterogeneous data to be modeled and calculated together, such as stops and moves.

During similarity analysis of semantic trajectories, MSTP considers the frequency at which stops are visited. It does not consider moves between stops and multiple data dimensions. The similarity rating in MSM is based on the matching scores of all pairs with at least one matching dimension. It allows definition of different similarity weights for every dimension and may assign a high score to trajectories with similarity only in a small portion of their length. SMSM considers both stops and moves within the trajectory and, by assigning weights, performs partial dimension matching and partial stop ordering. However, for users, calculating weights can be difficult.

LCSS and EDR require all elements to match across all dimensions when looking at the applicability of similarity measures based on trajectory dimensions, whereas MSM considers matching pairs in a single dimension. In situations where the trajectory contains outliers LCSS, EDR, MSM and SMSM which are robust to noise are applicable. MSM and SMSM are good options when dealing with semantic trajectories, though LCSS and EDR can be extended for semantic trajectory mining. The best measure is MSM when considering apps that use GPS trajectories annotated with stops only or trajectories extracted from social media, as it manages sparse data. MSM is useful when one wants to find users who visited the same place at similar times irrespective of the order of visits. When extracting the most similar paths or most popular routes between stops, SMSM is applicable.

2.3 Anomaly detection

Anomalies can be defined as "patterns in data that do not conform to a well-defined notion of normal behavior" [22]. These patterns can also be referred to as outliers or exceptions, and represent new, rare or unknown data which may be of interest in a specific domain. In the presence of labeled data, anomaly detection can be done using supervised learning techniques where it is considered a binary classification problem with data instances being either normal or abnormal. However, this is rarely the case due to the limitation in availability of labeled data and the fact that the anomalous events are quite rare. Due to availability of massive amounts of unlabeled data, most anomaly detection approaches adopt unsupervised learning techniques.

Anomalies can be viewed in two ways: (i) erroneous data generated due to device failure or system faults, and (ii) unusual data representing rare/exceptional activities/events which are anomalous but actually happened [23].Some of the anomalies linked to road networks include: vehicle collisions, vehicle breakdowns, debris on the road, pot holes, and vehicle(s) stopped in the middle of the road. Most of these can be attributed to driving behavior and the status of the road.

The main aim of traffic management is to reduce the number of anomalies and improve traffic flow. It is desirable to know the locations, time and frequency of occurrence of these anomalies for efficient traffic management.

Anomaly detection techniques are usually focused at identifying patterns which do not conform to expected behavior. However, according to [24] the challenge lies in the fact that: (i) there is no well-defined boundary between what is normal and what is considered abnormal; (ii) there is a high possibility of a normal behavior evolving to an abnormal representation in the future; (iii) it is difficult to apply anomaly detection techniques developed in one field to another field due to difference in applications and concepts; (iv) presence of noise in the data makes it difficult to distinguish between noise points and the real anomalous points.

3 Problem Statement and Methodology

3.1 Problem Statement

The vehicles of a Cooperative Intelligent Transport Network (C-ITS) exchange a lot of messages. Each message contains an identifier of the transmitting vehicle. In order to protect the privacy of users, each vehicle's identifiers are updated periodically. Given the various identifiers assigned to a vehicle, we wish to investigate whether it is possible to group all identifiers which belong to the same vehicle. We adopt the definition of [9] for Trajectory User-Linkability problem:

Let $T_{vi} = m_{i1}, m_{i2}, \ldots, m_{in}$ denote a trajectory generated by the vehicle v_i during a time interval, where $m_{ij}(j \in [1, n])$ is a message sent from a specific location at time t_j. Given that the identifier is changed after a time period, trajectory $T_x = m_1, m_2, \ldots, m_y$ generated by the same vehicle in the next time interval with a different identifier is considered unlinked. TUL can thus be defined as:

Suppose we have a number of unlinked trajectories $T = t_1, \ldots, t_m$ generated by a set of vehicles $V = v_1, \ldots, v_n(m \gg n)$, TUL learns a function that links unlinked trajectories to the vehicles: $T \rightarrow V$

Information on the presence of obstacles on the road is useful to road operators as it can enhance road safety through planned interventions to treat them. We intend to detect anomalies which are as a result of stopped cars and potholes on the road.

3.2 Methodology

In a C-ITS environment cooperative awareness is achieved through exchange of CAMs which contain position information. This can serve as a privacy risk especially in a scenario where an eavesdropper is able to recreate a comprehensive mobility pattern of the driver. In order to lessen the risk, pseudonyms are used to provide anonymous communication. To ensure unlinkability of actions, multiple pseudonyms are used per vehicle [25].This involves the periodic change

of pseudonyms so as to prevent linkability of one pseudonym to another which can in turn result in the identity of a vehicle and consequently that of the driver being revealed if one is able to identify the home address.

We begin by grouping as much as possible the different identifiers which represent sub-trajectories of one vehicle. A complete grouping with all the identifiers of each vehicle may be difficult to obtain but grouping some identifiers can be achieved. For example, if the last message of an identifier is spatially and temporally close to the first message obtained with another identifier and the change in attributes like speed and heading angle is consistent, then the change of identifier from the last message to the first one is obtained for the same vehicle. Thus the two identifiers are linked and belong to the same vehicle. In this example, the work consists in defining a reliable link between two messages with different identifiers.

Then we detect the contradictions between messages. For instance, if two messages give the same localization at the same time, then their identifiers cannot belong to the same vehicle. These contradictions help to define the group of identifiers for each vehicle by rejecting the identifiers leading to a contradiction. The framework to be followed in the analysis is shown in Fig.1.

Definition: *Trajectory*: A raw trajectory consists of a sequence of n points $T = [p_1, p_2, \ldots, p_n]$, in which $p_i = x, y, z, t, A$, where x, y ,z represent the position of the moving object in space, t is the timestamp and A represents other attributes associated with the point (i.e. speed, heading angle and drive direction)

In this study a trajectory is considered as the consolidation of messages uniquely identified by a single identifier. The second step is to detect obstacles on the road where we are interested in concept drifts with a sudden appearance (stopped vehicle scenario) and those of a slow appearance (growing pothole scenario). The assumption is that an obstacle will block the whole lane requiring the other vehicles to change lane as they avoid it.

3.3 Dataset Description

In our study we used a real dataset of Cooperative Awareness Messages (CAM) generated by the OMNET simulator together with SUMO, artery plug-in. A vehicle sends CAMs to its neighbourhood using Vehicle-to-Vehicle (V2V) or

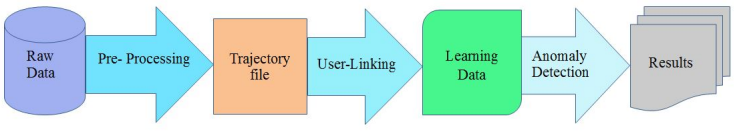

Fig. 1. Trajectory mining framework.

Vehicle-to-Infrastructure (V2I) communications. The frequency of CAM message generation varies from 10Hz to 1Hz (100 milliseconds to 1000 milliseconds). Each CAM is uniquely defined by a stationid (Pseudonym) and timestamp. In this dataset each vehicle has a defined stationid which changes periodically in order to guarantee privacy of drivers.

In this study, each message is defined by eight variables: an identifier associated with the transmitting vehicle, a timestamp, the location of the vehicle (latitude, longitude and altitude), speed, heading angle and the drive direction. The speed, heading angle and drive direction variables are used as descriptive variables of the behavior of the transmitting vehicle.

4 Performance evaluation

In order to link the trajectories we consider the following conditions for triggering CAM generation as specified in ETSI EN 302 637-2 standard [26]:

- If the absolute difference between the current heading value of the vehicle and the heading value included in the last transmitted CAM by the same vehicle exceeds 4 degrees;
- If the distance between the current position of the vehicle and the position included in the last transmitted CAM by the same vehicle exceeds 4 metres;
- If the absolute difference between the current speed of the vehicle and the speed included in the last transmitted CAM by the same vehicle exceeds 0.5 m/s.

We performed trajectory mining using PostgreSQL database with the spatial extension PostGIS used for storing and processing spatial data. We also used Quantum GIS (QGIS) an open-source cross-platform desktop geographic information system application that supports viewing, editing, and analysis of geospatial data. QGIS was majorly used for visualization and map matching of the trajectories as a validation step.

Considering the fact that each vehicle was assigned multiple identifiers, we sort out to group identifiers which occurred on the same day by comparing origin and destination pairs. Taking the destination points, we extracted the nearest origin point within 170 meters (since the highest speed recorded in the dataset was 163m/s) and also filtered out the results by implementing the CAM generation trigger conditions as additional constraints. The distance computation was done using the *ST_DistanceSpheroid* function in PostgreSQL which gives the linear distance between two longitude/latitude points. We also used the CAM generating frequency of 100 – 1000 milliseconds as a constraint in order to get exact matches in time and space.

4.1 Obstacle detection

In our study we have performed anomaly detection mainly focusing on road obstacle detection. We handle the data collected from vehicles as a data stream.

There can be different kinds of change in a data stream. From time to time an outstanding value appears, this is called an outlier. When the data is changing from one behavior to another, this is called a concept drift. A concept drift detector is designed to find when the data is changing from one concept to another, but it should be robust to outliers so as to avoid false positive detection. Different concept drift detection approaches are used for different kinds of data and streams. The parameters of these algorithms are used to tune the algorithm to avoid a trigger on outliers. However, too restrictive parameters can result in the algorithm not triggering at all. Parameters are varied depending on the context.

To handle data streams, the algorithms can store some global value relative to the stream that are updated for each new data or rely on a window model to store part of the stream and calculate the values on the stored data. In a window model, data is stored until the window is full and since the memory is limited older data will be removed from the window [28]. Here are some window models that can be used:

Sliding window model: In this window model the data is treated in a first-in first-out manner. The size of the window can be fixed or variable but when the window is full, oldest data are deleted so new data can be treated.

Damped window model: This window model associates an exponentially decaying weight to the observations and delete the data at the point when the weight is equivalent to zero.

Landmark window model: The landmark model rely on chunks of data separated by landmarks. A landmark can be a time value (hour, day, month, ...) or a number of observed elements. Every data in the landmark is treated until the next landmark is reached. When it is reached, the old data is removed and replaced by the new one.

In this study, we used two algorithms: Page-Hinkley and ADWIN. These are really popular approaches due to their effectiveness on many types of data, and we aim to know if they are adapted to our type of data:

The Page-Hinkley algorithm [27] [29] analyzes the data sequentially to detect change and does not use a window model since no data is stored except a mean and a sum. It recalculates the mean value of the data at each input. And it also recalculates the sum of the difference to the mean with the alpha and the delta parameters to adjust the sensitivity. The alpha and the delta parameters help to mitigate outliers both in different ways, the greater they are the more outliers will be needed to detect a drift. If this sum passes over the lambda threshold value then a drift signal is raised. The greater the threshold is, the fewer false positives are but actual errors could be missed or the detection delayed. Also the higher the alpha and delta values are the harder it is to detect small variations. Page-Hinkley consumes very few resources since it is not storing any part of the data stream. But its strongest issues are its sensitivity to outliers when trying to detect concept drift on low varying data and its delay on the detection of concept drift when tuned to resist to outliers.

The ADaptative WINdowing (ADWIN) algorithm [30] is based on a sliding window system. The size of the window, instead of being fixed, is recomputed: if a drift is detected, the window is reduced, if not, it is growing to its maximal size defined by the user. To change the window size, it is made into a bucket list that is split in bucket rows of the same size, and these bucket rows contains buckets. The algorithm takes data as input, stores it in a bucket that is put in the last bucket row. If the bucket list is full, the two oldest bucket rows are reduced and merged. The process to detect the drift is triggered every *clock* number of new data, only if the length of the window is greater than the minimal sub-window length. To detect a drift, the buckets are separated in two sub-windows, one containing the oldest data (this one is bigger than the second one) and the other containing newer data. If the data between these two windows are too different (according to the delta value) then a drift signal is raised, and the window size is reduced. ADWIN has a small memory consumption due to its bucket system and can detect quickly concept drift since part of the stream is stored. But since a small part is stored long and slow drift is hard to detect because the buckets are updated with more and more drifting data without noticing it. And if the algorithm is more sensitive to detect such change the rate of false-positive will be higher. In order to detect avoiding behaviors and the change of frequency of them, we use ADWIN, and Page-Hinkley algorithms.

Fig. 2 presents the results for the Page-Hinkley algorithm for the stopped car scenario. And Fig. 3 the result of Page-Hinkley on the pothole scenario. The red dots represent the number of messages with an avoiding behavior (real drift) in the last 600 messages and the blue dots, the number of drifts detected by the algorithm. The x-axis represents the generation time (corresponding to the number of messages). The y-axis represents the number of messages detected that contain an avoiding behavior.The higher on the y-axis the dots are, the stronger the change is on the period.

For the Page-Hinkley algorithm, it is difficult to have an accurate detection of large changes because the detection is highly delayed. That is why we used parameters to detect the smallest changes. This allows us to track the frequency of changes in the overtaking behaviors. We can see that the frequency increases as we enter the period when the overtaking rate is the highest. But by design, the Page-Hinkley algorithm has a certain delay in detecting new concepts, so the points do not directly follow the change. The performance of this algorithm is encouraging for the stopped car scenario since we can see the increase in the number of detections when the change occurs. But the delay in the detection of the events is a strong backlash because we need a filtering step that will delay the detection even more. For the pothole scenario, we can see a slight increase in the detection rate of changes with few spikes but this is not enough to be significant. And with the delay in detection, it is not possible to have a correct view of the detection until the concept stabilizes.

Fig. 4 presents the results for the ADWIN algorithm for the stopped car scenario and Fig. 5 the result for ADWIN on the pothole scenario.

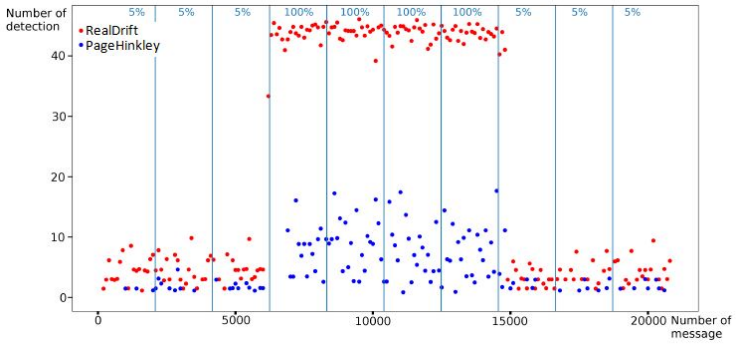

Fig. 2. Detection of Page-Hinkley algorithm for stopped car scenario

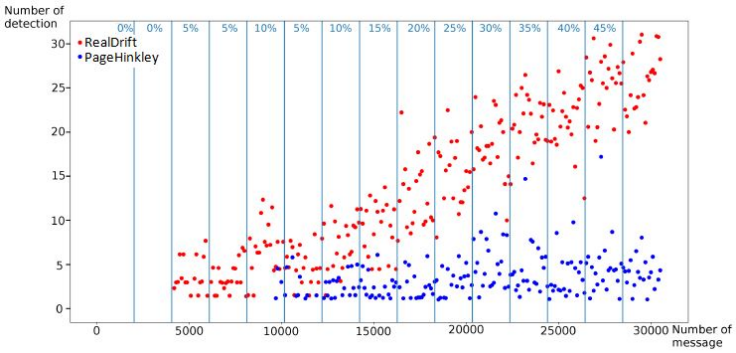

Fig. 3. Detection of Page-Hinkley algorithm for pothole scenario

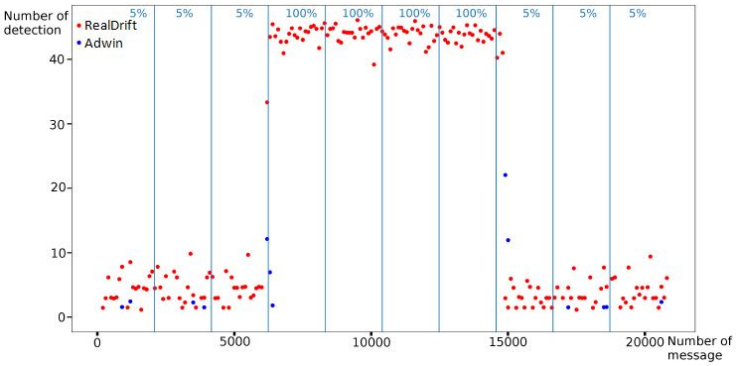

Fig. 4. Detection of ADWIN algorithm for stopped car scenario

For the ADWIN algorithm, in the stopped car scenario, there is little detection in the low avoiding rates, but they do not hide the high number of detections

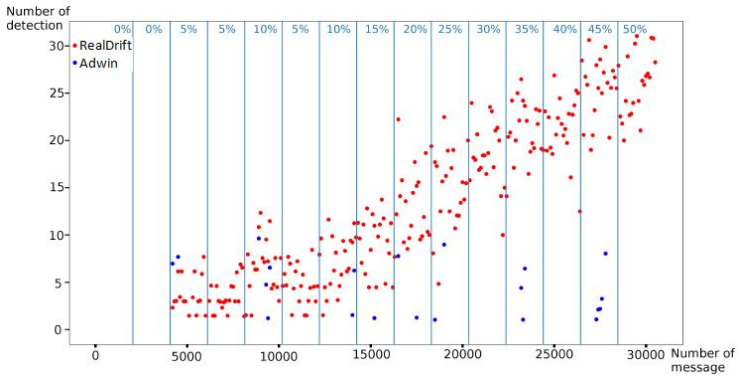

Fig. 5. Detection of ADWIN algorithm for pothole scenario

when large changes occur. In this scenario, the results are really convincing and should be tested in real cases. But in the case of the pothole scenario, the detection is not accurate. Initially, the algorithm manages to detect the change, but when the avoidance rate reaches 25%, the algorithm cannot detect the changes correctly. This is because the algorithm is designed to adapt to changes, and then, it fails to detect the next changes accurately because the avoiding behavior has become part of the concept it has learned and the difference in rates is no longer large enough for it to detect them. For this type of behavior, other algorithms may be better suited. Such algorithms should use a window model with a fixed historical window as a basis for learning since we want to detect abnormal behavior compared to typical behavior. But the loss of adaptability to change will require reconfiguration of the history window each time a change is made to the road, its environment, or driver behavior (the latter change could be due to an increasing number of C-ITS, automated vehicles or other technical improvements or recommendations).

5 Conclusion

In this work we considered the trajectory-linking problem and applied it to messages generated by vehicles in C-ITS in order to detect anomalies described by obstacles on roads. Based on our analysis, if other distinguishing attributes and background information on message generation are taken into account, it is possible to link trajectories to the vehicle which generated them. The detection of anomalies is achieved thanks to data stream analysis. We have shown in this study that such analysis should be done off-line in order to learn the main behavior of the system and later it could be run on-line in order to detect any dis-functioning at any time.

As future work, we plan to semantically enrich the trajectories and perform frequent pattern mining on the data. It is foreseeable that autonomous vehicles will need to communicate with C-ITS enabled vehicles which do not have

embedded cameras. It is then interesting to detect obstacles using C-ITS data and for future works these obstacles could be confirmed by processing images captured by autonomous vehicle cameras.

References

1. S. Shekhar, H. Xiong, and X. Zhou, Eds., *Encyclopedia of GIS*. Cham: Springer International Publishing, 2017.
2. Y. Zheng, 'Trajectory Data Mining: An Overview', *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 3, pp. 1–41, May 2015, doi: 10.1145/2743025.
3. F. Valdés and R. H. Güting, 'A framework for efficient multi-attribute movement data analysis', *The VLDB Journal*, vol. 28, no. 4, pp. 427–449, Aug. 2019, doi: 10.1007/s00778-018-0525-6.
4. F. Alesiani, L. Moreira-Matias, and M. Faizrahnemoon, 'On Learning From Inaccurate and Incomplete Traffic Flow Data', *IEEE Trans. Intell. Transport. Syst.*, vol. 19, no. 11, pp. 3698–3708, Nov. 2018, doi: 10.1109/TITS.2018.2857622.
5. T. Wu, J. Qin, and Y. Wan, 'TOST: A Topological Semantic Model for GPS Trajectories Inside Road Networks', *IJGI*, vol. 8, no. 9, p. 410, Sep. 2019, doi: 10.3390/ijgi8090410.
6. Y. Cao et al., 'Effective spatio-temporal semantic trajectory generation for similar pattern group identification', *Int. J. Mach. Learn. & Cyber.*, Jul. 2019, doi: 10.1007/s13042-019-00973-y.
7. Z. Yan, D. Chakraborty, C. Parent, S. Spaccapietra, and K. Aberer, 'Semantic trajectories: Mobility data computation and annotation', *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 3, p. 1, Jun. 2013, doi: 10.1145/2483669.2483682.
8. A. Nishad and S. Abraham, 'SemTraClus: an algorithm for clustering and prioritizing semantic regions of spatio-temporal trajectories', *International Journal of Computers and Applications*, pp. 1–10, Sep. 2019, doi: 10.1080/1206212X.2019.1655853.
9. Q. Gao, F. Zhou, K. Zhang, G. Trajcevski, X. Luo, and F. Zhang, 'Identifying Human Mobility via Trajectory Embeddings', in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, Melbourne, Australia, Aug. 2017, pp. 1689–1695, doi: 10.24963/ijcai.2017/234.
10. F. Zhou, Q. Gao, G. Trajcevski, K. Zhang, T. Zhong, and F. Zhang, 'Trajectory-User Linking via Variational AutoEncoder', in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, Jul. 2018, pp. 3212–3218, doi: 10.24963/ijcai.2018/446.
11. J. Feng et al., 'DPLink: User Identity Linkage via Deep Neural Network From Heterogeneous Mobility Data', in *The World Wide Web Conference on - WWW '19*, San Francisco, CA, USA, 2019, pp. 459–469, doi: 10.1145/3308558.3313424.
12. F. Vicenzi and L. M. Petry, 'Exploring Frequency-based Approaches for Efficient Trajectory Classification', In *Proceedings of the 35th Annual ACM Symposium on Applied Computing - SAC '20*, March 30-April 3, 2020, pp. 624-631, doi: 10.1145/3341105.3374045.
13. Q. Yu, Y. Luo, C. Chen, and S. Chen, 'Trajectory similarity clustering based on multi-feature distance measurement', *Appl Intell*, vol. 49, no. 6, pp. 2315–2338, Jun. 2019, doi: 10.1007/s10489-018-1385-x.
14. B. A. Sabarish, R. Karthi, and T. Gireeshkumar, 'Clustering of Trajectory Data Using Hierarchical Approaches', in *Computational Vision and Bio Inspired Computing*, vol. 28, D. J. Hemanth and S. Smys, Eds. Cham: Springer International Publishing, 2018, pp. 215–226.

15. C. A. Ferrero, L. O. Alvares, W. Zalewski, and V. Bogorny, 'MOVELETS: exploring relevant subtrajectories for robust trajectory classification', In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing -SAC'18*, Pau, France, 2018, pp. 849–856, doi: 10.1145/3167132.3167225.
16. M. Vlachos, G. Kollios, and D. Gunopulos, 'Discovering similar multidimensional trajectories', in *Proceedings 18th International Conference on Data Engineering*, San Jose, CA, USA, 2002, pp. 673–684, doi: 10.1109/ICDE.2002.994784.
17. L. Chen, M. T. Özsu, and V. Oria, 'Robust and fast similarity search for moving object trajectories', in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data - SIGMOD '05*, Baltimore, Maryland, 2005, p. 491, doi: 10.1145/1066157.1066213.
18. H.-Y. Kang, J.-S. Kim, and K.-J. Li, 'Similarity measures for trajectory of moving objects in cellular space', in *Proceedings of the 2009 ACM symposium on Applied Computing - SAC '09*, Honolulu, Hawaii, 2009, p. 1325, doi: 10.1145/1529282.1529580.
19. J. J.-C. Ying, E. H.-C. Lu, W.-C. Lee, T.-C. Weng, and V. S. Tseng, 'Mining user similarity from semantic trajectories', in *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks - LBSN '10*, San Jose, California, 2010, p. 19, doi: 10.1145/1867699.1867703.
20. A. S. Furtado, D. Kopanaki, L. O. Alvares, and V. Bogorny, 'Multidimensional Similarity Measuring for Semantic Trajectories: Multidimensional Similarity Measuring for Semantic Trajectories', *Trans. in GIS*, vol. 20, no. 2, pp. 280–298, Apr. 2016, doi: 10.1111/tgis.12156.
21. A. L. Lehmann, L. O. Alvares, and V. Bogorny, 'SMSM: a similarity measure for trajectory stops and moves', *International Journal of Geographical Information Science*, vol. 33, no. 9, pp. 1847–1872, Sep. 2019, doi: 10.1080/13658816.2019.1605074.
22. V. Chandola, A. Banerjee, and V. Kumar, Anomaly detection: A survey, *ACM computing surveys (CSUR)*, 41(3), pp.1-58, 2009.
23. X. Wang, A. Fagette, P. Sartelet, and L. Sun, A Probabilistic Tensor Factorization Approach to Detect Anomalies in Spatiotemporal Traffic Activities. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*,pp. 1658-1663, 2019, October, IEEE.
24. H. Wang, M.J. Bah, and M. Hammad, Progress in outlier detection techniques: A survey. *IEEE Access*, 7, pp.107964-108000, 2019.
25. J. Petit, F. Schaub, M. Feiri, and F. Kargl, 'Pseudonym Schemes in Vehicular Networks: A Survey', *IEEE Commun. Surv. Tutorials*, vol. 17, no. 1, pp. 228–255, 2015, doi: 10.1109/COMST.2014.2345420.
26. ETSI E. 302 637-2 V1. 3.1-Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service. ETSI, Sept. 2014.
27. Page, Ewan S. Continuous inspection schemes, Biometrika, vol 41(1/2). pp : 100–115, 1954.
28. Golab, Lukasz and Özsu, M Tamer. Issues in data stream management, ACM Sigmod Record, vol 32(2). pp :5–14, 2003.
29. Gama, João and Sebastião, Raquel and Rodrigues, Pedro Pereira. On evaluating stream learning algorithms, Machine learning, vol 90(3). pp : 317–346, 2013.
30. Bifet, Albert and Gavalda, Ricard. Learning from time-changing data with adaptive windowing, Proceedings of the 2007 SIAM international conference on data mining. pp : 443–448, 2007.