Analyzing Driving Behavior: Towards Dynamic Driver Profiling

Anas Ouardini[1(✉)], Imane El Ouazzany Ech-chahedy[1], Afaf Bouhoute[1], Ismail Berrada[2], and Mohamed El Kamili[3]

[1] Faculty of Science, Sidi Mohamed Ben Abdellah University, Fez, Morocco
anas.ouardini@usmba.ac.ma
[2] Mohammed VI Polytechnic University, SCCS, Ben Guerir, Morocco
[3] Higher School of Technology, Hassan II University of Casablanca, Casablanca, Morocco

Abstract. This paper aims to use driving data to create a profile of the driver behavior, which can be then added as an additional layer to the Local Dynamic Map of the vehicle. The main contribution of the paper consists of using the *Spherical KMeans Clustering*, an unsupervised clustering algorithm for multidimensional datasets, to segment the continuous driving data into multiple segments (*hyperspheres*). Unlike the state of the art, this helps in studying the behavior since all the data will be processed at the same time regardless of the number of features. The generated hyperspheres are an abstract form of the initial numerical values, and can be contribute to a better representation of the driver behavior. We used the UAH Dataset [9] to present the proposed approach, and the cross-validation technique to evaluate the segmentation results.

Keywords: Spherical $KMeans$ clustering · Driver profiling · Local Dynamic Map

1 Introduction

Connectivity is driving the future of transportation. Thanks to V2X technology, which stands for 'vehicle to everything', cars are becoming increasingly connected to each other, to infrastructure, to pedestrians and to the cloud. V2X relies on different communication technologies to realize the vehicle to vehicle, vehicle to infrastructure and vehicle to person communication and create what is known as Cooperative Intelligent Transportation Systems (C-ITS). CITS are defined as transportation system, where the cooperation between the road components enables and provides an ITS service that offers better quality and an enhanced service level [11]. The communication and cooperation allow drivers to share their car settings and driving profiles with other road users, to have enough knowledge about their surrounding environment. This information about the surrounding environment is maintained by vehicles in a Local Dynamic Map (LDM) [4].

LDM is a conceptual data store that includes different types of data, organized in 4 layers [3,5–7]. The first layer (at the bottom) contains static data such

L. Foschini and M. El Kamili (Eds.): ADHOCNETS 2020, LNICST 345, pp. 179–190, 2021.
https://doi.org/10.1007/978-3-030-67369-7_13

as geographical information, the road network, etc. The second layer includes semi-static data such as information on traffic signs (GPS position, content, etc.). The third layer includes semi-dynamic or precisely temporal data such as information on traffic (accident, traffic jam, etc.), traffic light (its value at a given moment), weather, etc. The forth and upper layer has dynamic or highly dynamic data. Information coming out with a high frequency such as the position, speed and driving information of nearby vehicles, etc.

The main idea of this paper is to use driving data to create a profile of the driver behavior, which can be then added as an additional layer to the LDM. The added layer will provide information about driving behavior, and can open up a whole new level of personalized applications when combined with other LDM data. As a first step towards such implementation, this paper focuses on the problem of driver profile generation. It proposes a dynamic and automatic segmentation of the continuous driving data, essentially needed to build an appropriate model of the driver behavior. Through this work, we intend to build a flexible, multi-use and personnel model. Every driver will have a profile that understand his patterns and behaviors and his preferences. Let's take an older driver with Alzheimer disease as an example [1]. A profile generated from tracking his vehicle data will understand his patterns, can be his mind and help him reduce the risk of crashes [2].

The remainder of the paper is organized as follows. Section 2 presents an overview of the proposed approach. The materials and methods used to deal with the data and to implement the proposed approach are presented in Sect. 3. Section 4 presents the validation approach and discusses the obtained results. Finally, some conclusions are drawn in Sect. 5.

2 Overview

In many behavioral studies, behaviors observed to proceed through a finite number of states may be represented using graphs. Driving is a continuous process, which is generally observed and recorded as continuous trajectories or as discrete sequences of measurable vehicle data. Therefore, driver behavior can be represented by a directed weighted graph, with nodes corresponding to observed driving states and edges to the transitions between them.

Driving behavior is analyzed in terms of changes in measurable properties such as position, direction, and speed, which are acquired using different sensors, either mounted within cars, user devices, or received through V2X communication. As the number of sensors increases, so the amount of data will also increase. This data will need more intelligent and automatic approach to deal with it and transform it into useful, easy to interpret information. The goal of the approach presented in this work is to transform the numerical sensor data into a high level abstracted form that will represent the driver behavior and make easier the extraction of his driving patterns.

Figure 1 shows an overview of the steps involved in the proposed approach. The first step consists of analyzing and studying the different driving variables

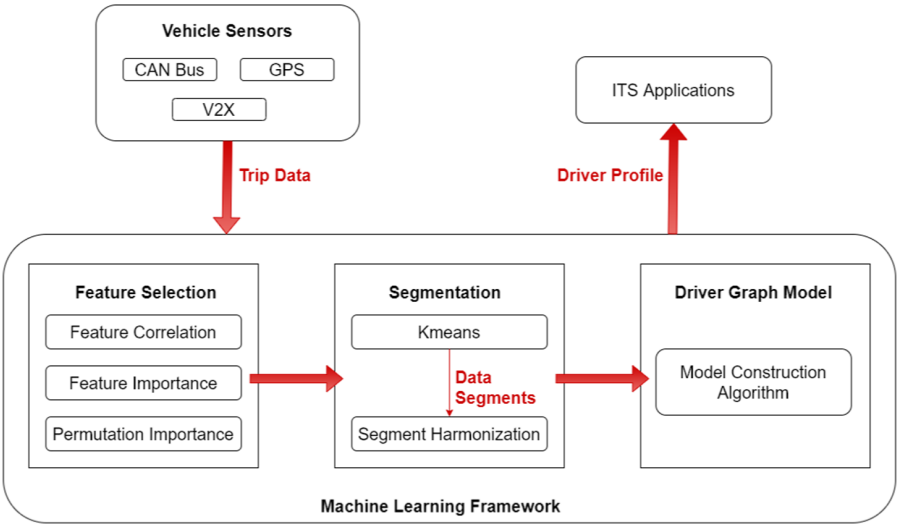


Fig. 1. Overview of the proposed approach

to select important and uncorrelated features. This is mainly performed using three techniques: *Feature Correlation*, *Feature Importance*, *Permutation Importance*. Once feature selected, the next step is segmentation. Segmenting the data prior to building appropriate models is important to effectively use data. This considered as the core of our approach: we propose an automatic, unsupervised, clustering based segmentation. Since driving data is generally organized as trip data, Kmeans algorithm [8] is used to generate K clusters per trip. Harmonization is then used to fuse and update the clusters across all trips. These clusters are finally used to construct the graph representing the driver behavior; the clusters will represent the states of the graph. The transitions between states are weighted and updated according to their occurrence, as in [10]. The generated graph is thus a representation of the driver individual driving patterns and can be used by ITS applications to provide more personalized assistance.

3 Materials and Methods

3.1 Dataset

UAH driveset is a public dataset containing naturalistic driving data captured by a smartphone using the driving monitoring app DriveSafe [12]. The dataset contains plenty of information of 6 different drivers on two different routes. A list of the dataset variables used in this paper are presented below:

1. Timestamp (seconds)
2. Speed (km/h)
3. Latitude coordinate (degrees)

4. Longitude coordinate (degrees)
5. Altitude (meters)
6. Acceleration in X (Gs)
7. Acceleration in Y (Gs)
8. Acceleration in Z (Gs)
9. Acceleration in X filtered by KF (Gs)
10. Acceleration in Y filtered by KF (Gs)
11. Acceleration in Z filtered by KF (Gs)
12. Roll (degrees)
13. Pitch (degrees)
14. Yaw (degrees)
15. X: car position relative to lane center (meters)
16. Phi: car angle relative to lane curvature (degrees)
17. W: road width (meters)
18. Distance to ahead vehicle in current lane (meters)
19. Time of impact to ahead vehicle (seconds) [distance related to own speed]
20. GPS speed (km/h) [same as in RAW GPS]
21. Course (degrees)
22. Difcourse: course variation (degrees)

3.2 Feature Selection

Feature selection refers to the process of selecting a reduced subset of the most relevant features in data. Feature selection generally involves studying feature correlation and feature importance.

Feature Correlation. Correlation-based selection is one of the popular techniques used for feature selection. Formally, correlation is a measure among the measure statistics that describes the association between the different random variables. There exist many methods for calculating the correlation coefficient, each measuring different types of strength of association. A strong association between two variables means that they are strongly correlated and only one of them need to be selected. Below are three of the best known methods, which were considered in this work.

- Pearson Correlation Coefficient
- Spearman's Correlation
- Linear Correlation

We studied the correlation between 18 variables instead of 22. The variables Timestamp (seconds), Latitude coordinate (degrees), Longitude coordinate (degrees), and Altitude (meters) are by default selected as they will be needed later to make projection on the map.

The results of correlation using Pearson Correlation Coefficient are presented in Fig. 2. The other two methods provided the same correlation information. Unfortunately, these methods are not enough as they perform well only for linear

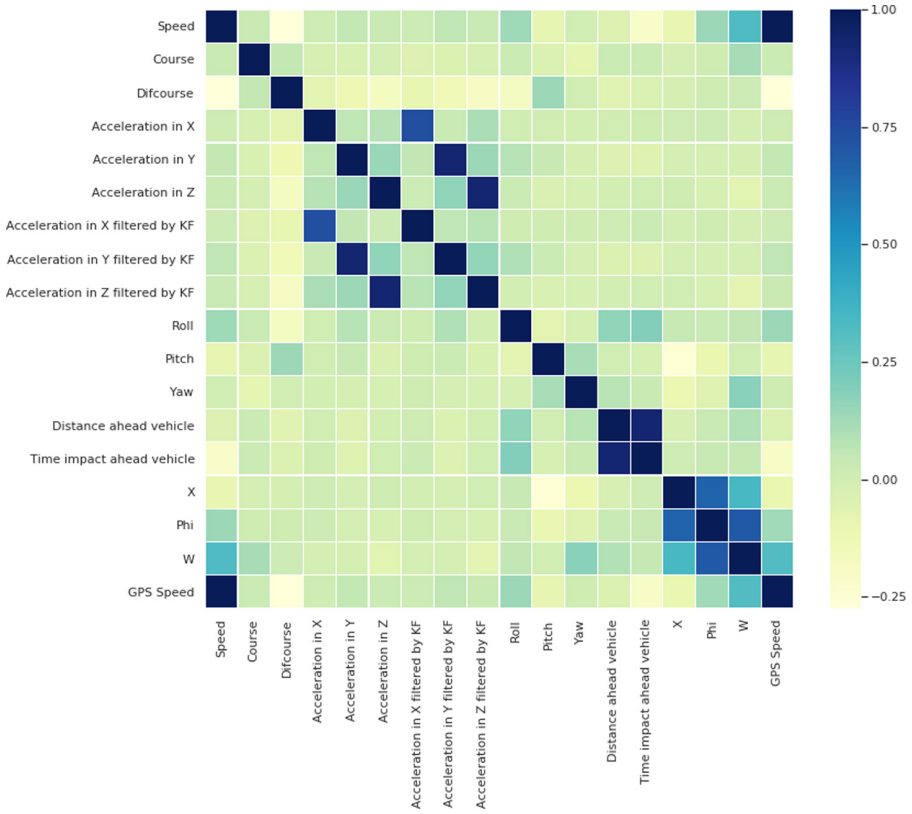


Fig. 2. Pearson correlation

relationships while many other forms of correlation may exist, as illustrated in Fig. 3.

Therefore, we opted for another method to study the correlation. It consists of presenting graphically each feature of data in function of an other one; in our case there are 17^2 combinations resulting in at most 17^2 graphs.

The advantage of the used method is that, in addition to linear correlation, can also detect non-linear relations, i.e. features that have a relation of unknown geometric form, because the graph take a harmonic form. The method results in 6 highly correlated features, which are presented in Table 1.

Feature Importance. According to Table 1, we should either delete the features in column 1 or those in column 2 as the features in both columns are highly correlated. In order to decide which features to keep, we need to study the importance of each feature. Features that are less important will be removed. One common approach is to describe the importance of features relative to a

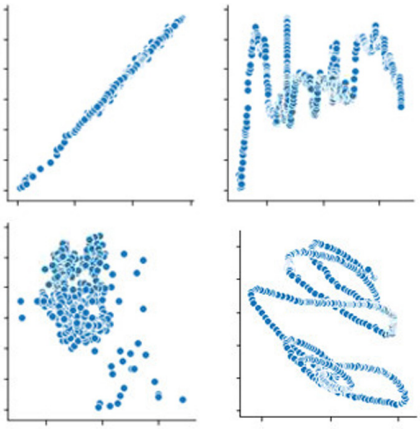


Fig. 3. The different forms of correlation

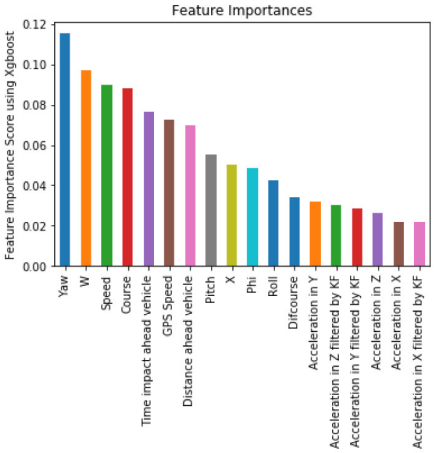
Table 1. List of correlated features. Columns 1 correlated with columns 2

Columns 1	Columns 2
GPS speed	Speed
Difcourse	Course
Time impact ahead vehicle	Distance ahead vehicle
Acceleration in X	X filtered by KF
Acceleration in Y	Y filtered by KF
Acceleration in Z	Z filtered by KF

model. This can be performed in two ways, using an *Impurity-Based approach* or *Permutation Importance*.

Impurity-Based Feature Importance. Some popular tree-based models provide importance scores computed based on the reduction in the criterion used to select split points. In this work, we used the *XGboost* model to measure importance. The results are presented in Fig. 4a. According to the obtained scores, the features *Yaw, W, Course and Speed* have the higher scores and are thus the most important, while *Acceleration in X filtered by KF, Acceleration in Y filtered by KF, Acceleration in Z filtered by KF, Acceleration in X, Acceleration in Y, Acceleration in Z* have the lowest scores and are thus the least important. Combining these scores with the correlation results in Table 1, we choose to keep the following variables: *Speed, Acceleration in X filtered by KF, Acceleration in Y filtered by KF, X, Phi, Acceleration in Z filtered by KF, W, Course, Time impact ahead vehicle, Roll, Pitch, Yaw.*

Permutation Feature Importance. Permutation importance describes feature importance based on the impact each feature has on the trained model's



(a) Impurity-Based Feature importance

Weight	Feature
0.4538 ± 0.0080	Yaw
0.3023 ± 0.0082	W
0.2151 ± 0.0057	Course
0.0660 ± 0.0061	GPS Speed
0.0632 ± 0.0039	Speed
0.0437 ± 0.0046	X
0.0410 ± 0.0034	Roll
0.0395 ± 0.0052	Pitch
0.0280 ± 0.0043	Time impact ahead vehicle
0.0234 ± 0.0043	Phi
0.0209 ± 0.0026	Distance ahead vehicle
0.0200 ± 0.0023	Difcourse
0.0105 ± 0.0014	Acceleration in Z filtered by KF
0.0092 ± 0.0019	Acceleration in Y filtered by KF
0.0060 ± 0.0034	Acceleration in Y
0.0054 ± 0.0025	Acceleration in Z
0.0048 ± 0.0013	Acceleration in X filtered by KF
0.0046 ± 0.0014	Acceleration in X

(b) Permutation Feature Importance

Fig. 4. Impurity-based vs permutaion feature importance using XGBoost

predictions. This is done by measuring the increase in the model's prediction error after permuting the feature's values. Here also we used the *XGboost* model. The results are presented in Fig. 4b. Comparing the results of the both techniques, presented in Fig. 4, we see that there is a small difference in classifying the importance of some variable, e.g. *Speed* has a higher score than *GPS speed* in the impurity based method but a little less importance in case of permutation importance.

The results of feature importance after removing features is presented in Fig. 5. Considering all the resulted obtained so far we decided to only keep the variables *Speed, Course, Roll, Pitch, Yaw, Distance ahead vehicle, X*. Features that have an importance score less than the mean were deleted. Moreover, the variable W corresponding to the road width was deleted as it can be extracted from *GPS* data.

3.3 Data Segmentation

To efficiently deal with the continuous driving data and in order to keep pertinent information enough to be useful in real applications, an unsupervised and automatic segmentation method is needed. In this work, we propose the use of Kmeans clustering to segment driving data.

As shown in Fig. 6, driving data is organized as trip data. Therefore, segmentation will be performed in such way that Kmeans is applied to each trip separately to generate clusters of data which are iteratively merged to form the final clusters. The Kmeans algorithm is applied over the 7 features selected in the previous section. Each trip will then generate K clusters. One of the important things in Kmeans clustering is to use an optimal number of cluster K. Using

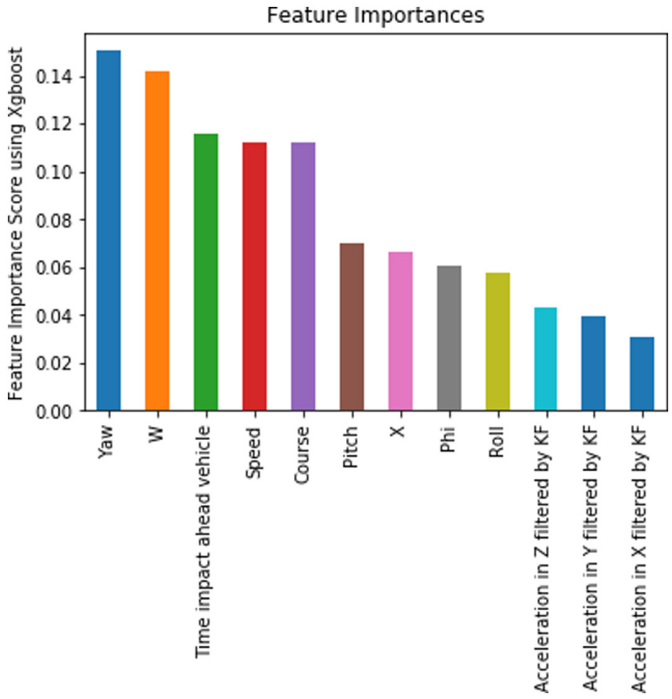


Fig. 5. Feature importance using XGBoost after removing least important features.

a low number could lead to clusters with significant differences while a very large number will generate too many clusters which could lead to overfitting. In order to find the optimal number of clusters, we used the elbow method, a common technique used for this task. Applied over multiple features, Kmeans will generate clusters that can be represented by hyperspheres (n-sphere), i.e. using 7 features will generate 7-dimensional spheres ($\mathcal{E}^7$). Each hypersphere will be defined by its center and its radius. Thus, we can store a minimum data by storing the radius and the center without losing any information.

3.4 Data Harmonization

Segmenting driving trips allows keeping only relevant information. However, segmentation can produce overlapping clusters, which can lead to grouping different behaviors in the same clusters (same hyperspheres). To deal with issue, the harmonization process is introduced. Harmonization is the process by which the hyperspheres are updated, either by merging or splitting. The update is decided by comparing the distance between the hyperspheres' centers with a predefined threshold. This update is applied iteratively to the hyperspheres generated by each trip (output) as illustrated in Fig. 7. The harmonization method can be briefly described as followed.

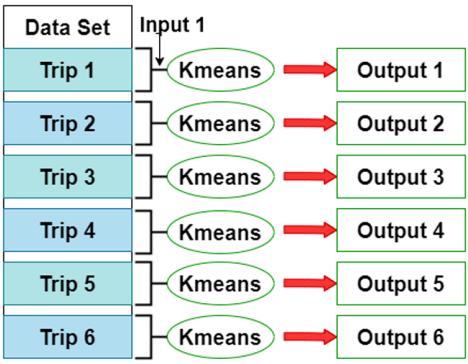


Fig. 6. *KMeans* applied to data trips

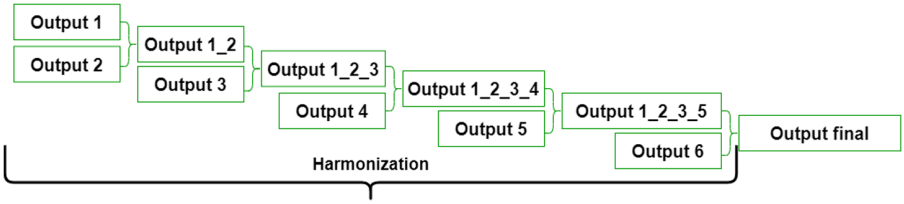


Fig. 7. Data harmonization

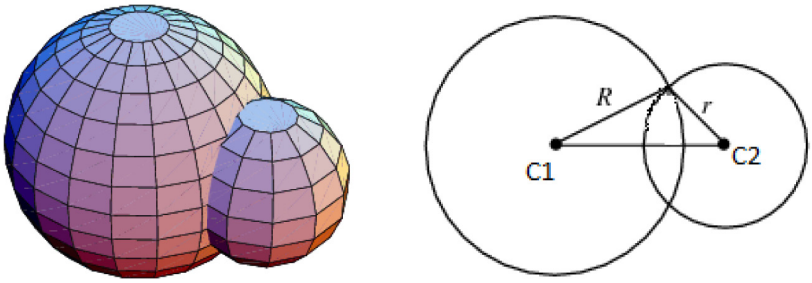
Considering two spheres with centers C1 and C2 and radius R1 and R2, respectively. Figure 8a shows an example of two intersecting spheres. Determining whether an intersection exists between the two spheres is similar to the case of circles, as shown in Fig. 8b:

- if $distance(C1, C2) > R1+R2$ then there is an intersection (or a union exists), and an update may be considered
- Others, no update is needed.

To decide whether to update the hyperspheres requires defining a certain threshold D. The update is considered $if\ [R2 - (distance(C1, C2) - R1)] > D$ otherwise the hyperspheres are kept. To illustrate the two cases of intersection and union, we show, for sake of simplicity, figures of circles instead of hyperspheres (Fig. 9).

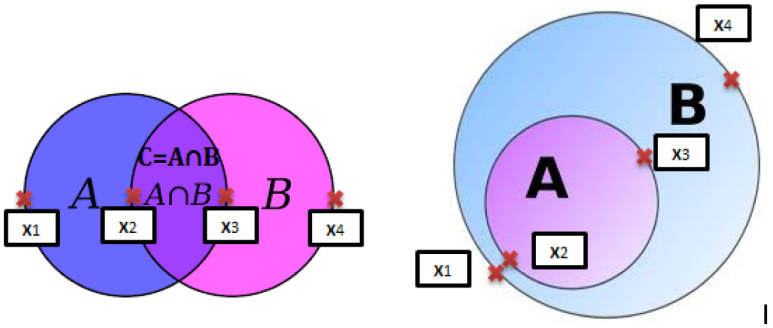
Considering the case of intersection illustrated in Fig. 9, where x1, x2, x3 and x4 are min(A), min(B), max(B) and max(A), respectively. The update is ruled by the following:

$if\ Distance(x1, x2) \geq D\ then\ create\ an\ hypersphere$
$if\ Distance(x2, x3) \geq D\ then\ create\ an\ hypersphere$
$if\ Distance(x3, x4) \geq D\ then\ create\ an\ hypersphere$



(a) Two intersecting Hyperspheres (b) Two intersecting Circles

Fig. 8. An example of intersecting spheres and circles



(a) Circle Intersection (b) Circle Union

Fig. 9. Illustration of union and intersection in case of circles

The update may thus result in creating 3 or 2 new hyperspheres or just a single hypersphere merging the two original ones. The case of union is dealt with in the same way as intersection.

4 Experiments and Results

To validate the proposed segmentation approach, we use the leave one out cross validation method. UAH dataset [9] contains an average of 7 trips per driver. Thus, at each round of the cross validation, one trip is chosen as a test trip and the others are used as training trips. Then, our segmentation method is applied to the training trips and the test trip to get two separate sets of segments, namely, the training segments *train_hyperesphere* and the test segments *test_hyperesphere*. The results are evaluated by comparing the segments in *train_hyperesphere* and *test_hyperesphere*. The error at each round is measured as the number of the test segments not included in any training segments. We define two metrics of errors.

– Metric 1 (E_1): measured as the number of $Test\_hyperspheres$ that can make a modification in the $train\_hyperesphere$, with a given threshold D, divided by the total number of $test\_hyperspheres$. The test segments which can modify the segments from the train are defined as False Positives (FP).

$$E_1 = \frac{FP}{|test\_hyperspheres|} \tag{1}$$

– Metric 2 (E_2): measured s the number of the $hyperspheres$ in the train that can be modified by the $test\_hyperspheres$, divided by the total number of $train\_hyperesphere$.

$$E_2 = \frac{modified\_train\_hyperspheres}{|train\_hyperspheres|} \tag{2}$$

The values of E_1 and E_2 computed using cross validation are 6% and 1%, respectively. This difference in the obtained values is logical since the second error metric depends on the size of the train set, which is large compared to the test set. Moreover, E_2 computes the number of trains that can be modified, however, some $hyperspheres$ can be repeated several times, i.e. they can be modified by several $test_hypersphere$. This repetition is not taken into account, i.e. the number will not increase.

5 Conclusion

In this paper, we deal with the problem of driver profiling. The aim is to transform the numerical sensor data into a high level abstracted form that will represent the driver behavior and make easier the extraction of his driving patterns. Driving behavior is represented by a directed weighted graph, with nodes corresponding to driving states and edges to the transitions between them. The driving states are considered as an abstraction of continuous numerical values. The paper focused on data segmentation. An unsupervised method (using kmeans) to segment multiple driving features is presented. Since we are dealing with multiple variable, the segments had a form of $hypersphere$. The resulting segments are to be used as driving states to build the driver profile. Since driving is generally recorded as trip data, rules to continuously update the segments were proposed. The proposed approach was applied on the UAH-Dataset and evaluated using cross-validation. In order to evaluate the data segments, two error metrics were defined. Our future work focuses on segmentation of GPS data to create road segments on which driver data will be projected.

References

1. Babulal, G.M., Traub, C.M., Webb, M., et al.: Creating a driving profile for older adults using GPS devices and naturalistic driving methodology. F1000Research **5**, 2376 (2016). https://doi.org/10.12688/f1000research.9608.2

2. Divya, G., Sabitha, A., Sudha, D.S., Spandana, K., Swapna, N., Hepsiba, J.: Advanced vehicle security system with theft control and accident notification using GSM and GPS module. Int. J. Innov. Res. Electr. Electron. Instrum. Control Eng. **4**(3), 64–68 (2016)
3. Andreone, L., Brignolo, R., Damiani, S., Sommariva, F., Vivo, G., Marco, S.: Safespot final report. Technical report D8.1.1 (2010)
4. SAFESPOT Integrated Project (2012). http://www.safespot-eu.org/
5. Zott, C., Yuen, S.Y., Brown, C.L., Bertels, C., Papp, Z., Netten, B.: Safespot local dynamic maps: context-dependent view generation of a platform's state and environment. In: 15th Intelligent Transport Systems World Congress, November 2008
6. Shimada, H., Yamaguchi, A., Takada, H., Sato, K.: Implementation and evaluation of local dynamic map in safety driving systems. J. Transp. Technol. **5**, 102–112 (2015)
7. ETSI TR 102 863 V1.1.1: Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Local Dynamic Map (LDM); Rationale for and guidance on standardization (2011)
8. Jain, A.K.: Data clustering: 50 years beyond K-means. Pattern Recogn. Lett. **31**, 651–666 (2010)
9. Romera, E., Bergasa, L.M., Arroyo, R.: Need data for driver behaviour analysis? presenting the public UAH-DriveSet (Brazil). In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)
10. Bouhoute, A., Oucheikh, R., Boubouh, K., Berrada, I.: Advanced driving behavior analytics for an improved safety assessment and driver fingerprinting. IEEE Trans. Intell. Transp. Syst. **20**(6), 2171–2184 (2019). https://doi.org/10.1109/TITS.2018.2864637
11. Car-2-car.org. 2020. About C-ITS. https://www.car-2-car.org/about-c-its/. Accessed 14 Oct 2020
12. Bergasa, L.M., Almería, D., Almazán, J., Yebes , J.J., Arroyo, R.: DriveSafe: an app for alerting inattentive drivers and scoring driving behaviors. In: IEEE Intelligent Vehicles Symposium (IV), pp. 240–245, Dearborn, Michigan, USA, June 2014