2022 3rd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE) | 978-1-6654-5160-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICBAIE56435.2022.9985844

Driving Behavior Identification of Operating Vehicles Based on Real-Time Trajectory Data

Xingchuan Wang *
National Engineering Research Center of Transportation Safety and Emergency Informatics,
China Transport Telecommunications & Information Center
Beijing, China
asteroids@126.com

Lei Deng
School of Traffic & Transportation,
Beijing Jiaotong University;
China Transport Telecommunications & Information Center
Beijing, China
627964668@qq.com

Abstract—**Trajectory data of operating vehicles contains abundant spatiotemporal information about driving behaviors. With the development of big data processing and mode identification techniques, these methods can be used to mine the patterns of driving behaviors from the trajectory data, and compensate the disadvantages of visual image analysis results. In this research, the trajectory data is partitioned by reactive greedy randomized adaptive search procedure into small segments, and driving behavior on the segments is identified by gated recurrent unit neural network model. The proposed method is verified by real-time trajectory data from Chongqing and Shanghai with over 370000 trajectory points, and the result shows that the accuracy of identifying operating vehicles' driving behavior has reached 91.57%, which indicates that the proposed method improves performances in dealing with large scale trajectory data. This could be used to enhance the supervision on operating vehicles' safety for transportation companies as well as transportation management departments, and improve operating vehicles transportation management strategies.**

Keywords-trajectory data, driving behavior identification, trajectory segmentation, gated recurrent unit

I. INTRODUCTION

With the development of BeiDou navigation system and the promotion usage on domestic operating vehicles, a large amount of trajectory data has been generated. Although, the scale of trajectory data is far beyond the normal datasets, and sometimes it cannot be applied directly, with the development of big data technology, it is more convenient to dig deeper into the trajectory data and it has already become a widely concerned issue. In the meantime, with the widely use of Advanced Driver Assistance System (ADAS) and Driver Monitor System (DMS) on operating vehicles, it is possible to record and analyze the drivers' behaviors in the whole process.

However, using ADAS and DMS to identify operating vehicles' driving behaviors has a serious problem, that is ADAS and DMS are depend on visual image analysis to distinguish drivers' behavior, but the image and the camera is easily affected by the device signal noise, camera occlusion, camera deviation and image jitter caused by vehicle vibration, insufficient light, bad weather, operation error and other factors, which will decline the accuracy and credibility of identifying the operating vehicles' driving behavior. Under this condition, the trajectory data could be used to restore operating vehicles' driving behaviors as the data is constant and is able to compensate the deficiency of visual image analysis by ADAS and DMS.

Trajectory segmentation is one of the most important measures to process trajectory data, and it is the process of partitioning a given trajectory into a small number of homogeneous segments with respect to some criteria [1]. Trajectory segmentation not only reduces the computational complexity, but also provides more abundant information. The core of trajectory segmentation is to understand the features of spatiotemporal movement. The main methods of trajectory segmentation include three basic strategies: based on time threshold, based on geometric topology and based on trajectory semantics [2]. Zheng et al. [3][4] used tourists' experience and location interest as indicators to recommend popular attractions, and used historical data to obtain stopping points, and segmented the original trajectory data, which greatly improved the data mining efficiency. Leiva et al. [5] modified the famous K-means method into trajectory segmentation algorithm, and used the time series data to cluster sequential data, but the disadvantage of their method is that the number of trajectory segments should be determined in advance, which, in most cases, cannot be known in advance, so this method has certain limitations. The frequency weight is integrated to optimize the model of interest point mining, in Tiwari's research [6], and the idea of trajectory segmentation is combined to mine the region of interest point.

Based on trajectory data, the analyzing and mining of multidimensional driving behavior characteristics and identifying individual drivers' driving styles can effectively assist in understanding driving behavior and the decision making logic. Scholars [7-16] studying driving behavior and related fields at home and abroad have explored a variety of methods for driving behavior recognition, and with the development of deep learning technology, it has become the most widely used method on mode identification nowadays. Comparing with traditional methods, deep learning algorithms could automatically learn and summarize the features from massive trajectory data. Shahverdy et al [17] proposed a method using convolution neural network (CNN) model to classify the image to five types of driving styles, including normal, aggressive, distracted, drowsy, and drunk driving, and the data was consisted of acceleration, gravity, throttle, speed, and RPM, which contains different roads and different driver behaviors and converted to images using recurrence plot technique to feed the CNN training model. However, their method takes a long time in computing, which may be unsuitable in real time driving behavior forecasting. Alahi et al. [18] used long-short term memory (LSTM) network to predict the short-term trajectory of pedestrians. Zheng et al. [19] used a variety of depth models to analyze the behavior patterns of taxis, which can automatically divide the tracks into different types. Bessa et al. [20] proposed an abnormal trajectory detection tool based on multi-layer convolution network, which can effectively detect buses on abnormal routes and track segments with abnormal time and space. Recurrent neural network (RNN) has many variant structures, such as the gated recurrent unit (GRU), which is widely popular in practice. GRU [21] could maintain long-term dependence by adding internal gating mechanism, and the recurrent structure only depends on all past states. Thus, the current state may also depend on future information.

The remaining part of this paper is organized as follows. Section 2 demonstrates the trajectory segmentation algorithm. Section 3 presents the operating vehicles' driving behavior identification method. Section 4 gives an empirical study based on real-time trajectory point dataset. Section 5 presents the main conclusions of this research.

II. Trajectory Segmentation Algorithm

A. Trajectory Data

The trajectory data *Tra* consists of a series of trajectory points, which can be expressed as a set $Tra=\{P_1, P_2, \ldots, P_n\}$, and the trajectory points $P_i=(X_i, Y_i, T_i, W_i)$, $0\leq i\leq n$, $t_0<t_1< \cdots < t_n$, X_i and Y_i in the trajectory point represent the coordinates of the trajectory point, T_i represents the time recorded by the trajectory point, and W_i represents the feature information set of the trajectory point, $W_i= \{f_1, f_2,..., f_n\}$, feature f_i is also called a point feature of the trajectory point, which can be directly obtained from the information of the trajectory point, and can be instantaneous velocity (calculated according to the position information and time information of two adjacent trajectory points), or the angle change information of two continuous points, etc.

Trajectory segment S is a sub trajectory of trajectory *Tra*, $S=\{\overline{W}, P_u,\ldots,P_v\}$, $\overline{W}$ Represents the average value of each feature of the trajectory of this segment, which is taken as the segment feature of trajectory segment s, i.e$\overline{\ }= \{\overline{f_1}, \overline{f_2},\ldots, \overline{f_n}\}$, $\overline{f_1}$ Represents the average value of the first feature in the segment, and {pu,..., pv} represents a segment of the trajectory *Tra* from P_u to P_v, $0 \leq u \leq v \leq n$. The relationship between trajectory and trajectory segment is shown in Figure 1.

The difference between segment feature and point feature is that segment feature can better represent a certain trend within a segment, while point feature is more about a certain point. On the other hand, the point feature will not change because it is directly obtained from the trajectory points, while the segment feature is obtained by calculating all the trajectory points in the segment, which will change dynamically due to the increase or decrease of the trajectory points in the segment.

B. Loss Function in Trajectory Segmentation Algorithm

In the trajectory segmentation, two rules need to be followed. The first is how to measure the homogeneity within the trajectory segment so that the characteristics of the trajectory points within the trajectory segment are as similar as possible. Continuous and similar trajectory points are generally considered to have a greater possibility of belonging to the same trajectory segment. Secondly, it is necessary to consider the feature distance between the trajectory segments, because we want to ensure that the behavior between the trajectory segments is as different as possible on the premise of ensuring the homogeneity within the trajectory segments as much as possible.

The extreme of high homogeneity within the trajectory segment is to regard the trajectory point itself as a separate trajectory segment, but there are too many similar motion behaviors between these periods. If only one segment is divided in the trajectory, the homogeneity within the segment is the worst, so it is easy to conclude that the first point and the second point are relative in a sense.

The expression of feature distance is defined in the form of Euclidean distance, as shown in Equation (1), where P_{1i} represents the *i*th feature of P_1 point. The variance within the trajectory segment is used to represent the homogeneity within the trajectory segment, as shown in Equation (2), and the variance can reflect the fluctuation degree within the trajectory segment.

$$dst(P_1,P_2)=\sqrt{\sum_{i=1}^{n}(P_{1i}-P_{2i})^2} \tag{1}$$

$$H[\overline{W},\{P_u,\ldots,P_v\}]=\frac{1}{v-u+1}\sum_{i=u}^{v}dst(\overline{W},P_i)^2 \tag{2}$$

As for features of the points, in order to make them have the same effect on the loss function when using the features of trajectory points, all features are normalized by Equation (3) so that the value of each feature falls within the range of (0,1).

$$nor(x)=\frac{x - x_{min}}{x_{max} - x_{min}} \tag{3}$$

The loss function is represented by Equation (4), where the θ indicates the set of the average value of the trajectory $\theta(W_1, W_2, \ldots, W_T)$, φ represents the set of trajectory segments S, and $\varphi = (S_1, S_2, \ldots, S_T)$. Equation (5) represents the distance between adjacent trajectory segments. The similarity of motion behavior between each trajectory segment is expressed by the characteristic distance of $\overline{W}$. The parameter max means the maximum feature distance between adjacent $\overline{W}$, and since each feature is normalized, the value range of each feature is in the range of (0,1). Therefore, for the feature set W with k features, $max=\sqrt{k}$. Equation (6) measures the homogeneity within the trajectory segment. By separately summing up the feature distances with all the trajectory points in the segment to obtain the degree of homogeneity within each trajectory segment, and then add the results of each segment to obtain the degree of homogeneity of the whole trajectory data.

$$cos(\theta,\varphi)=F(\theta)+G(\varphi) \tag{4}$$

$$F(\theta)=ln[1+\sum_{i=1}^{T-1}max-dst(\overline{W}_i,\overline{W}_{i+1})] \tag{5}$$

$$G(\varphi) = ln\left(1 + \sum_{i=1}^{T-1} H[\overline{W}_i, \{P_u, ..., P_v\}_i]\right) \tag{6}$$

C. Reactive Greedy Randomized Adaptive Search Procedure

Greedy randomized adaptive search algorithm (GRASP) is a heuristic algorithm for solving combinatorial optimization problems. The algorithm is divided into two parts, greedy construction part and local search part. Because of its good global search ability and convergence stability, it has been used to solve complex combinatorial optimization problems in many fields. Amilcar et al. [23] built a reactive GRASP model and used semi-supervised dataset to separate trajectory segments as well as forecast trajectory behavior. The reactive greedy randomized adaptive search procedure (RGRASP) algorithm showed good effect in trajectory segmentation.

In the first stage of the algorithm, greed and randomness are combined to construct a feasible solution. In the second stage, the neighborhood of the feasible solution is searched to try to improve the feasible solution.

The step of RGRASP calculates as follows. First, the values of *alist* and *minTimelist* are initialized. The *alist* set stores the α values that may be used by the algorithm. *α* Indicates the greedy coefficient of the algorithm, where greedy search will only select the solution of top *α*% cost from restricted candidate list (RCL) each time. The range of *α* value is from 0.1 to 0.6, and if *α* value is too small, only the solution with the minimum cost will be selected each time, so that the diversity of generated trajectory segments will be affected, and it is more likely to fall into local optimization. While, if *α* value is greater than 0.6, a large number of suboptimal solutions will be selected, which will affect the convergence speed of the algorithm. *MinTime* is defined to limit the minimum length of trajectory segments. Trajectory segments shorter than *minTime* are not allowed to be created.

In the construction phase, a feasible solution is constructed each time in an iterative way. All the constructed solutions are added to the candidate list and their results are sorted. The results are extracted from the candidate list as the input of local search through greedy function control. The greediness of the construction lies in that each iteration will add the result changes caused by small changes to the candidate list for subsequent selection. The randomness lies in that the greedy function determines to randomly select a good solution from the solutions with higher ranking in the candidate list as the input of local search. This solution is not necessarily the highest ranked solution in the candidate list.

The local search phase will take the solution provided by greedy construction phase as the input, and a better result will be searched based on this solution. Because the solution extracted from the candidate list is not necessarily the most optimal solution in the candidate list in the process of greedy random construction process, so the task of local search starts from the initial solution and searches for better results near the current solution in an iterative manner to replace the current results. If no better results are found near the current result, the search is stopped and the search results are returned. After the local search phase, it is compared with the current global optimal solution to determine whether to update the current global optimal solution.

III. DRIVING BEHAVIOR IDENCIFICATION MODEL

A. Recurrent Neural Network

RNN is an important part of deep learning, and it can be regarded as a kind of neural network with short term memory. In convolutional neural network, the neurons can only process one input separately, and the previous input has nothing to do with the latter input. In RNN, the neurons not only get information from other neurons, but also receive information from themselves, therefore, the loop network structure is established (as shown in Figure 1), which makes the network possess stronger memory ability. The RNN will add the previous time's state as the input to obtain the output of the next time, so that it can solve the temporal correlation problems. Because of the stronger memory ability, RNN model is widely used in sequence data and sematic data analyzing.

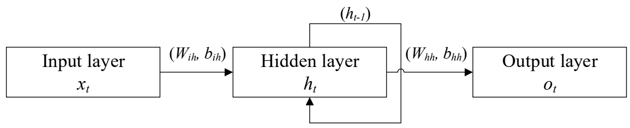

Figure 1. RNN structure

The hidden state h_i of time t is calculated by Equation (7), where x_t is the input in time t, h_{t-1} is the hidden state in time $t-1$, W_{ih} is the weight parameter from input layer to hidden layer, W_{hh} is the weight between hidden layers, b_{ih} represents the bias from input layer to hidden layer, and b_{hh} is the bias between different hidden layers, σ is the activation function.

$$h_t = \sigma(W_{ih}x_t + b_{ih} + W_{hh}h_{t-1} + b_{hh}) \tag{7}$$

In dealing with sequence data, RNN performs better than convolution neural network because of the short term memory ability, however, the simple RNN always suffer from problems like exponential explosion and gradient disappearance with the increase of iterations, and fail to capture the long term correlation, which will lead to convergence difficulties during training. Therefore, some modification of RNN is needed to improve the performance in dealing with large sequence data.

B. Gated Recurrent Unit

In order to solve the problems of RNN model, the long-short term memory(LSTM) RNN is created, which is consist of forget gate, input gate and output gate, and the LSTM RNN could maintain memory state and acquire selectively remember or forget certain information with these three gates. However, when using small-scale data sets, the efficiency of LSTM is not ideal, and the complexity of the model increase significantly, the training time becomes longer. When LSTM is dealing with data that have longer time scale, it will generate a huge amount of computation and time, which will lead to hysteresis in the predict results.

GRU is a simplified modification of LSTM, where the output gate and the forget gate in LSTM are coupled to the update gate, and the input gate is redefined as reset gate. GRU also retains existing information and adds filtered information on the basis of existing information content. The model has storage function. The difference is that GRU eliminates the memory control in the LSTM, which simplifies the computation of LSTM. GRU's structure (Figure 2) is the simplified modification of LSTM, reduces the parameters and shortens the training time, and solves the gradient vanishing problem, which makes it more widely used in time series data

$$r_t = \sigma(W_{ir}x_t + b_{ir} + W_{hr}h_{t-1} + b_{hr}) \tag{8}$$

$$z_t = \sigma(W_{iz}x_t + b_{iz} + W_{hz}h_{t-1} + b_{hz}) \tag{9}$$

$$n_t = tanh\left[W_{in}x_t + b_{in} + r_t \times (W_{hn}h_{t-1} + b_{hn})\right] \tag{10}$$

$$h_t = (1 - z_t) \times n_t + z_t \times h_{t-1} \tag{11}$$

The calculation method in GRU is shown in Equation (8) to Equation (11), where r_t and z_t are representing the state of reset gate and update gate at time t separately, r_t is used to control the former memory, if r_t=0, then, n_t will only contain the information of the current input, z_t controls the hidden layer's forgotten information of the previous moment.

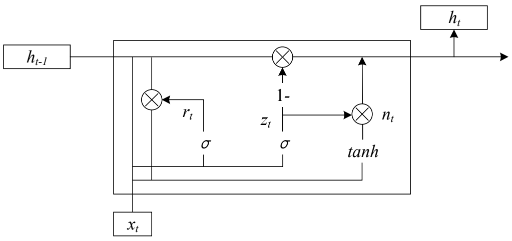

Figure 2. Structure of gated recurrent unit

IV. CASE STUDY

The operating vehicles are all installed with ADAS and DMS equipment, which could classify drivers' behaviors using visual image analysis as long as the camera are working properly. However, because of the influence of longtime use, vehicle vibration, camera occlusion, bad weather, operation error and other factors, the visual image analysis maybe invalid in accurately identifying drivers' driving behaviors. In this research, trajectory data of 35 operating vehicles in Chongqing and Shanghai between October 17th 2020 to November 24th 2020 is used as the dataset to verify the method proposed above. After data filtering and arrangement, the dataset contains over 370,000 trajectory points, including vehicles' real-time latitude, longitude, data acquisition time, instantaneous velocity, rotation angle, altitude, etc. The ADAS and DMS could identify 13 different and detailed driving modes, which could be classified into 4 different driving behaviors, such as normal driving, fatigue driving, distracted driving and aggressive driving, the correspondence between driving modes and driving behaviors is shown in TABLE I, and part of the trajectory spatial distribution is shown in Figure 3.

The trajectory data is preliminary separated by date, and 38 long trajectories are generated for further segmentation. The RGRASP algorithm is adopted and the *minTime* bound is set to 5 minutes to 20 minutes, which means that the shortest segment should contain at least 10 trajectory points (trajectory data is uploaded every 30 seconds as long as the ADAS and DMS device is turned on). In order to verify the effect of RGRASP, GRASP is calculated at the same time. The effect of segmentation is evaluated by indicator called segment *coverage*. *Coverage* is computed for each segment that should be found by a segmentation algorithm. This metric is calculated by extracting the biggest segment found by the segmentation algorithm that covers the segment that must be found, and it could be calculated by Equation (12).

TABLE I. CORRESPONDENCE BETWEEN DRIVING BEHAVIORS AND DRIVING MODES

Driving behavior	Driving mode
Normal driving	Normal
Fatigue driving	Yawning Eyes closed
Distracted driving	Using telephone Smoking Driver not detected Looking at other direction Head down
Aggressive driving	Front collision warning Left side deviation Right side deviation Vehicle distance warning Lane departure

$$coverage(\Psi_V, \theta_T) = \frac{100}{V}\left(\sum_{i=1}^{V} \underset{j\in[1,T]}{arg\ max} \frac{N_{inst}(\Psi_i,\ \theta_j)}{N_i}\right) \tag{12}$$

Where, Ψ_V represents the set of segments that should be found by the segmentation algorithm, θ_T is the set of segments found by the segmentation algorithm, Ni is the total number of points of segment, ψ_i and $N_{inst}(\psi i,\ \theta j)$ represents the number of points of the segment θ_j that intersected the segment ψ_i that should be found by the segmentation algorithm.

The segmentation results are listed in the table below, from which a conclusion could be summed up that RGRASP performs better in segment coverage, but takes a little bit longer time than GRASP in trajectory segmentation. This means that RGRASP algorithm could utilize the trajectory more completely, and could extract more useful information from the trajectory point data.

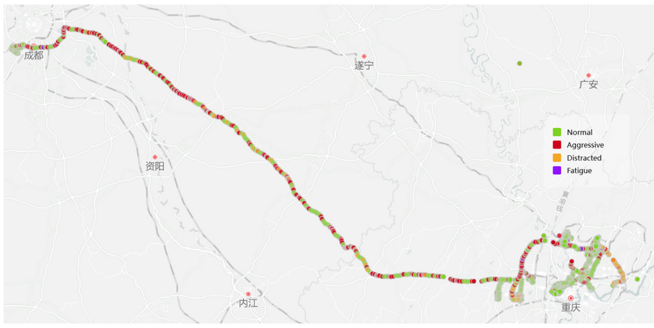

Figure 3. Driving behavior distribution near Chongqing

TABLE II. THE EFFECTS OF TRAJECTORY SEGMENTATION ALGORITHMS

Algorithm	Coverage(%)	Compute time(second)
RGRASP	92.35	2057
GRASP	89.71	1983

Based on the separated trajectory segments, the GRU-RNN algorithm is applied to identify drivers' driving behaviors. Therefore, the segment dataset is divided into training dataset and test dataset, where the former one takes 80% and the test dataset takes the remaining 20%. As the length of each trajectory segment is not equal to each other, the longest length of the trajectory segment is taken as the number of the networks input neurons. When taking shorter segments into calculation, the unused neurons are automatically dropped out to improve computation performances.

The calculation results show that accuracy of GRU-RNN identifying driving behaviors from trajectory segments has reached 91.57%, which indicates that RGRASP algorithm and GRU-RNN is capable of identifying driving behaviors from trajectory segments, and the method proposed in this research is useful in correcting ADAS and DMS devices' visual image analysis results on operating vehicles, and it is an compensation measure for ADAS/DMS device manufactures as well as transportation companies and transportation management departments to enhance the supervision on operating vehicles.

V. Conclusions

In this research, the trajectory data generated from operating vehicles is arranged and used to divide into smaller segments so that the drivers' driving behaviors could be more detailedly analyzed and identified. The conclusions of this research could be summed up as follows:

(1) The RGRASP algorithm is put to use on operating vehicles' trajectory data, the dynamic optimal methods in construction phase and local search phase could improve the performance in dealing with large scale trajectory data, and the trajectory segmentation algorithm shows good effect comparing to similar algorithms.

(2) In order to improve the identification effect of the driving behaviors, the GRU-RNN model is applied with dynamic input drop out to accommodate various length of trajectory segments, and the identification results show that the accuracy has reached 91.57%, which also shows good effect on operating vehicles' driving behavior identification.

To sum up, operation vehicles' trajectory data contains lots of information that could be used to enhance transportation supervision and promote whole transportation safety, and the method proposed in this research have shown good effects on analyzing real-time trajectory data and could be used to improve transportation management strategies.

Acknowledgment

This work was financially supported by the China Transport Telecommunications & Information Center Reserve Program (2020CB03).

References

[1] Yoon, H. and Shahabi, C., 2008. Robust time-referenced segmentation of moving object trajectories. Proceedings of the 2008 8th IEEE international conference on data mining, ICDM '08, December. Washington, DC: IEEE Computer Society, 1121–1126. doi:10.1109/ICDM.2008.133.

[2] Xu JJ, Zheng K, Chi MM, Zhu YY, Yu XH, Zhou XF. Trajectory big data: Data, applications and techniques. Journal on Communications, 2015,36(12):97−105 doi: 10.11959/j.issn.1000-436x.2015318.

[3] Zheng Y, Xie X. Learning travel recommendations from user-generated GPS traces. ACM Trans. on Intelligent Systems & Technology, 2011,2(1):389−396. doi: 10.1145/1889681.1 889683.

[4] Zheng Y, Zhang LZ, Xie X, Ma WY. Mining interesting locations and travel sequences from GPS trajectories. In: Proc. of the Int'l Conf. on World Wide Web. ACM Press, 2009. 791−800, doi: 10.1145/1526709.1526816.

[5] Leiva L A, Vidal E J I S.Warped k-means: An algorithm to cluster sequentially distributed data,2013, 237: 196-210.

[6] Tiwari S, Kaushik S. Popularity estimation of interesting locations from visitor's trajectories using fuzzy inference system. Open Computer Science, 2016,6(1):8−24, doi: 10.1515/comp-2016-0002.

[7] Wang W, Xi J. A rapid pattern-recognition method for driving styles using clustering-based support vector machines. 2016 American Control Conference (ACC). IEEE, 2016: 5270–5275.

[8] Wang W, Xi J, Chong A, et al. Driving Style Classification Using a Semi-Supervised Support Vector Machine. IEEE Transactions on Human-Machine Systems, 2017, 47(5): 650–660.

[9] Han W, Wang W, Li X, et al. Statistical-based approach for driving style recognition using Bayesian probability with kernel density estimation. IET Intelligent Transport Systems, 2019, 13(1): 22–30.

[10] Li G, Li S E, Cheng B, et al. Estimation of driving style in naturalistic highway traffic using maneuver transition probabilities. Transportation Research Part C: Emerging Technologies, 2017, 74: 113–125.

[11] Chen S H, Pan J S J, Lu K. Driving behavior analysis based on vehicle OBD information and adaboost algorithms. Proceedings of the International Multi-Conference of Engineers and Computer Scientists. 2015, 1: 102–106.

[12] Zhang Z.H.. Research on Driving Behavior Based on OBD Data Analysis. Chang'an University, 2017.

[13] Fugiglando U, Massaro E, Santi P, et al. Driving Behavior Analysis through CAN Bus Data in an Uncontrolled Environment. IEEE Transactions on Intelligent Transportation Systems, 2019, 20(2): 737–748.

[14] Zhu B, Yan S, Zhao J, et al. Personalized Lane-Change Assistance System with Driver Behavior Identification. IEEE Transactions on Vehicular Technology, 2018, 67(11): 10293–10306.

[15] Li Z.G.. Driving Behavior Analysis Based on Data Mining Technology. Army Medical University, 2017.

[16] Feng Y, Pickering S, Chappell E, et al. Driving Style Analysis by Classifying Real-World Data with Support Vector Clustering.IEEE International Conference on Intelligent Transportation Engineering (ICITE). IEEE, 2018: 264–268.

[17] Shahverdy, M., Fathy, M., Berangi, R., & Sabokrou, M.. Driver Behavior Detection and Classification Using Deep Convolutional Neural Networks. Expert Systems with Applications, 2020, 113240.

[18] Alahi A, Goel K, Ramanathan V, et al. Social lstm: Human Trajectory Prediction in Crowded Spaces. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016:961-971.

[19] Zheng Y, Chen Y, Li Q, et al. Understanding Transportation Modes Based on GPS Data for Web Applications. ACM Transactions on the Web (TWEB),2010,4(1):1.

[20] BESSA A, DE MESENTIER SILVA F, NOGUEIRA R F, et al. Rio Bus Data: Outlier detection in bus routes of rio de janeiro. 2016: 160106128,.

[21] LIU W, WANG Q, ZHU Y L, et al. GRU: optimization of NPI performance. Journal of Supercomputing, 2020, 76(5): 3542-3554.

[22] Soares Junior A , Moreno B N , Times V C , et al. GRASP-UTS: an algorithm for unsupervised trajectory segmentation. International Journal of Geographical Information Science, 2015, 29(1-2):46-68.