Hindawi
Journal of Advanced Transportation
Volume 2020, Article ID 8859689, 12 pages
https://doi.org/10.1155/2020/8859689

WILEY | Hindawi

Research Article

UB-LSTM: A Trajectory Prediction Method Combined with Vehicle Behavior Recognition

Haipeng Xiao,[1] Chaoqun Wang,[1] Zhixiong Li,[2] Rendong Wang,[3] Cao Bo,[1] Miguel Angel Sotelo,[4] and Youchun Xu[3]

[1]*Army Military Transportation University, Tianjin 300181, China*
[2]*School of Mechanical, Materials, Mechatronic and Biomedical Engineering, University of Wollongong, Wollongong, NSW 2522, Australia*
[3]*Institute of Military Transportation, Army Military Transportation University, Tianjin 300181, China*
[4]*Department of Computer Engineering, University of Alcalá, Alcalá de Henares (Madrid) 28801, Spain*

Correspondence should be addressed to Zhixiong Li; zhixiong_li@uow.edu.au and Rendong Wang; wrd1992@163.com

Received 3 June 2020; Revised 9 July 2020; Accepted 13 July 2020; Published 1 August 2020

Academic Editor: Wen LIU

In order to make an accurate prediction of vehicle trajectory in a dynamic environment, a Unidirectional and Bidirectional LSTM (UB-LSTM) vehicle trajectory prediction model combined with behavior recognition is proposed, and then an acceleration trajectory optimization algorithm is proposed. Firstly, the interactive information with the surrounding vehicles is obtained by calculation, then the vehicle behavior recognition model is established by using LSTM, and the vehicle information is input into the behavior recognition model to identify vehicle behavior. Then, the trajectory prediction model is established based on Unidirectional and Bidirectional LSTM, and the identified vehicle behavior and the input information of the behavior recognition model are input into the trajectory prediction model to predict the horizontal and vertical speed and coordinates of the vehicle in the next 3 seconds. Experiments are carried out with NGSIM data sets, and the experimental results show that the mean square error (MSE) between the predicted trajectory and the actual trajectory obtained by this method is 0.124, which is 97.2% lower than that of the method that does not consider vehicle behavior and directly predicts the trajectory. The test loss is 0.000497, which is 95.68% lower than that without considering vehicle behavior. The predicted trajectory is obviously optimized, closer to the actual trajectory, and the performance is more stable.

1. Introduction

Trajectory prediction is an important research direction in the field of autopilot [1, 2]. The research on the decision-making characteristics of the driver shows that factors such as the relative speed and relative distance between the car and the surrounding moving vehicles will greatly affect the driver's decision [3] and then affect the driving safety. For static vehicles in the driving environment, intelligent vehicles can drive safely along the planned trajectory; for dynamic vehicles, human drivers can use past experience and intuition to predict the behavior of other drivers to avoid potential accidents [4]. Intelligent vehicles need to improve their driving safety by predicting the trajectories of the moving vehicles around them in real time.

The existing methods are basically separate research on vehicle behavior recognition and vehicle trajectory prediction, and there are not many methods to combine the two to make a more accurate trajectory prediction [5–10]. Improving the accuracy of vehicle trajectory prediction is the most urgent problem to be solved. In fact, the accurate identification of vehicle behavior is very important to improve the accuracy of vehicle trajectory prediction, so this paper will take vehicle behavior recognition into consideration.

There are many existing trajectory prediction methods, such as the Markov model [11]. Its advantage is that it can

calculate the probability of maintenance capability and multiple degraded state systems, but it is not suitable for long-term prediction and is easily affected by the external environment. The advantage of Bayesian model [12] is that it is not sensitive to missing data but needs to know a priori probability, and a priori probability often depends on assumptions, and there can be many hypothetical models; therefore, at some point, the prediction effect is poor due to the hypothetical a priori model. The advantage of Kalman filter [13] is that the prediction in a short time (1 step or 2 steps) can be judged stably and accurately, but the trajectory prediction for a long time (such as more than 3 seconds or more than 5 steps) will seriously affect the prediction accuracy due to the increase of prediction error, and the model is very complex and easy to be affected by external noise.

LSTM has a long-term memory function and can learn to translate languages; control robots; make image analysis, document summaries, speech recognition, image recognition, and handwriting recognition; control chatbots, predict diseases; click rates and stocks; and synthesize music [14–17]. Vehicle trajectory prediction is also a time series problem, and there is a correlation between each track point and the historical track point, so it is very suitable to use LSTM to solve the trajectory prediction problem.

In order to solve the problem of vehicle trajectory prediction, a UB-LSTM vehicle trajectory prediction model combined with vehicle behavior recognition is proposed in this paper. Firstly, a many-to-one vehicle behavior recognition model is established based on LSTM, and then a trajectory prediction model is established based on one unidirectional LSTM and one bidirectional LSTM. The identified vehicle behavior information and the input information of the vehicle behavior recognition model are input into the trajectory prediction model to predict the horizontal and longitudinal speed and coordinates of the vehicle, and in the follow-up process, based on the predicted horizontal and longitudinal velocity, an acceleration trajectory optimization algorithm is proposed, and the predicted trajectory based on the optimization algorithm is more consistent with the actual trajectory and more stable. The contributions of this paper are as follows:

A better trajectory prediction model is proposed, and a small test loss is obtained. In this paper, the influence of the surrounding vehicles on the predicted vehicles (that is, interactive information) is taken into account in the model, and the prediction effect of the model is improved.

This paper considers a method that combines the vehicle behavior recognition model with the trajectory prediction model; that is, the behavior of the vehicle is input into the trajectory prediction model as one of the input information, which can obviously improve the accuracy of trajectory prediction.

Based on the predicted horizontal and vertical velocity, an acceleration trajectory optimization algorithm is proposed, which obviously improves the accuracy and stability of the predicted trajectory.

2. Related Work

At present, in the aspect of vehicle trajectory prediction, the main methods are trajectory prediction based on physical motion model, trajectory prediction based on driving behavior, and trajectory prediction based on intention recognition. Gambs et al. [18] put forward the trajectory prediction method of high-order Markov model, which has high accuracy, but high computational overhead, so it is difficult to meet the real-time requirements of intelligent vehicles. Chandra et al. [10] proposed a new method based on the combination of graph analysis and deep learning to predict vehicle trajectories in urban traffic scenes, and they learned how to predict future trajectories and behaviors from the extracted vehicle trajectories. In order to reduce the error of long-term prediction (3–5 seconds) and improve the accuracy of prediction, spectral clustering regularization method was introduced and experiments were carried out on Argoverse, LYFT, and Apolloscape data sets. Deo and Trivedi [19] proposed an LSTM encoder-decoder model, which uses a convolutional social pool as an improvement to the social pool level to learn the interdependence in vehicle motion stably. In addition, based on the variability of trajectories, the model also outputs the multimodal prediction distribution of future trajectories and uses US-101 and I-80 sections of NGSIM data sets to evaluate the model. Chang et al. [20] proposed the Argoverse data set, which is designed to support self-driving vehicle perception data including 3D tracking and motion prediction. Argoverse data set includes sensor data collected by self-driving teams in Pittsburgh and Miami, as well as 3D tracking notes, extracting 300000 vehicle tracks and rich semantic maps. Using this data set can greatly reduce trajectory prediction errors. Chandra et al. [21] proposed an end-to-end vehicle trajectory prediction algorithm-RobustTP, which uses a tracking algorithm to obtain noise sensor input tracks from RGB cameras (whether stationary or moving) to predict vehicle trajectories in dense traffic and regards noise as a deviation from the real track. Firstly, the online motion model and the case segmentation algorithm based on deep learning are combined to calculate the trajectory. These noise tracks are trained by LSTM-CNN neural network structure, which simulates the interaction between different traffic objects in dense and nonuniform traffic. Chandra et al. [22] used a new LSTM-CNN hybrid network to model the interaction between different traffic objects for trajectory prediction, including buses, cars, scooters, bicycles, or pedestrians. Giuliar et al. [23] proposed a transformer network for pedestrian trajectory prediction and achieved good prediction results. Monti et al. [24] proposed a new recursive generation model to predict the trajectory of obstacles, which takes into account not only the future target of a single obstacle but also the interaction between different obstacles, and in this

model, a graph neural network based on double attention is used to collect the interactive information between different obstacles, which is combined with the possible future trajectory of the obstacles, and a good prediction effect is obtained in the urban scene. Mohamed [25] proposed the social space-time graph convolution neural network (Social-STGCNN), which uses the method of modeling interaction as a graph instead of the aggregation method. The experimental results show that the final displacement error (FDE) is 20% higher than the existing methods, the average displacement error (ADE) is 8.5 times higher, and the reasoning speed is 48 times faster. Li et al. [26] proposed a universal generative neural system (Social-WaGDAT) for trajectory prediction of various obstacles and evaluated the system on three common trajectory prediction data sets, and the experimental results show that the model has a good prediction effect in terms of prediction accuracy, in which the types of obstacles include pedestrians, bicycles, and vehicles. Hao et al. [27] proposed an end-to-end generative model called attention map encoder network (AMENet), which can accurately and truly realize multipath trajectory prediction and achieve good prediction results.

In order to make an accurate prediction of moving vehicle trajectory, a UB-LSTM vehicle trajectory prediction model combined with vehicle behavior recognition is proposed to obtain a smaller prediction error. Finally, based on the predicted horizontal and vertical velocity, an acceleration trajectory optimization algorithm is proposed.

3. Methods

Long short-term memory (LSTM) network [28] is a recurrent neural network (RNN) structure proposed by Hochreitrer and Schmidhuber in 1997. LSTM mainly solves the problems of gradient explosion and gradient vanishing of RNN [29]. LSTM mainly adds forgetting gate, input gate, and output gate on the basis of RNN to realize selective forgetting and memory of information, thus realizing the function of long-term memory. LSTM realizes the function of long-term memory through long-term memory. Because there is only simple multiplication and addition on the track of long-term memory, and there is no nonlinear operation, information flows more smoothly at different times, which can effectively restrain the problem of gradient dissipation of long-term memory.

The process of this method is to first establish a vehicle behavior recognition model to realize vehicle behavior recognition and finally input the behavior identified by the vehicle behavior recognition model and vehicle information into the established trajectory prediction model to predict the horizontal and vertical speed and coordinates in the next 3 seconds. The overall flow chart is shown in Figure 1. The concrete realization is to establish a separate vehicle behavior recognition model and a vehicle trajectory prediction model and train them separately, in which the training data of the vehicle trajectory model contains the marked vehicle behavior information. Then, the vehicle behavior recognition model and the trajectory prediction model are tested together, and finally, the acceleration trajectory optimization algorithm is used to generate a more accurate prediction trajectory.

The following will introduce in detail the vehicle behavior recognition model, trajectory prediction model, and acceleration trajectory optimization algorithm in turn.

3.1. Vehicle Behavior Recognition Model. The behavior recognition model proposed in this section is mainly used to identify the five behaviors of each vehicle trajectory, including going straight, turning left and right, and changing lanes. Not only the state of the vehicle can affect vehicle behavior but also the surrounding vehicles, pedestrians, bicycles and so on. For example, avoiding pedestrians and bicycles will obviously affect the vehicle behavior. Therefore, the input features of each trajectory point include the vehicle state (i.e., coordinates, velocity, and acceleration, etc.) and interactive information features. Before predicting the trajectory, the input characteristics of each trajectory point are input into the vehicle behavior recognition model to get the behavior characteristics of the vehicle. Then, the behavior characteristics, vehicle state characteristics, and interactive information characteristics are input into the trajectory prediction model together.

In this paper, a vehicle behavior recognition model is established based on a many-to-one LSTM classifier, as shown in Figure 2, where the seq_length is the number of features of the input data (i.e., the number of LSTM units); batch training is used to load data, and the batch_size is the load size; the embedding is the corresponding vector length of the input LSTM unit, where embedding = 1; the hidden_size is the number of LSTM hidden layer nodes; the output_size is the output category size; the n_layers is the number of hidden layers of LSTM. Finally, a full connection layer FC is used to make the classification, and only the last node $y_{\text{seq\_length}}$ is taken as the classification result. The output layer does not use activation functions. The Adam is used to update weights. The loss function is CrossEntropyLoss Function L, as defined in

$$L = -\sum_{i}^{N} y^{(i)} \log \widehat{y}^{(i)} + \left(1 - y^{(i)}\right) \log\left(1 - \widehat{y}^{(i)}\right), \tag{1}$$

where $y^{(i)}$ is the actual value and $\widehat{y}^{(i)}$ is the predicted value.

3.2. Vehicle Trajectory Prediction Model. The vehicle behavior recognition model is shown in Figure 3, which is mainly used to predict the vehicle trajectory. It can be seen that not only the state of the vehicle but also the surrounding vehicles, pedestrians, and bicycles can affect the trajectory of the vehicle and that the current behavior of the vehicle can also determine the future trajectory. For example, the trajectories of straight lines and turns are obviously different. The behavior characteristics can be marked in advance in the dataset, or they can be identified by the trained vehicle behavior recognition model. In this paper, the trajectory point of the output trajectory includes the characteristics of horizontal and longitudinal velocity and horizontal and longitudinal coordinates, that is, to predict the future

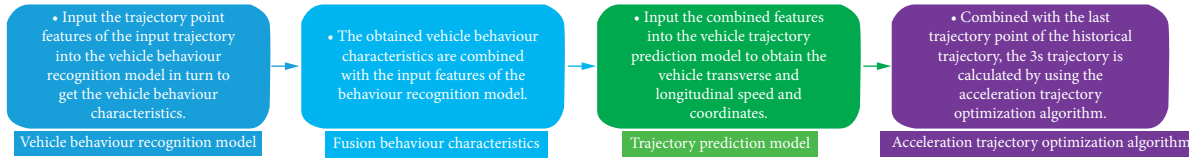

Figure 1: The basic flow of trajectory prediction.

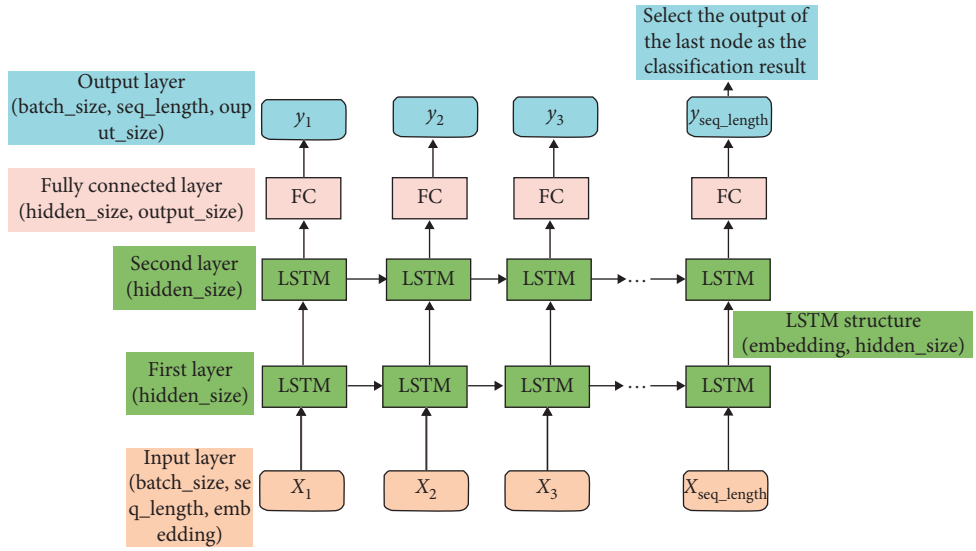

Figure 2: Vehicle behavior recognition model.

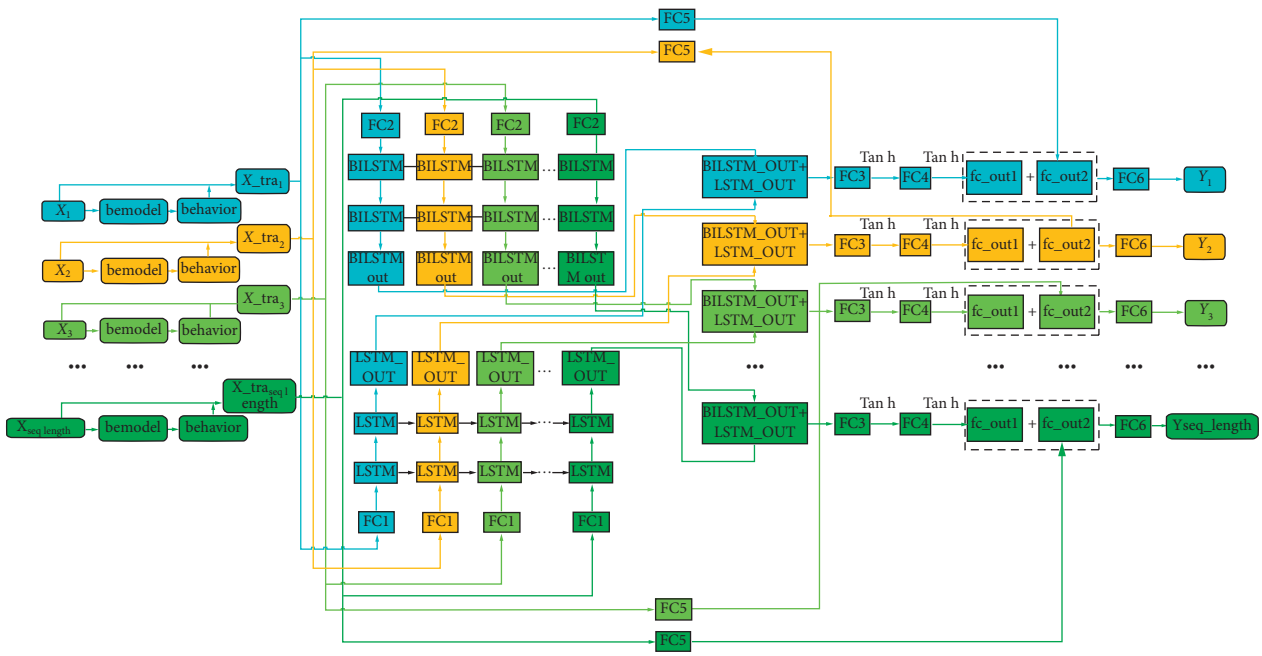

Figure 3: Trajectory prediction model.

horizontal and longitudinal speed and coordinates of the vehicle.

Compared with the traditional Unidirectional or Bidirectional LSTM trajectory prediction model, this paper uses UB-LSTM to establish the trajectory prediction model. Bidirectional LSTM can make use of not only the previous information but also some of the latter information to make the prediction results more accurate. Then, combining the prediction results of the two LSTM can further improve the overall prediction accuracy of the model. The results of the

(1) Input: x
(2) Output: out
(3) (batch_size, seq_length, n_feature-1) ⟵ x.shape;
(4) behavior ⟵ bemodel(x);
(5) x_tra ⟵ torch.cat(x, behavior);
(6) (batch_size, seq_length, n_feature) ⟵ x_tra.shape;
(7) Linear(n_feature, input_size) ⟵ FC1;
(8) LSTM(input_size, hidden_size, n_layers, bidirectional = False, dropout) ⟵ LSTM;
(9) Linear(n_feature, input_size) ⟵ FC2;
(10) LSTM(input_size, hidden_size//2, n_layers, bidirectional = True, dropout) ⟵ BILSTM;
(11) LSTM_OUT, hidden ⟵ LSTM(FC1(x_tra));
(12) BILSTM_OUT, hidden ⟵ BILSTM(FC2(x_tra));
(13) Linear(hidden_size, hidden_size ∗ 2) ⟵ FC3;
(14) Linear(hidden_size ∗ 2, hidden_size//2) ⟵ FC4;
(15) LSTM_FC_OUT ⟵Tanh(FC3(LSTM_OUT + BILSTM_OUT));
(16) fc_out1 ⟵ Tanh(FC4(LSTM_FC_OUT));
(17) Linear(n_feature, hidden_size//2) ⟵ FC5;
(18) fc_out2 ⟵ FC5(x_tra);
(19) Linear(hidden_size//2, output_size) ⟵FC6;
(20) out ⟵FC6(fc_out1 + fc_out2);
(21) end;
(22) **Return** out;

Algorithm 1: Trajectory prediction model.

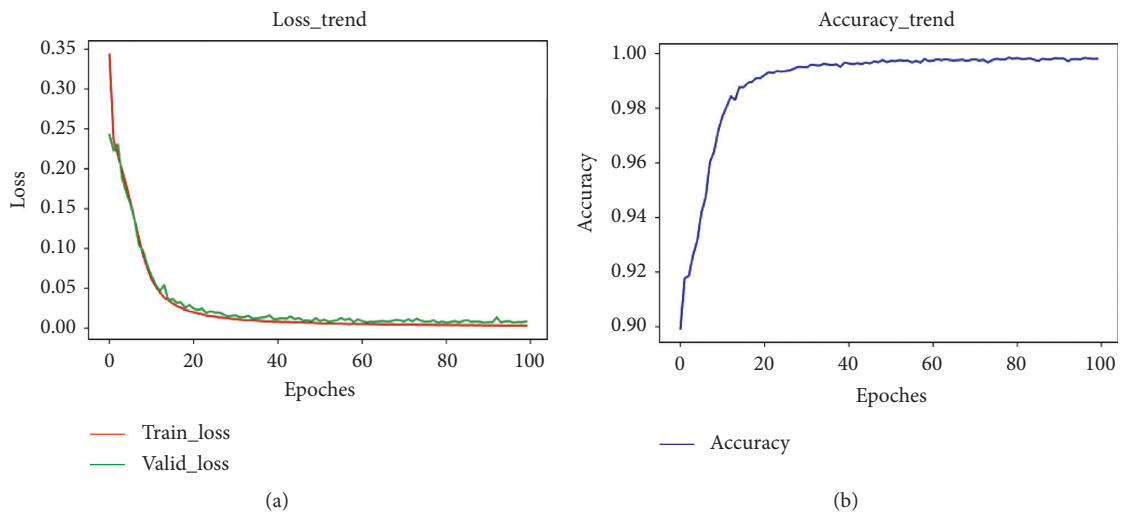

Figure 4: Training process of behavior recognition model. (a) Loss change of behavior recognition model. (b) Accuracy change of behavior recognition model.

Table 1: Test results of the behavior recognition model.

	Left_turn	Left_change	Right_turn	Right_change	Keep	All	Test_loss
Correct_n/ALL	9017/9042	8635/8691	1708/1720	1524/1552	152166/152195	173050/17320	0.0026
Accuracy (%)	99.72	99.36	99.30	98.20	99.98	99.91	

addition of the two LSTM output layers are input into the two full connection layers in turn. Because the value range of the Tanh function is from −1 to 1, the range of the output value is limited, so the input data is directly input into a full connection layer to get an output data and then process the output value by the Tanh activation function. Finally, a full connection layer is used to get the output result. The pseudo-code of the trajectory prediction model is as follows: (Algorithm 1).

The bemodel is the vehicle behavior recognition model mentioned in Part A of Section 3. The seq_length is the

number of input trajectory points. The n_feature is the number of features contained in the input trajectory points. The input_size is the length of the vector input into LSTM cells, the hidden_size is the number of hidden layer nodes of LSTM, and the n_layers is the number of hidden layers. Using dropout method to prevent overfitting, the dropout is in the discarding rate. The behavior is the vehicle behavior identified by the vehicle behavior recognition model, and bidirectional indicates whether the LSTM is a bidirectional LSTM. The Tanh is a hyperbolic tangent activation function. The loss function uses the mean square error loss function (MSELoss), as shown in

$$\text{MSELoss} = \frac{1}{2n}\sum_{i=1}^{n}\left(y_i - \widehat{y}_i\right)^2, \tag{2}$$

where n represents the total number of variables, y_i represents the actual value, and $\widehat{y}_i$ represents the predicted value.

3.3. Acceleration Optimization Trajectory Algorithm. Based on the trajectory prediction model, an acceleration trajectory optimization algorithm is proposed in this paper. This algorithm refers to the acceleration and displacement formula in physics, as shown in

$$s = \frac{\left(v^2 - v_0^2\right)}{2a}, \tag{3}$$

where s is the displacement, v is the final velocity, v_0 is the initial velocity, and a is the acceleration at v_0. The transverse and longitudinal acceleration is calculated according to

$$\begin{cases} a_{x_{t-1}} = \dfrac{\left(v_{x_t} - v_{x_{t-1}}\right)}{\Delta t}, \\ a_{y_{t-1}} = \dfrac{\left(v_{y_t} - v_{y_{t-1}}\right)}{\Delta t}, \end{cases} \tag{4}$$

where Δt represents the interval time between two points, v_x and v_y represent the horizontal and longitudinal speeds of the vehicle predicted by the trajectory prediction model, and t represents the time of the trajectory point. The horizontal and longitudinal displacements are calculated according to

$$\begin{cases} s_{x_{t-1}} = \dfrac{v_{x_t}^2 - v_{x_{t-1}}^2}{2a_{x_{t-1}}}, \\ s_{y_{t-1}} = \dfrac{v_{y_t}^2 - v_{y_{t-1}}^2}{2a_{y_{t-1}}}. \end{cases} \tag{5}$$

The coordinate values are calculated according to

$$\begin{cases} x_t = x_{t-1} + s_{x_{t-1}}, \\ y_t = y_{t-1} + s_{y_{t-1}}, \end{cases} \tag{6}$$

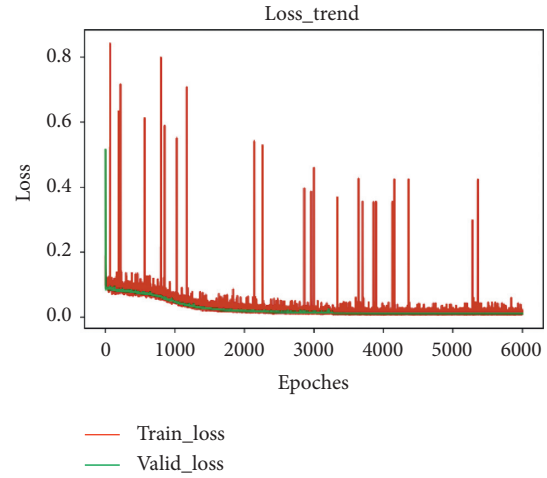

Figure 5: Training process of trajectory prediction model without vehicle behavior.

where x_{t-1} and y_{t-1} represent the coordinates of the previous trajectory point, and x_t and y_t represent the coordinates of the trajectory point. In this paper, the last point of the historical trajectory point is taken as the starting trajectory point and then calculated in turn until the coordinates of all the predicted trajectory points are calculated. In this paper, the time interval of trajectory points is 0.1 s.

4. Experimental Validation

4.1. Experiment Platform. In this paper, the experiment is carried out on the ubuntu16.04 system, the GPU is Tesla V100-PCIE-32GB, and the model is built on Jupyter Notebook based on PyTorch.

4.2. Data Sources and Preprocessing. This paper uses the NGSIM data set for experiments. The NGSIM data set is derived from the Next Generation Simulation (NGSIM) program initiated by the Federal Highway Administration of the United States. The sampling frequency is 10 Hz and records information including vehicle coordinates, speed, acceleration, vehicle type, and lane number [30].

4.2.1. Calculate Angle, VEL_X, VEL_Y, and Behavior. The vehicle heading angle θ is calculated using

$$\theta = \arctan\left(\frac{x^i - x^{i-1}}{y^i - y^{i-1}}\right), \tag{7}$$

where (x^i, y^i) represents the coordinates of the vehicle at i time, and (x^{i-1}, y^{i-1}) indicates the coordinates of the vehicle at $i-1$ time.

According to the change rate of heading angle θ between the two trajectory points of the vehicle, ω marks the vehicle behavior Label, as in formula (8). A total of five behaviors are marked, including going straight, turning left and right, and changing left and right lanes, which are represented by 0, 1, 2, 3, and 4, respectively:

$$\omega = \frac{\theta_i - \theta_{i-1}}{\Delta t}, \tag{8}$$

where Δt represents the interval time between two points.

The transverse and longitudinal velocities Vel_x and Vel_y of the vehicle are calculated according to

$$\begin{cases} v_{x_t} = \dfrac{(x_t - x_{t-1})}{\Delta t}, \\ v_{\text{avg}_{x_t}} = \dfrac{\left(v_{x_t} + v_{x_{t+1}}\right)}{2}, \\ v_{\text{begin}} = 2v_{x_0} - v_{\text{avg}_{x_0}}, \\ v_{\text{end}} = 2v_{x_{\text{end}}} - v_{\text{avg}_{x_{\text{end}}}}, \\ v_{x_0} = v_{\text{begin}}, \\ v_{x_1} = v_{\text{avg}_{x_0}}, \\ v_{x_2} = v_{\text{avg}_{x_1}}, \\ \vdots \\ v_{x_{\text{end}}} = v_{\text{end}}, \end{cases} \tag{9}$$

$$\begin{cases} v_{y_t} = \dfrac{(y_t - y_{t-1})}{\Delta t}, \\ v_{\text{avg}_{y_t}} = \dfrac{\left(v_{y_t} + v_{y_{t+1}}\right)}{2}, \\ v_{\text{begin}} = 2v_{y_0} - v_{\text{avg}_{y_0}}, \\ v_{\text{end}} = 2v_{y_{\text{end}}} - v_{\text{avg}_{y_{\text{end}}}}, \\ v_{y_0} = v_{\text{begin}}, \\ v_{y_1} = v_{\text{avg}_{y_0}}, \\ v_{y_2} = v_{\text{avg}_{y_1}}, \\ \vdots \\ v_{y_{\text{end}}} = v_{\text{end}}, \end{cases} \tag{10}$$

where Δt denotes the interval time between two points, (x, y) represents the coordinate of the trajectory point, t represents the time of the occurrence of the trajectory point, and end represents the last trajectory point of the vehicle.

4.2.2. Acquisition of Interactive Information. The main purpose is to obtain the vehicles that may exist in the left-top, left-bottom, middle-top, middle-bottom, right-top, and right-bottom positions around the vehicle, which are represented as L_Top, L_Bot, C_Top, C_Bot, R_Top, and R_Bot, respectively. Firstly, the lateral coordinate difference dis_x and the longitudinal coordinate difference dis_y between the surrounding vehicle and the predicted trajectory vehicle at a certain time are calculated, and then the position of the surrounding vehicle relative to the predicted trajectory vehicle is judged according to

$$\text{relative\_position} = \begin{cases} \text{L\_Top} & -5 < \text{dis\_}x < -1, 0 < \text{dis\_}y < 5, \\ \text{L\_Bot} & -5 < \text{dis\_}x < -1, -5 < \text{dis\_}y < 0, \\ \text{C\_Top} & -1 < \text{dis\_}x < 1, 0 < \text{dis\_}y < 5, \\ \text{C\_Bot} & -1 < \text{dis\_}x < 1, -5 < \text{dis\_}y < 0, \\ \text{R\_Top} & 1 < \text{dis\_}x < 5, 0 < \text{dis\_}y < 5, \\ \text{R\_Bot} & 1 < \text{dis\_}x < 5, -5 < \text{dis\_}y < 0. \end{cases} \tag{11}$$

Then, we record the id, relative distance, speed, acceleration, heading angle, and behavior of the vehicle that meets the conditions. If there is no vehicle in a certain position, the above information is all set to 0, so the interactive information of the vehicle is obtained.

4.2.3. Generate Trajectory and Behavior Dataset. In the experiment, the Savizkg-Golag smoothing algorithm is firstly used to smooth the coordinates of vehicle trajectory points to eliminate the noise, then remove the vehicles with less than 130 trajectory points or two trajectory points with a distance of more than 5 meters, and at the same time, limit the maximum number of trajectories of each vehicle to 61. By doing so, the number of trajectories of each vehicle will keep consistent. Finally, a total of 17,320 trajectory data are generated, and each trajectory contains 100 trajectory points, including a total of 321 vehicles. The trajectory is standardized using

$$x = \frac{\widehat{x} - \text{mean}}{\text{std}}, \tag{12}$$

where $\widehat{x}$ represents a single value in each column, x represents the changed value, mean represents the average of each column, and std represents the standard deviation.

In the experiment, a total of 26 features are selected as the input trajectory points. At the same time, a total of four features including the vehicle horizontal and longitudinal velocity and coordinates of the trajectory point 3 seconds behind the input trajectory point are selected as the output trajectory point.

The vehicle behavior data is first restored according to equation (13), then the vehicle behavior feature of each input trajectory point is separated as a tag, and the rest of 25 features are used as input features:

$$x = \widehat{x} * \text{std} + \text{mean}. \tag{13}$$

In order to better observe the training process, the vehicle trajectory data is divided into a training set, verification

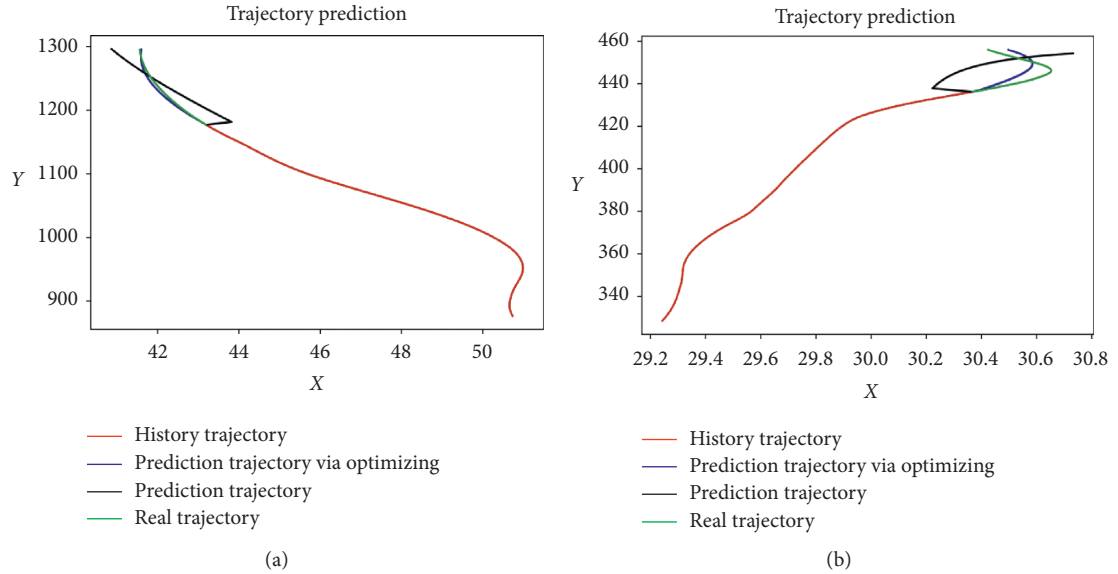

Figure 6: Trajectory prediction results without vehicle behavior. (a) Prediction trajectory 1. (b) Prediction trajectory 2.

set, and test set according to the proportion of 8 : 1 : 1. The training set is used to train the model, the verification set is used to monitor the training process in real time, and the test set is used to evaluate the effect of the model after the training.

4.3. Vehicle Behavior Recognition Model Experiment. Set batch_size = 100, seq_length = 25, embedding = 1, hidden_size = 256, n_layers = 2, output_size = 5, and learning rate = 0.0001. Only the model with minimum validation loss is saved with a total of 100 iterations. The training process is shown in Figure 4.

The training time is 3.943 hours, and the minimum verification loss is 0.0066. The accuracy of vehicle behavior recognition is the ratio of the correct recognition number of each vehicle behavior to the total number of each vehicle behavior. The test results are shown in Table 1. As can be seen from the test results, the recognition accuracy of each behavior of the vehicle is higher than 98%.

4.4. Trajectory Prediction Model Experiment. In order to verify the influence of vehicle behavior, the trajectory prediction accuracy is compared with and without considering the vehicle behavior in this study. In the experiments, the trajectory data is restored using equation (13). In order to reflect the superiority of the acceleration trajectory optimization algorithm proposed in this paper, the trajectory generated by the coordinate of the predicted trajectory point is compared with that generated by the trajectory optimization algorithm.

4.4.1. Trajectory Prediction Model Experiment without Vehicle Behavior. Set batch_size = 128, seq_length = 100, n_feature = 25, input_size = 256, hidden_size = 256, n_layers = 2, output_size = 4, dropout = 0.5, and we used the

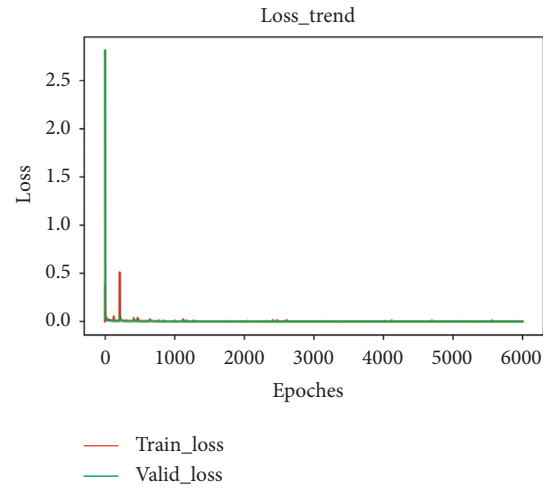

Figure 7: Training process of the trajectory prediction model considering behavior.

ReduceLROnPlateau method to adjust the learning rate. The initial learning_rate is 0.001. When the verification loss does not decrease for 20 iterations, the learning rate is adjusted to 10% of the existing rate. Only the model with minimum validation loss is saved, within a total of 1,000 iterations. The training process is shown in Figure 5, where vehicle behavior is not considered here.

The training time is 3.944 hours, and the minimum verification loss is 0.0114. It can be seen from Figure 5 that the loss decreases slowly during the training. In the testing process, the test loss is 0.0115, the MSE between the predicted trajectory and the actual trajectory is 4.3, and the MSE produced by the optimization algorithm is 2.4. Figure 6 shows the predicted results, where the axis x represents the lateral coordinates of the predicted trajectory point, and the axis Y represents the longitudinal coordinates of the predicted trajectory point. As can be seen from Figure 6, the

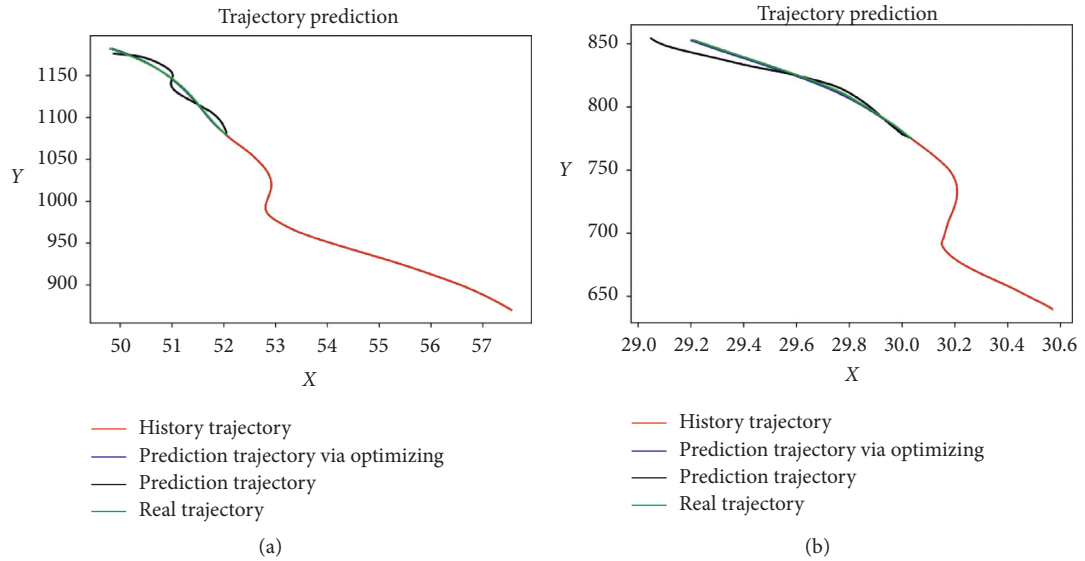

Figure 8: Trajectory prediction results using the two models in the series. (a) Prediction trajectory 1. (b) Prediction trajectory 2.

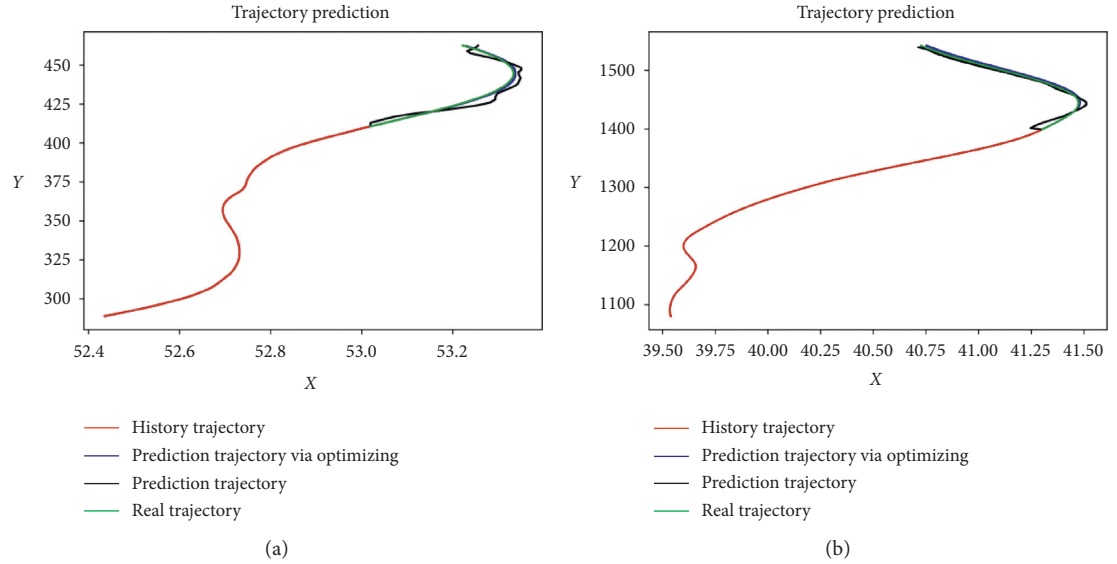

Figure 9: Trajectory prediction results using the two models in parallel. (a) Prediction trajectory 1. (b) Prediction trajectory 2.

trajectory generated by the optimization algorithm is very close to the actual trajectory; but the prediction performance without the optimization algorithm is far away from satisfactory.

4.4.2. Trajectory Prediction Model Experiment Combined with Vehicle Behavior. Set batch_size = 128, seq_length = 100, *n*_feature = 26, input_size = 256, hidden_size = 256, *n*_layers = 2, output_size = 4, dropout = 0.5, and initial Learning_rate = 0.001. Figure 7 depicts the training result.

The training time is 7.12 hours, and the minimum verification loss is 0.00041732. In order to test the fusion effect of the vehicle behavior model and the trajectory prediction model, first of all, the trajectory prediction model is used alone after the behavior recognition model, and then two models are used in parallel.

The vehicle trajectory prediction model predicts the horizontal and longitudinal speed and coordinates of the vehicle and uses the transverse and longitudinal velocity and the last trajectory point of the historical trajectory to realize the acceleration trajectory optimization. First of all, the experiment is carried out without using the vehicle behavior recognition model. The test loss is 0.000486. The MSE between the predicted trajectory and the actual trajectory is 1.361. The MSE generated by the optimization algorithm is 0.122. Figure 8 shows the predicted results, where the axis x represents the lateral coordinates of the predicted trajectory point, and the axis Y represents the longitudinal coordinates of the predicted trajectory point.

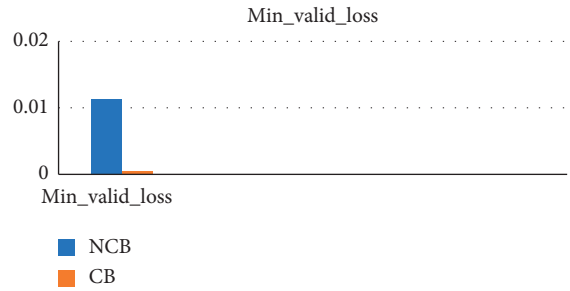

Figure 10: Comparison of minimum verification loss.

Comparing Figures 6 and 8, one can find that the prediction effect is obviously improved; thus, it is very necessary to consider vehicle behavior in the prediction model. The prediction trajectory obtained by the optimization algorithm is more consistent with the actual trajectory, which indicates that the usage of the optimization algorithm makes the prediction performance stable.

When the behavior recognition model and the trajectory prediction model are integrated, the test loss is 0.000497, the recognition accuracy of the behavior recognition model is 99.9%, the MSE without the optimization algorithm is 1.36, and the MSE with the optimization algorithm is 0.124. It can be found that the MSE and test losses after fusion are slightly higher than those of the individual use of the prediction model in Figure 8. Figure 9 shows the predicted results, where the axis x represents the lateral coordinates of the predicted trajectory point, and the axis Y represents the longitudinal coordinates of the predicted trajectory point. It can be seen from Figure 9 that the trajectory prediction effect of the proposed method is good, which is basically consistent with the actual trajectory, and is stable, without large fluctuations, and basically achieves the expected effect. It shows that the trajectory prediction method proposed in this paper has high prediction accuracy and good stability.

4.5. Discussion. As can be seen from Figures 5, 7, and 8, the trajectory generated by the optimization algorithm is much better, more reasonable, and more consistent with the actual trajectory than the trajectory directly predicted in this paper. It shows that the trajectory optimization algorithm proposed in this paper contributes a very good improvement effect.

In order to examine the behavior recognition effect, the following comparisons are further conducted. Herein, the test without considering behavior recognition is abbreviated as NCB; the test with considering behavior recognition is abbreviated as CB; the alone test model considering vehicle behavior is abbreviated as CB_A; the fusion test model considering vehicle behavior is abbreviated as CB_F; the test based on the coordinate method without considering behavior recognition is abbreviated as NCB_CB; the test based on optimization algorithm without considering behavior recognition is abbreviated as NCB_OB; the alone test based on coordinate method considering vehicle behavior recognition is abbreviated as CB_A_CB; the alone test based on optimization algorithm considering vehicle behavior recognition is abbreviated as CB_A_OB; the fusion test based on coordinate method considering vehicle behavior recognition is abbreviated as CB_F_CB; the fusion test based on optimization algorithm considering vehicle behavior recognition is abbreviated as CB_F_OB. The comparison results are demonstrated as follows.

Table 2: Comparison of minimum verification loss.

	NCB	CB
Min_valid_loss	0.0114	0.000417

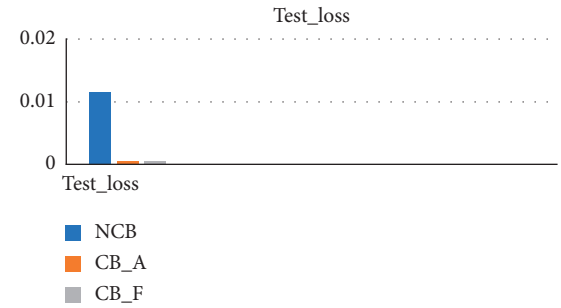

Figure 11: Comparison of test losses.

Table 3: Comparison of test losses.

	NCB	CB_A	CB_F
Test_loss	0.0115	0.000486	0.000497

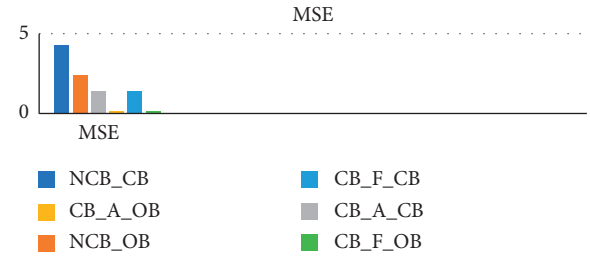

Figure 12: MSE comparison.

4.5.1. Minimum Verification Loss. The minimum verification loss is shown in Figure 10 and Table 2. As can be seen, the minimum verification loss with consideration of behavior recognition is much smaller than that without considering it, which indicates that it is necessary to consider vehicle behavior.

4.5.2. Test Loss. The test losses are shown in Figure 11 and Table 3. It can be seen that the test loss of the model without considering the behavior recognition is the largest and the test loss of the individual test model is the smallest when considering the behavior, which shows the importance of vehicle behavior recognition. Hence, it suggests that in the real world application, it is very important to use a vehicle behavior recognition model to identify vehicle behavior before predicting the trajectory.

Table 4: MSE comparison.

	NCB_CB	CB_A_CB	CB_F_CB	NCB_OB	CB_A_OB	CB_F_OB
MSE	4.3	1.361	1.36	2.4	0.122	0.124

4.5.3. MSE Comparison. The MSE size under each method is shown in Figure 12 and Table 4. It can be seen from the results that the MSE generated by the optimization algorithm is much lower than that of the coordinate method, which means that the optimization algorithm can greatly improve the accuracy of trajectory prediction. It can be also seen that the MSE of the model considering behavior recognition is always lower than that not considering vehicle behavior. In addition, by considering vehicle behavior, the MSE produced by the series configuration of the recognition model and prediction model is very close to that of the parallel configuration of the two models, which shows that combining the two models is feasible.

To sum up, the UB-LSTM based trajectory prediction model, which combines the vehicle behavior recognition and acceleration trajectory optimization algorithm, is able to predict the trajectory accurately and stably.

5. Conclusions and Future Plan

This paper proposes a UB-LSTM trajectory prediction model, which inputs vehicle state information, vehicle behavior information, and interactive information into the trajectory prediction model to predict the vehicle transverse and longitudinal speed and coordinates. A vehicle behavior recognition model is trained to make a prediction combined with the vehicle trajectory prediction model. An acceleration trajectory optimization algorithm is proposed to improve the trajectory prediction accuracy. Experimental results show that the model test loss obtained by the proposed UB-LSTM is 0.000497, and the prediction MSE is 0.124, which is 97.2% lower than that without considering the vehicle behavior information. As a result, the proposed UB-LSTM method is suitable for practical usage.

In the next step, we will consider predicting the trajectory for a longer time and collect more experimental data to improve the model training effect. Although a real car is not available at this moment, the real vehicle experiment will be considered in the near future. The NGSIM dataset used in this experiment records the time stamp, id, coordinates, speed, and acceleration of the vehicles. However, a LIDAR can also provide the time stamp, id, coordinates, velocity, and acceleration of the surrounding obstacles, and the types of obstacles include vehicles, pedestrians, and bicycles. Thus, the interactive information provided by a LIDAR is more comprehensive. Moreover, environmental information can be obtained through high-precision maps, so more input information can be used in the process of real cars. Based on the above characteristics, the real vehicle dataset, similar to NGSIM dataset, can be collected by a LIDAR to carry out real car experiments. The intelligent car uses the C++ language to realize automatic driving, while the training model in this paper is realized by Python language based on PyTorch, so we will consider using LibTorch to migrate the model to the intelligent vehicle. The basic process is as follows: the LIDAR is firstly used to collect the vehicle trajectory dataset, then train the UB-LSTM model, and lastly, transfer the trained model to the intelligent vehicle for real vehicle experiments. However, the difference between the real vehicle experiment and the simulation is that the data processing should be carried out in real time. This will be carried out in the future.

Compared with the physical model method, the disadvantage of the proposed method is that the calculation speed is slower due to a large number of calculations, and it is basically necessary to use GPU to meet the real-time requirements. In the next work, we will try our best to improve the real-time performance of the model.

Data Availability

The code of this article has been uploaded to https://github.com/xhpxiaohaipeng/Vehicle-trajectory-Prediction-combined-with-vehicle-behavior-recognition-.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research was supported by the National Key Research and Development Program of China (2016YFB0100903), NSFC (No. 51979261), Fundamental Research Funds for the Central Universities (No. 201941008), Applied Fundamental Research Project of Qingdao (No. 2019-9-1-14-jch), and Australia Research Council (No. DE190100931).

References

[1] C. Badue, R. Guidolini, R. V. Carneiro et al., "A survey," 2019, http://arxiv.org/abs/1901.04407.

[2] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planningand-decision-makingforautonomousvehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 187–210, 2018.

[3] Y. Liao, S. E. Li, W. Wang, Y. Wang, G. Li, and B. Cheng, "Detection of driver cognitive distraction: a comparison study of stop-controlled intersection and speed-limited Highway," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1628–1637, 2016.

[4] Y. Hu, W. Zhan, and T. Masayoshi, "Probabilistic prediction of vehicle semantic intention and motion," in *Proceedings of the Intelligent Vehicles Symposiun*, April 2018.

[5] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "Traphic:Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8483–8492, Long Beach, CA, USA, June 2019.

[6] RohanChandra, UttaranBhattacharya, X. Li TrishaMittal, A. Bera, and D. Manocha, "Graphrqi:classifying driver behaviors using graph spectrums," 2019, http://arxiv.org/abs/1910.00049.

[7] R. Chandra, U. Bhattacharya, C. Roncal, A. Bera, and D. Manocha, "Robusttp : end-toend trajectory prediction for heterogeneous road-agents in dense traffic with noisy sensor inputs," 2019, http://arxiv.org/abs/1907.08752.

[8] E. Cheung, A. Bera, E. Kubin, K. Gray, and D. Manocha, "Identifying driver behaviors using trajectory features for vehicle navigation," in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3445–3452, IEEE, Madrid, Spain, October 2018.

[9] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," 2018, http://arxiv.org/abs/1805.06771.

[10] R. Chandra, T. Guan, S. Panuganti et al., "Forecasting Trajectory and Behavior of Road-Agents Using Spectral Clustering in Graph-LSTMs," 2019, http://arxiv.org/abs/1912.01118.

[11] J. Firl, H. St¨ubing, S. A. Huss, and C. Stiller, "Predictive maneuver evaluation for enhancement of car-to-x mobility data," in *Proceedings of the Intelligent Vehicles Symposium (IV)*, pp. 558–564, IEEE, Alcala de Henares, Spain, June 2012.

[12] St´ephanie Lef`evre, C. Laugier, and J. Ibañez Guzm´an, "Exploiting map information for driver intention estimation at road intersections," in *Proceedings of the Intelligent Vehicles Symposium (IV)*, pp. 583–588, IEEE, Baden-Baden, Germany, June 2011.

[13] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[14] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proceedings of the INTERSPEECH 2012 13th Annual Conference of the International Speech Communication Association*, Portland, OR, USA, September 2012.

[15] Z. Wu, O. Watts, and S. King, "Merlin: an open source neural network speech synthesis system," in *Proceedings of the 9th ISCA Speech Synthesis Workshop*, Sunnyvale, CA, USA, September 2016.

[16] A. L. Bianne, F. Menasri, R. Al-Hajj, C. Mokbel, C. Kermorvant, and L. Likforman-Sulem, "Dynamic and contextual information in HMM modeling for handwriting recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 10, pp. 2066–2080, 2011.

[17] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of the Eleventh Annual Conference of the International Speech Communication Association*, Makuhari, Chiba, Japan, September 2010.

[18] GAMBS Sebastien, K. Marc-Oliver, MIGUEL Nunez del et al., "Next place prediction using mobility Markov chains," in *Proceedings of the 1st Workshop on Measurement, Privacy, and Mobility*, vol. 3, pp. 1–6, New York, NY, USA, 2012.

[19] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Salt Lake City, UT, USA, June 2018.

[20] M.-F. Chang, J. Lambert, P. Sangkloy et al., "Argoverse: 3D tracking and forecasting with rich maps," in *Proceedings of the CVPR (Computer Vision and Pattern Recognition)*, Long Beach, CA, USA, June 2019.

[21] R. Chandra, U. Bhattacharya, and D. Manocha, "RobustTP: end-to-End trajectory prediction for heterogeneous road-agents in dense traffic with noisy sensor inputs," in *Proceedings of the ACM Computer Science in Cars Symposium on-CSCS '19*, Kaiserslautern, Germany, May 2019.

[22] R. Chandra, U. Bhattacharya, and D. Manocha, "TraPHic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proceedings of the CVPR (Computer Vision and Pattern Recognition)*, Long Beach, CA, USA, June 2019.

[23] F. Giuliari, I. Hasan, M. Cristani, and F. Galasso, "Transformer networks for trajectory forecasting," 2020, http://arxiv.org/abs/2003.08111.

[24] A. Monti, A. Bertugli, S. Calderara, and R. Cucchiara, "DAG-Net: double attentive graph neural network for trajectory forecasting," 2020, http://arxiv.org/abs/2005.12661.

[25] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-STGCNN: a social spatio-temporal graph convolutional neural network for human trajectory prediction," in *Proceedings of the the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 2020.

[26] J. Li, H. Ma, Z. Zhang, and M. Tomizuka, "Social-WaGDAT: interaction-aware trajectory prediction via wasserstein graphdouble-attention network," 2020, http://arxiv.org/abs/2002.06241.

[27] H. Cheng, W. Liao, M. Y. Yang, B. Rosenhahn, and M. Sester, "AMENet: attentive maps encoder network for trajectory prediction," 2020, http://arxiv.org/abs/2006.08264.

[28] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[29] S. Hochreiter, *Untersuchungen zu dynamischen neuronalen Netzen. Diploma*, Technische Universität München, , June 1991.

[30] Federal Highway Administration, "NG-SIM-Next Generation Simulation[EB/OL]," https://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm.