

Received February 28, 2019, accepted March 18, 2019, date of publication March 22, 2019, date of current version April 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2907000

Modeling Vehicle Interactions via Modified LSTM Models for Trajectory Prediction

SHENGZHE DAI^{1,2}, LI LI¹, (Fellow, IEEE), AND ZHIHENG LI^{1,2}, (Member, IEEE)

¹Department of Automation, Tsinghua University, Beijing 100084, China

²Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, China

Corresponding author: Zhiheng Li (zhli@mail.tsinghua.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61790565, in part by the Science and Technology Innovation Committee of Shenzhen under Grant JCYJ20170818092931604, and in part by the Beijing Municipal Commission of Transport Program under Grant ZC179074Z.

ABSTRACT The long short-term memory (LSTM) model is one of the most commonly used vehicle trajectory predicting models. In this paper, we study two problems of the existing LSTM models for long-term trajectory prediction in dense traffic. First, the existing LSTM models cannot simultaneously describe the spatial interactions between different vehicles and the temporal relations between the trajectory time series. Thus, the existing models cannot accurately estimate the influence of the interactions in dense traffic. Second, the basic LSTM models often suffer from vanishing gradient problem and are, thus, hard to train for long time series. These two problems sometimes lead to large prediction errors in vehicle trajectory predicting. In this paper, we proposed a spatio-temporal LSTM-based trajectory prediction model (ST-LSTM) which includes two modifications. We embed spatial interactions into LSTM models to implicitly measure the interactions between neighboring vehicles. We also introduce shortcut connections between the inputs and the outputs of two consecutive LSTM layers to handle gradient vanishment. The proposed new model is evaluated on the I-80 and US-101 datasets. Results show that our new model has a higher trajectory predicting accuracy than one state-of-the-art model [maneuver-LSTM (M-LSTM)].

INDEX TERMS Trajectory prediction, vehicle interactions, shortcut connection, long short-term memory (LSTM).

I. INTRODUCTION

Autonomous vehicles sense static traffic facilities and movements of surrounding vehicles and pedestrians by various sensors all the time to predict their trajectories for future motion planning. The input of a vehicle trajectory prediction model is the historical trajectory of the object vehicle during the last few seconds and the output is the predicted trajectory in the next few seconds.

According to the detailed predicting process, the predicting models used in previous work could be roughly divided into two types: maneuver-based models and end-to-end models.

A. MANEUVER-BASED MODELS

As shown in Fig. 1(a), a maneuver-based trajectory prediction model contains two sequential steps: maneuver recognition step and prediction step [1]–[4]. The maneuver recognition step outputs an intermediate result indicating the motion

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Awais Javed.

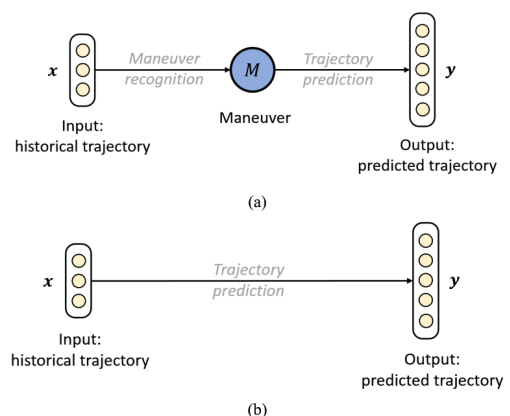


FIGURE 1. Two kinds of trajectory prediction model: (a) Maneuver-based model, (b) End-to-end model.

maneuver of the vehicle of interest whose types are pre-defined. For example, in the scenario of a straight road, the motion of vehicles can be divided into two types: go straight and lane change.

In the maneuver recognition step, conventional approaches use SVM [5], [6] or probabilistic graphical models, such as HMM [3], [5], [7], Bayesian Network [4], random forest classifiers [8]. Some recent work also uses RNN [9], [10] to improve recognition accuracy. In the prediction step, conventional approaches use prototype trajectories method [2], [11], cluster-based models [3], kinetic models [12], polynomial models [13]–[15], Gaussian process [16], [17], RRT [18], Gaussian mixture models [1], [19], and so on. However, many previous approaches are not competent for long-term [4], [20] prediction, since the vehicle trajectories are nonlinear [9] and the features of motion models are complex. Therefore, some recent work starts to use RNN for long-term maneuver-based trajectory prediction [9].

The maneuver-based models have two advantages. First, the vehicle movements can be decomposed into several concise maneuvers that are relatively easy to recognize [2]. Second, the pre-defined maneuver categories are in agreement with human drivers' intuitions. So, maneuver-based models are usually interpretable.

However, maneuver-based models have two shortcomings that may lead to large prediction errors. First, when scenarios or vehicle motions become complex, it is hard to naturally divide the movement of a vehicle into several categories reasonably. For example, some distinctly dissimilar trajectories may be classified as the same maneuver, but they cannot be modeled by one simple trajectory model. Second, we have to label the maneuver of the trajectories for the training process manually; while labeling tasks are time-consuming and expensive. The labeling errors may increase the error of the trained model, too.

B. END-TO-END MODELS

As shown in Fig. 1(b), end-to-end models attempt to skip the maneuver recognition process and perform a trajectory prediction directly. Some studies prove that LSTM model is capable of modeling complex movement of vehicles or pedestrians, even without the maneuver recognition step [21]–[23]. Moreover, using end-to-end models can avoid errors caused by incorrect maneuver division.

However, two problems of end-to-end LSTM models still exist. First, the basic LSTM models cannot simultaneously model the spatial interactions between vehicles and trajectory sequences. Second, the basic LSTM models are hard to train when training **long-term** trajectory prediction models. When modeling long time series, such LSTM models are equivalent to a very deep neural network in the time dimension. Therefore, such LSTM models may suffer from the vanishing gradient problem [24] in the back-propagation.

To solve the two problems above, we propose a new model called ST-LSTM with two structural modifications to the basic LSTM models for trajectory prediction in this paper.

The first modification is that we adopt the idea of Structural-RNN (S-RNN) [25] and model all vehicle trajectories and interactions between vehicles by a new LSTM model. Specially, we construct the temporal relations and

spatial interactions as different time series and handle them separately by LSTM models. The new model is more suitable for traffic scenarios than those models (e.g. S-RNN) that have no clear division between temporal relations and spatial interactions.

The second modification is to introduce shortcut connections between the input and output of every LSTM layer, aiming to directly pass the prior information of historical trajectories to the subsequent layers. This structure can alleviate the gradient vanishment in the back-propagation [26].

We train and evaluate ST-LSTM on the NGSIM I-80 and US-101 datasets. The obtained prediction results yield higher accuracy than the result of one the-state-of-the-art model (i.e., the M-LSTM in [9]), which proves the effectiveness of our modifications.

To better present our findings, the rest of this paper is arranged as follows. *Section II* presents the problem we study, declares the notations of this paper and lists the prediction steps of ST-LSTM. *Section III* introduces the structure of ST-LSTM in detail. *Section IV* presents the detailed training process, presents the results of the experiments and discusses the effectiveness of ST-LSTM. Finally, *Section V* concludes the paper.

II. PROBLEM PRESENTATION AND THE MODEL

A. THE FRAMEWORK OF ST-LSTM

1) RESEARCH SCENARIO

In this paper, we study the end-to-end **long-term** trajectory prediction in **dense traffic**. The “long-term” here means that the model is capable to predict the trajectories of the entire process of nontrivial movements (movements except going straight) while keeping a low prediction error. The “dense traffic” here means that every vehicle can influence trajectories of its surrounding vehicles, but the road is not fully blocked. In such scenarios, the motion of the vehicle is complicated due to the influence of surrounding vehicles.

We denote the object vehicle of trajectory prediction as V_s . We assume that one surrounding vehicle V_i can affect the future motion of V_s , if and only if it is close to the vehicle of interest. If the longitudinal (along with the lane) distance between V_i and V_s is greater than 80 m, we will ignore the impact of V_i on V_s . We only pay attention to the six closest surrounding vehicles in six directions (left-front, front, right-front, right-rear, rear, left-rear), which are denoted as $V_1 \sim V_6$; see Fig. 2.

2) INPUT AND OUTPUT

We denote the historical trajectories of V_s and $V_1 \sim V_6$ as \mathbf{x}_s^t and $\mathbf{x}_i^t (i = 1, 2, \dots, 6)$. As shown in Fig. 2, the input of our end-to-end model is $\mathbf{x}^t = [\mathbf{x}_s^t, \mathbf{x}_1^t, \mathbf{x}_2^t, \dots, \mathbf{x}_6^t]$. The output \mathbf{y}_s^t is the predicted trajectory of V_s .

We express vehicle trajectories as sequences of position displacements $[\Delta X, \Delta Y]$. X and Y are the coordinates in the lateral and longitudinal directions respectively. That is,

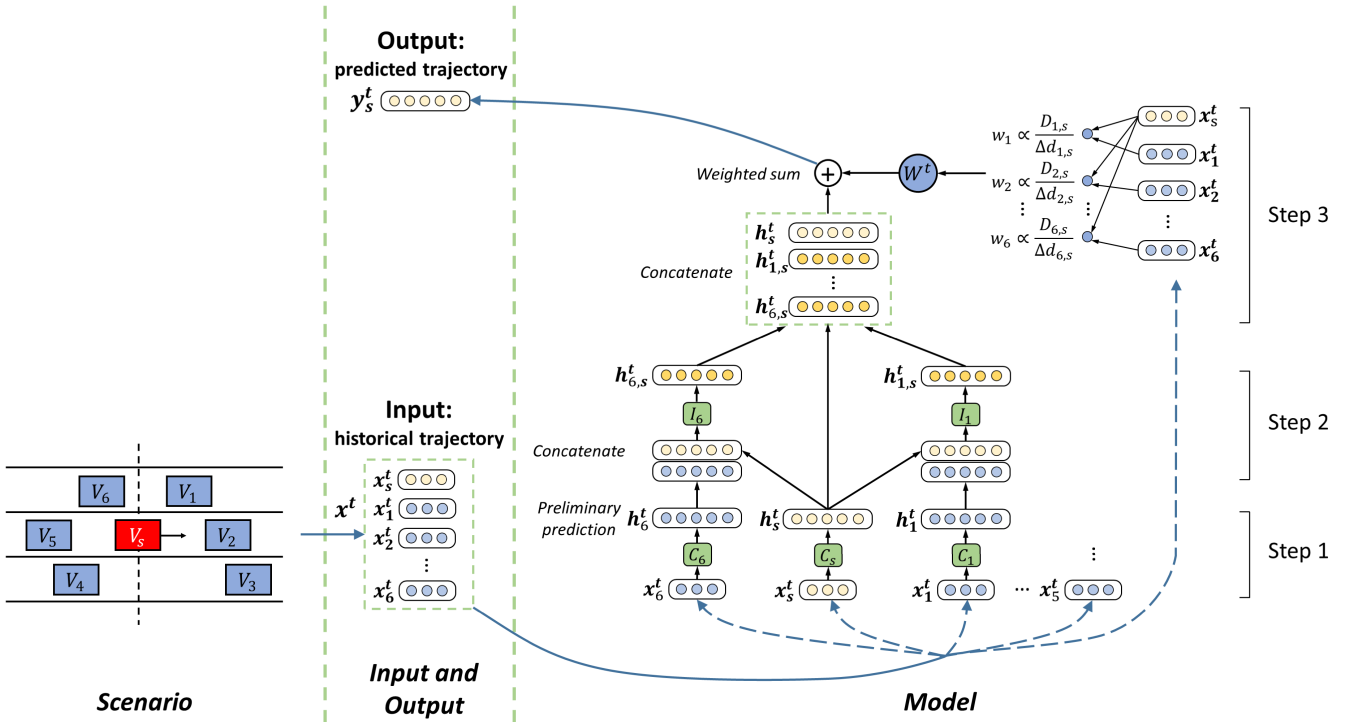


FIGURE 2. The overall structure and calculation process of ST-LSTM.

the inputs and the output are:

$$\begin{cases} \mathbf{x}_s^t = [\Delta X_s^{(t-t_h+\Delta t)}, \Delta Y_s^{(t-t_h+\Delta t)}, \dots, \Delta X_s^t, \Delta Y_s^t] \\ \mathbf{x}_i^t = [\Delta X_i^{(t-t_h+\Delta t)}, \Delta Y_i^{(t-t_h+\Delta t)}, \dots, \Delta X_i^t, \Delta Y_i^t] \end{cases} \quad (1)$$

$$\mathbf{y}_s^t = [\Delta X_s^{(t+\Delta t)}, \Delta Y_s^{(t+\Delta t)}, \dots, \Delta X_s^{(t+t_p)}, \Delta Y_s^{(t+t_p)}] \quad (2)$$

where t is the current time. t_h defines the historical time horizon. t_p defines the prediction time horizon. Δt defines the prediction step size. $[\Delta X_s^t, \Delta Y_s^t]$ and $[\Delta X_i^t, \Delta Y_i^t]$ define the position displacements of V_s and V_i from the time $(t - \Delta t)$ to t . The number of prediction time $n_p = t_p / \Delta t$. The number of historical time $n_h = t_h / \Delta t$.

3) THE STRUCTURE OF ST-LSTM

We model all vehicle trajectories and all interactions by LSTM models¹ and integrate them into a high-level structure. As shown in Fig. 2, the LSTM model of the trajectory of V_i (V_s) is denoted as C_i (C_s). The LSTM model of the interaction between V_i and V_s is denoted as I_i . For simplicity, only part of the LSTM models are displayed in the figure.

B. A THREE-STEP TRAJECTORY PREDICTION

As shown in Fig. 2, we divide the overall trajectory prediction process into three steps:

¹If not specified, the LSTM model in the rest of this paper represents the modified LSTM model proposed in Section III-B

Step 1: preliminarily predict the trajectories of V_i (V_s) by C_i (C_s).

The input of C_i (C_s) is the historical trajectory \mathbf{x}_i^t (\mathbf{x}_s^t). The output is denoted as \mathbf{h}_i^t (\mathbf{h}_s^t), which is a preliminary predicted trajectory of V_i (V_s):

$$\begin{cases} \mathbf{h}_i^t = [\Delta X_i^{(t+\Delta t)}, \Delta Y_i^{(t+\Delta t)}, \dots, \Delta X_i^{(t+t_p)}, \Delta Y_i^{(t+t_p)}] \\ \mathbf{h}_s^t = [\Delta X_s^{(t+\Delta t)}, \Delta Y_s^{(t+\Delta t)}, \dots, \Delta X_s^{(t+t_p)}, \Delta Y_s^{(t+t_p)}] \end{cases} \quad (3)$$

Step 2: evaluate the spatial interactions by I_i and outputs trajectory correction sequences of V_s .

The inputs of I_i are \mathbf{h}_s^t and \mathbf{h}_i^t . The output $\mathbf{h}_{i,s}^t$ is a trajectory correction sequence of \mathbf{h}_s^t , which considers the impact of spatial interaction between V_i and V_s on the movement of V_s . The length of $\mathbf{h}_{i,s}^t$, \mathbf{h}_s^t and \mathbf{h}_i^t are all equal to $2n_p$.

Step 3: correct the predicted trajectory of V_s and output the final predicted trajectory.

The inputs of this step are \mathbf{h}_s^t and all the $\mathbf{h}_{i,s}^t$, the final output prediction sequence \mathbf{y}_s^t is calculated by the weighted sum of the inputs:

$$\mathbf{y}_s^t = \mathbf{h}_s^t + \sum_i w_i \mathbf{h}_{i,s}^t = \mathbf{h}_s^t + (\mathbf{W}^t)^T \mathbf{H}^t \quad (4)$$

where $\mathbf{W}^t = [w_1, w_2, \dots, w_6]^T$ and $\mathbf{H}^t = [\mathbf{h}_{1,s}^t, \mathbf{h}_{2,s}^t, \dots, \mathbf{h}_{6,s}^t]^T$.

The weight w_i reflects the degree of the influence of I_i on the future trajectory of V_s . We can estimate w_i from the perspective of safe distance, since every vehicle actively

keeps maintaining safe distances from surrounding vehicles and thus generates spatial interactions.

In this paper, we measure the safe distance by the formula of safe gap proposed in [27], [28]. Let V_l (V_f) be the leading (following) vehicle between V_i and V_s . The safe distance between the two vehicles is:

$$D_{i,s} = v_f \rho + \frac{(v_f)^2 - (v_l)^2}{2a_{brake}} + L = \frac{\bar{v}\Delta v}{a_{brake}} + (v_f \rho + L) \quad (5)$$

where v_l (v_f) is the longitudinal velocity of V_l (V_f). L is the average length of the two vehicles. ρ is the mean response time of drivers. a_{brake} is the brake deceleration of the two vehicles. The average velocity $\bar{v} = (v_f + v_l)/2$. The relative velocity $\Delta v = v_f - v_l$.

This safe distance imagines a car-following scenario, the leading vehicle brakes by a_{brake} until a full stop, and the following vehicle keeps uniform during the response time ρ , and then brakes by a_{brake} until a full stop. This formula can be considered as (a lower bound of) safe distance because when two vehicles are closer than $D_{i,s}$, the following vehicle may not be able to avoid collision by braking.

We suppose $\alpha = 1/a_{brake}$ and $\beta = v_2\rho + L$, and furtherly simplify β as a constant. We consider a kind of a priori knowledge of weight as follows:

$$w_i \propto \frac{D_{i,s}}{\Delta d_{i,s}} = \frac{\alpha\bar{v}\Delta v + \beta}{\Delta d_{i,s}} \quad (6)$$

where $\Delta d_{i,s}$ defines the longitudinal distance between V_i and V_s . The constant terms α and β need to be determined during the training process. All the w_i should be normalized before performing **Step 3**.

In Eq.(6), the increase of \bar{v} or Δv and the decrease of $\Delta d_{i,s}$ will lead to an increase of w_i . It indicates that V_i will have a stronger influence on V_s , which is in agreement with our intuitions.

If V_i does not exist or $\Delta d_{i,s}$ is too large, we could directly set $w_i = 0$ so that the lack of V_i will not affect the prediction process. Therefore, ST-LSTM can be applied to vehicles with an arbitrary number of surrounding vehicles, which reflects the flexibility of ST-LSTM.

III. MODEL DESIGN

A. THE STRUCTURE AND FEATURES OF ST-LSTM

To enable the LSTM models simultaneously model spatial interactions between vehicles and temporal relations between trajectories series, Jain *et al.* proposed in [25] a high-level spatio-temporal model, which is called Structural-RNN (S-RNN). The key idea is to consider the spatial interactions as time series which can be handled by LSTM models.

As shown in Fig. 2, we adopt the idea of S-RNN and construct a spatio-temporal model similarly. However, there are two main differences between ST-LSTM and S-RNN:

The first difference is the definitions of the outputs of C_i and I_i (h_i^t and $h_{i,s}^t$). In S-RNN, the outputs of C_i and I_i are not specified. In contrast, ST-LSTM clearly distinguish h_i^t (temporal relations) and $h_{i,s}^t$ (spatial interactions) to

highlight their different roles on trajectory prediction. This is because temporal relations usually dominate the overall driving actions rather spatial interactions.

The second difference is the weight W^t in **Step 3**. In S-RNN, the outputs of all the I_i are integrated by the weighted sum that are directly learned during training. Differently, we evaluate w_i based on our prior knowledge of the influence degree of interactions shown in Eq.(4)-(6). The introduction of this prior knowledge will also help to accelerate learning.

According to the comparison, ST-LSTM is more suitable for characterizing traffic scenarios than S-RNN.

B. THE SHORTCUT CONNECTIONS FOR LSTM MODELS

Since the basic LSTM model is hard to train in long-term trajectory prediction, we perform some modifications to the structure of basic LSTM models, as shown in Fig. 3.

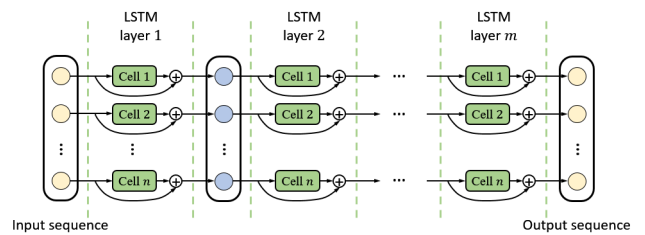


FIGURE 3. The structure of modified LSTM models with shortcut connections.

Inspired by ResNet [26], we introduce shortcut connections to solve the vanishing gradient problem. The information of the input sequence can be transmitted to subsequent layers directly, thus the modified LSTM model can alleviate the gradient vanishment when performing back-propagation. In other words, the learning object of the modified LSTM model becomes a residual-like sequence instead of a trajectory sequence. We will prove that this change can make LSTM easier to train in *Section IV-D*.

In order to construct shortcut connections, the length of input and output sequences should be equal ($n = 2n_h$). The number of LSTM layers (m in Fig. 3) and the number of LSTM cells in each layer (n in Fig. 3) are two main structure parameters of the modified LSTM model.

C. NETWORK PARAMETER SELECTION

Now we focus on the detailed network structure of each LSTM (C_i and I_i). Specifically, we need to set the historical time horizon t_h ($n = 2n_h = 2t_h/\Delta t$) and the number of hidden LSTM layers N_{hidden} ($m = N_{hidden}$).

The values of t_h and N_{hidden} can directly affect the performance of the modified LSTM model. In general, when t_h is too small, it is unable to provide enough valid historical information for trajectory prediction; when t_h is too large, redundant historical information may be harmful to prediction accuracy. When N_{hidden} is too small, the LSTM model may be incapable of modeling the complex vehicle trajectory; when N_{hidden} is too large, the network may be too deep and converge slower.

The detailed network parameter selection of all the C_i and I_i is shown in *Appendix A*. The selection result is shown in *Section IV-B*.

IV. TRAINING RESULTS AND DISCUSSIONS

A. DATASET

The research scenario in this paper is a straight road with multiple lanes. We can divide the vehicle motions in this scenario into two types: go straight and lane change.

We use the NGSIM I-80 and US-101 trajectory datasets to train and evaluate ST-LSTM.

The I-80 dataset was collected on the eastbound direction of Interstate 80 in Emeryville, California on April 13, 2005, containing data from 4:00 p.m. to 4:15 p.m., 5:00 p.m. to 5:15 p.m. and 5:15 p.m. to 5:30 p.m. The US-101 dataset was collected on the southbound direction of U.S. Highway 101 in Los Angeles, California on June 15, 2005, containing data from 7:50 a.m. to 8:05 a.m., 8:05 a.m. to 8:20 a.m., 8:20 a.m. to 8:35 a.m. The experimental scenes of the two datasets are both straight roads with dense traffic flow, including a large number of lane changes and car-following movements. Therefore, the datasets fit well with the scenario we study in this paper.

In the I-80 dataset, 2052, 1836, 1790 trajectories are collected respectively in the three periods of time. The three datasets respectively contain 1025, 913, 945 successful lane change. In the US-101 dataset, 2169, 2017, 1915 trajectories are collected in the three periods of time, containing 1006, 660, 657 successful lane change respectively. Each data frame includes the vehicle's position, velocity, yaw angle, size, etc. The sampling frequency of the dataset is 10 Hz, thus we set $\Delta t = 0.1s$ in this paper.

TABLE 1. Statistics of different types of trajectories in I-80.

	4:00- 4:15 p.m.	5:00- 5:15 p.m.	5:15- 5:30 p.m.	Total	Balanced
left only	531	494	478	1503	672
Right only	91	58	75	224	672
Both	47	36	40	123	123
Total lane change	669	588	593	1850	1467
No lane change	1383	1248	1197	3828	300
Total	2052	1836	1790	5678	1767

However, as shown in TABLE 1 and TABLE 2, both of the datasets are seriously unbalanced. Therefore, we downsample the sequences of **left only** and **no lane change** and oversample the sequences of **right only** to ensure the proportions of different kinds of trajectories are roughly equal [29].

To facilitate the learning of the characteristics of lane change, we trim every trajectory with lane change and only reserve part of the data that contains lane change. We also randomly trim the trajectories without lane change to guarantee the balance of different types of trajectories.

We randomly sample N trajectories from the balanced datasets (shown in TABLE 1 and TABLE 2) for training, leaving the remaining trajectories for testing. We will discuss

TABLE 2. Statistics of different types of trajectories in US-101.

	7:50- 8:05 a.m.	8:05- 8:20 a.m.	8:20- 8:35 a.m.	Total	Balanced
left only	296	269	305	870	706
Right only	141	124	88	353	706
Both	164	90	76	330	177
Total lane change	601	483	469	1553	1589
No lane change	1568	1534	1446	4548	330
Total	2169	2017	1915	6101	1919

the selection of training set scale N in *Appendix B*. The selection result is shown in *Section IV-B*.

B. TRAINING DETAILS

According to the parameter selection in *Appendix A*, we set $N_{hidden} = 3$ and $t_h = 3s$ for all the C_i and I_i . That is, all the C_i and I_i have three hidden layers and each layer has 60 LSTM cells. According to the training set scale selection in *Appendix B*, we respectively adopt the critical scale (with 580 trajectories) and a bigger scale (with 1350 trajectories) to train ST-LSTM models.

Since most vehicles have similar trajectory characteristics, all the C_i in ST-LSTM share a public LSTM model. In contrast, Since the impacts of surrounding vehicles in different directions on V_s are different, each I_i is trained independently and does not share network parameters with each other.

ST-LSTM is trained in a step-by-step process using Tensorflow. We first pre-train C_i , then we pre-train I_i based on C_i . All the pre-train LSTM models are trained with an initial learning rate of 0.001 for about 20k iterations. Finally, all the pre-train LSTM models are united and trained for about 5k iteration to train the constant terms α and β of Eq.(6).

During training, the batch size N_b of each training step is set as 50×50 , that is, each batch has 50 trajectories and the length of each trajectory sequence is 50 (that is, 5 s). All the network parameters are randomly initialized. Different from other parameters, α and β have specific physical meanings, but we have tested that the initialization values of α and β have little impact on the overall training result. We use 50% dropout in the training process. The optimizers adopt ADAM optimizer. The loss function of the training process adopts MSE (mean square error) between the predicted displacement sequence y_s^t and the ground truth displacement sequence y^t .

$$y^t = [\Delta \hat{X}^{(t+\Delta t)}, \Delta \hat{Y}^{(t+\Delta t)}, \dots, \Delta \hat{X}^{(t+t_p)}, \Delta \hat{Y}^{(t+t_p)}] \quad (7)$$

The MSE loss function of each single trajectory is:

$$loss = \sum_{i=1}^{n_p} (\delta_i^t)^T \delta_i^t \quad (8)$$

where $\delta_i^t = \left[a(\Delta X^{(t+i\Delta t)} - \Delta \hat{X}^{(t+i\Delta t)}), \Delta Y^{(t+i\Delta t)} - \Delta \hat{Y}^{(t+i\Delta t)} \right]^T$. We set the constant $a = 10$ in this paper to magnify the impact of the lateral error on the loss function.

The overall MSE loss function is:

$$Loss = \frac{1}{N_b} \sum loss \quad (9)$$

where N_b is the total number of trajectories in the training dataset.

C. TESTING RESULTS

We compare the RMS prediction error between the following models: Maneuver-LSTM (M-LSTM, proposed in [9]), ST-LSTM trained by 580 trajectories (denoted as ST-LSTM-580) and ST-LSTM trained by 1350 trajectories (denoted as ST-LSTM-1350). According to our statistics, the average time of lane change processes is $4 \sim 5s$, thus we set the prediction horizon $t_p = 6s$. Since $t_p > t_h$, we should repeat the prediction process for several times to perform a larger horizon trajectory prediction.

Here, the RMS prediction error is calculated from the predicted trajectory and the ground truth trajectory. Thus we need to accumulate the displacement sequence y_s^t and y^t to get trajectory representations z_s^t and z^t :

$$z_s^t = [X^{(t+\Delta t)}, Y^{(t+\Delta t)}, \dots, X^{(t+t_p)}, Y^{(t+t_p)}] \quad (10)$$

$$z^t = [\hat{X}^{(t+\Delta t)}, \hat{Y}^{(t+\Delta t)}, \dots, \hat{X}^{(t+t_p)}, \hat{Y}^{(t+t_p)}] \quad (11)$$

$$\left\{ \begin{aligned} X^{(t+i\Delta t)} &= \sum_{j=1}^i \Delta X^{(t+j\Delta t)} \\ Y^{(t+i\Delta t)} &= \sum_{j=1}^i \Delta Y^{(t+j\Delta t)} \\ \hat{X}^{(t+i\Delta t)} &= \sum_{j=1}^i \Delta \hat{X}^{(t+j\Delta t)} \\ \hat{Y}^{(t+i\Delta t)} &= \sum_{j=1}^i \Delta \hat{Y}^{(t+j\Delta t)} \end{aligned} \right. \quad (12)$$

Therefore, the RMS prediction error of each single trajectory given prediction horizon P is:

$$rms^P = \left(\frac{1}{P} \sum_{i=1}^P (\sigma_i^t)^T \sigma_i^t \right)^{\frac{1}{2}} \quad (13)$$

$$RMS^P = \frac{1}{N_t} \sum rms^P \quad (14)$$

where $\sigma_i^t = [X^{(t+i\Delta t)} - \hat{X}^{(t+i\Delta t)}, Y^{(t+i\Delta t)} - \hat{Y}^{(t+i\Delta t)}]^T$. N_t is the total number of trajectories in the testing dataset.

The RMS values are shown in TABLE 3, which illustrates that both ST-LSTM-580 and ST-LSTM-1350 outperform M-LSTM. From the comparison between ST-LSTM-580 and ST-LSTM-1350, we can find that the latter model uses a much larger training set but only gets a slight improvement in prediction accuracy. This phenomenon proves the effectiveness of ‘‘critical point’’, which can guide us to properly reduce the training set, losing a little prediction accuracy in exchange for a significant increase in training speed.

TABLE 3. RMS value of prediction error.

Prediction horizon	M-LSTM [9]	ST-LSTM-580		ST-LSTM-1350	
		I-80	US-101	I-80	US-101
1	0.58	0.59	0.62	0.54	0.58
2	1.26	1.23	1.28	1.16	1.21
3	2.12	1.95	2.01	1.88	1.97
4	3.24	2.80	2.89	2.70	2.85
5	4.66	3.78	3.95	3.63	3.89
6	-	4.70	5.06	4.64	5.04

To analyze the main source of the trajectory prediction error, we calculate the mean and standard deviation of velocity deviation (denoted as VD_i) in different prediction horizons, as shown in Fig. 4. And we plot the mean absolute deviation of position deviation (denoted as PD_i) of different prediction horizons in Fig. 5. The definitions of VD_i and PD_i are:

$$VD_i = \frac{1}{\Delta t} \left[\Delta X^{(t+i\Delta t)} - \Delta \hat{X}^{(t+i\Delta t)}, \Delta Y^{(t+i\Delta t)} - \Delta \hat{Y}^{(t+i\Delta t)} \right]^T \quad (15)$$

$$PD_i = \sigma_i^i \quad (16)$$

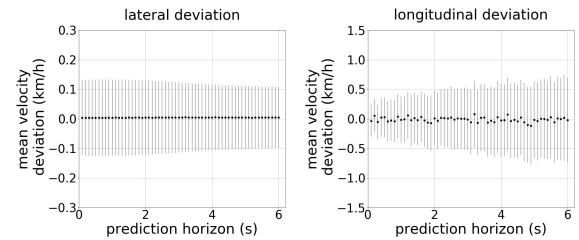


FIGURE 4. Statistics of mean and standard deviation of velocity deviation (ST-LSTM-1350 on I-80 dataset).

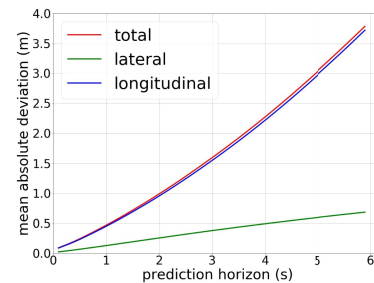


FIGURE 5. Mean absolute deviation of position deviation (ST-LSTM-1350 on I-80 dataset).

Fig. 4 shows that the distributions of VD_i in different prediction horizons are roughly the same. The accumulation of velocity error dominates the total long-term prediction error. This is the reason that we use RMS of **position** error to evaluate ST-LSTM.

When the prediction horizon becomes too large, the mean absolute deviation increases and the prediction is severely

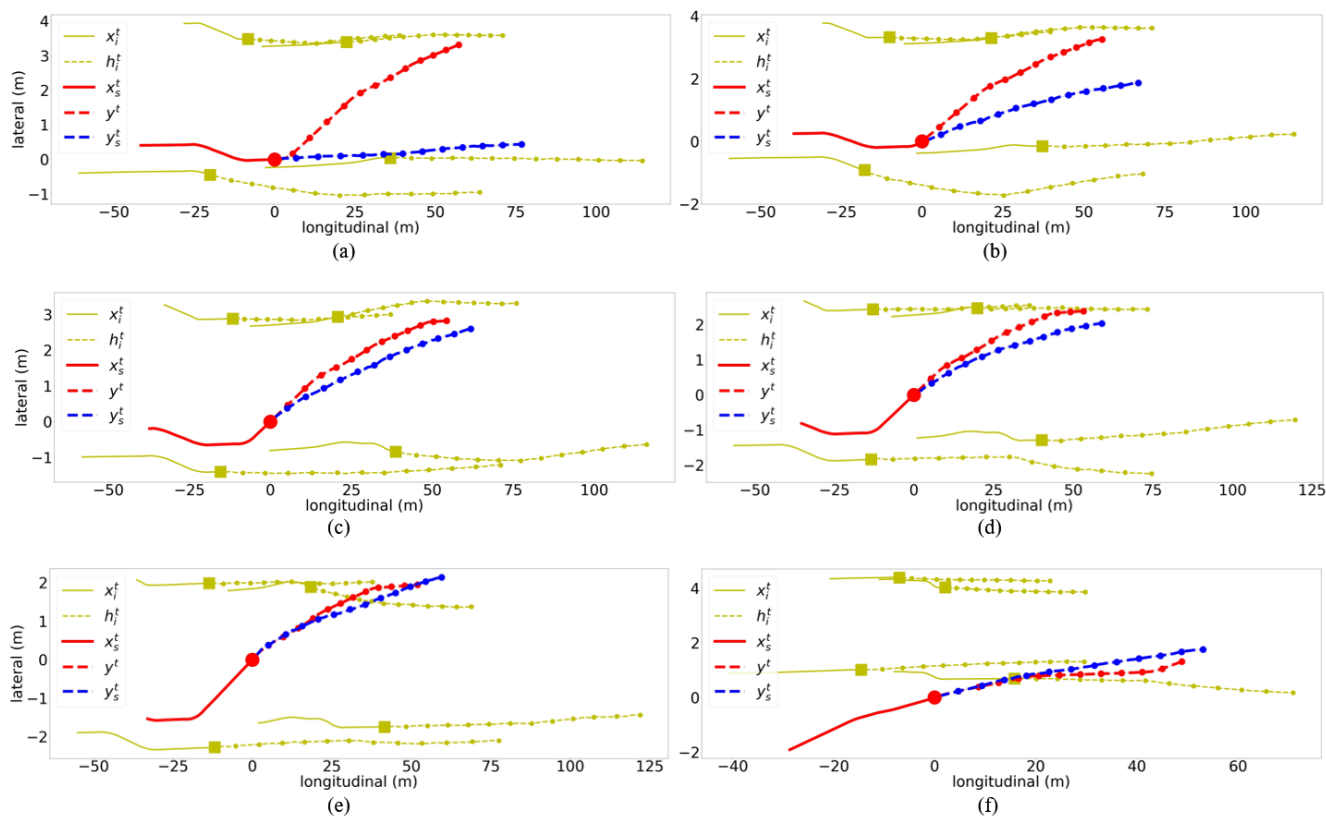


FIGURE 6. The illustration of predicted trajectory and ground truth trajectory of several snapshots of one typical case: (a) $T = 0\text{ s}$, (b) $T = 0.5\text{ s}$, (c) $T = 1\text{ s}$, (d) $T = 1.5\text{ s}$, (e) $T = 2\text{ s}$, (f) $T = 4\text{ s}$.

unstable due to the error accumulation. From another perspective, historical trajectories are not sufficient to provide effective information for the prediction of a large time interval. Therefore, the prediction horizon selected in this paper (6 s) is already large enough.

To more intuitively display the trajectory prediction results, we randomly select a lane change trajectory and observe the prediction results during the whole lane change process. We took six snapshots in this time interval, as shown in Fig. 6. Fig. 6 (a)(b) illustrate that the predicted trajectories do not yet present characteristics of lane change at the very beginning of the lane change process. The predicted trajectories in Fig. 6 (c)-(e) show typical characteristics of lane changes. Fig. 6 (f) illustrate that the predicted trajectory changes back to straight at the end of the lane change.

D. THE ROLE OF SHORTCUT CONNECTIONS

During the training process, we find that the modified LSTM model is easier to train and could reduce the RMS error to a low value.

In one experiment, we train another C_s by the basic LSTM model and compare it to the C_s in Section IV-C). Besides shortcut connections, everything of the two models is identical, such as input and output, N_{hidden} , t_h , the training process, etc. There was no manual intervention during the training

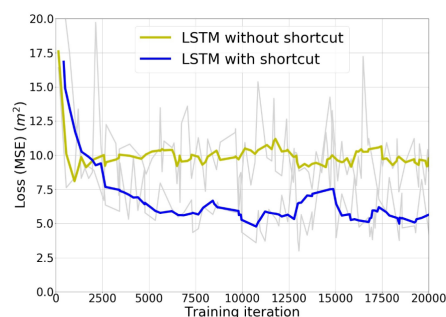


FIGURE 7. The loss variation of the two models during the training process.

process. The values of loss function are plotted in Fig. 7, indicating the modified LSTM model converge faster than the basic LSTM model.

V. CONCLUSION

In this paper, we propose a novel end-to-end long-term trajectory prediction model for dense traffic. To solve the two problems of long-term prediction in dense traffic, we modify the classical LSTM model by adding shortcut connections and model all the spatial interactions and temporal relations by the modified LSTM model. Finally, we construct a spatio-temporal trajectory prediction model. The experiments

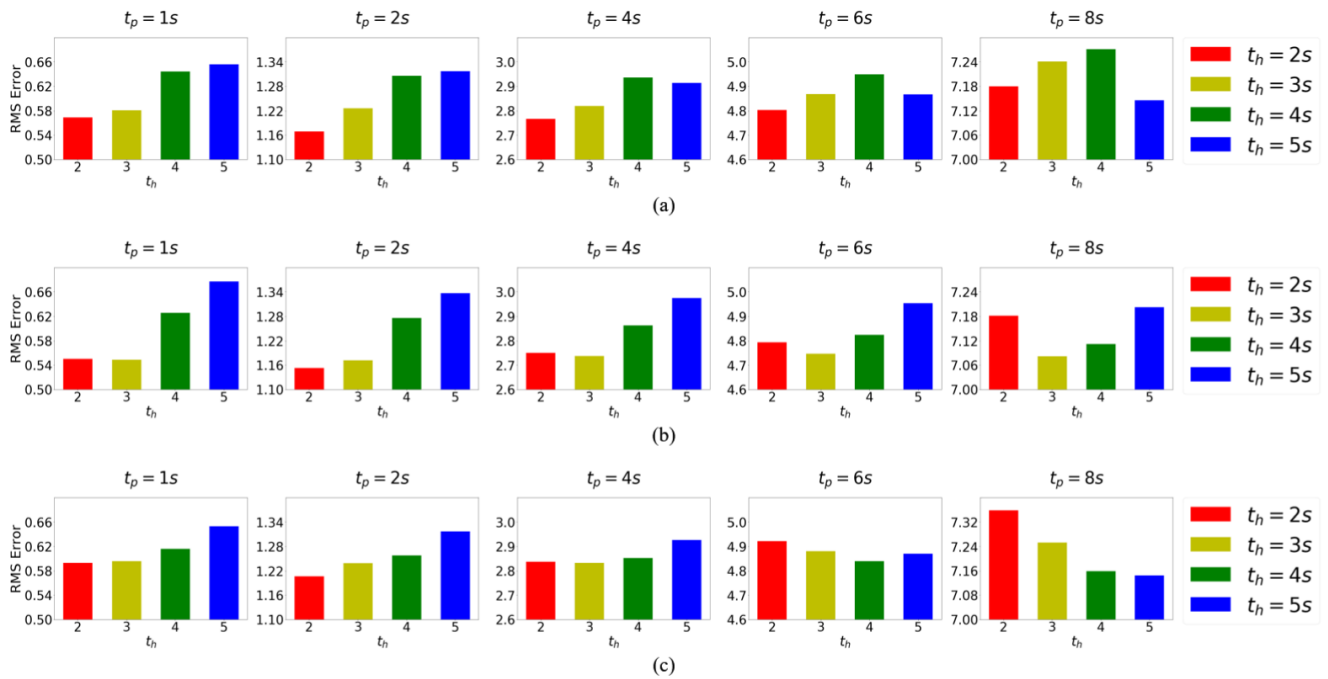


FIGURE 8. RMS error compare between modified LSTM models with different N_{hidden} and t_h : (a) $N_{hidden} = 2$, (b) $N_{hidden} = 3$, (c) $N_{hidden} = 4$.

on NGSIM I-80 and US-101 prove that ST-LSTM can perform more precise trajectory prediction than one state-of-the-art model (M-LSTM).

However, we only test ST-LSTM on NGSIM I-80 and US-101, which are relatively small datasets in single simple scenario. In the future work, we will collect more data and evaluate ST-LSTM in more complex scenarios.

APPENDIX A NETWORK PARAMETER SELECTION

We train C_i by several groups of $N_{hidden} \in \{2, 3, 4\}$ and $t_h \in \{2s, 3s, 4s, 5s\}$. In this experiment, we randomly select a training set with $N = 1200$ trajectories on the I-80 dataset. The detailed training process is the same as what is described in Section IV-B). We adopt 10-fold cross-validation on the training set to calculate mean RMS errors in different prediction horizons.

As shown in Fig. 8, we compare the RMS error of these models in different prediction horizons. We select the best group ($t_h = 3s$ and $N_{hidden} = 3$) as the parameters of C_i because the RMS error of the model with this group of parameters is almost always the lowest (the second lowest when $t_p = 2s$). We also test deeper networks and larger t_h , but those models do not perform better than the selected model.

Due to the limits of the model structure, the t_h of I_i should be equal to the t_h of C_i . We repeat the previous steps and find that $N_{hidden} = 3$ is also a good value for I_i .

Similarly, we try different training set scales and different dataset (i.e., US-101), and find that the above values of t_h and N_{hidden} are also relatively suitable for those training sets.

Therefore, we set $t_h = 3s$ and $N_{hidden} = 3$ for all the C_i and I_i in ST-LSTM.

APPENDIX B OVERFITTING ANALYSIS AND TRAINING SET SCALE SELECTION

Deep learning models (including ST-LSTM) require a large amount of data for training. Insufficient training data can lead to overfitting (low generalization capability), but excessive data also cannot infinitely improve the model generalization capability. In particular, it is often hard to obtain large-scale datasets for deep learning, so we need to find a “critical point” for data volume to reach a compromise between improving model generalization capability and reducing the need of training data. This critical point can also be considered as the minimum size of an effective training set.

In this paper, we respectively use training sets of different sizes to train ST-LSTM models and calculate their training and testing error (of different prediction horizons) to observe if the models are over-fitted.

We take the US-101 dataset as an example. According to Section IV-A), the balanced dataset includes a total of 1919 effective trajectories. We respectively randomly sample 40, 95, 190, 380, 580, 960, 1350, 1540 trajectories (about 2%, 5%, 10%, 20%, 30%, 50%, 70%, 80% of the total dataset) for training and plot the training and testing errors in Fig.9. The network parameters follow the results of Appendix-A. The detailed training process follows Section IV-B)

Fig.9 illustrates that the trained ST-LSTM occurs serious overfitting when the training set is too small. The overfitting

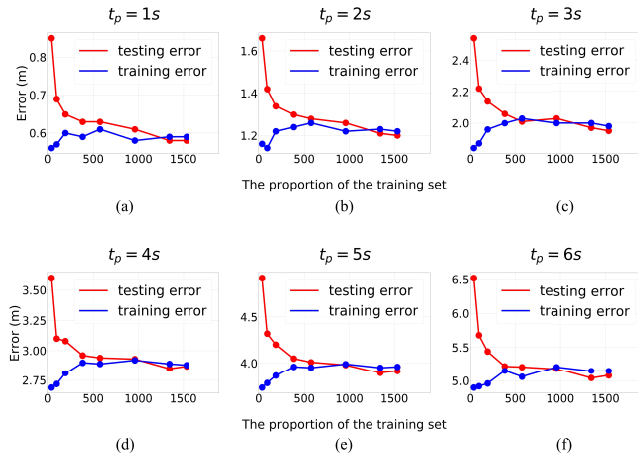


FIGURE 9. The comparison of training and testing error (of different prediction horizons) between ST-LSTM models trained by different sizes of training sets: (a) prediction horizon $t_p = 1s$, (b) $t_p = 2s$, (c) $t_p = 3s$, (d) $t_p = 4s$, (e) $t_p = 5s$, (f) $t_p = 6s$.

phenomenon is weakened as the number of trajectories increases, and it is almost eliminated when the training set contains more than 580 trajectories. After that, the model generalization capability only has slight improvement despite the large increase in training set volume. Therefore, we think that 580 trajectories are a suitable amount of training set for our ST-LSTM model. When training ST-LSTM model on other dataset, such as I-80, we built a training set with 580 random trajectories and could also get a well-trained model.

The above method explores the “marginal” performance of the model, that is, for a certain model, the “critical point” represents how much training data at least should we use for training. We believe that the method we proposed above is worth promoting. Base on “critical point”, we can directly compare the demands for training data of different deep network models. When we only have a limited scale of training data, we can select a model that has stronger generalization capability (and cannot over-fit) under that condition based on this index and achieve better performance. When the model is determined, we can also set a reasonable training set scale according to this index, so as to avoid the waste of computing resources caused by training data redundancy.

REFERENCES

- [1] N. Deo, A. Rangesh, and M. M. Trivedi, “How would surround vehicles move? A unified framework for maneuver classification and motion prediction,” *IEEE Trans. Intell. Vehicles*, vol. 3, no. 2, pp. 129–140, Jun. 2018.
- [2] S. Lefevre, D. Vasquez, and C. Laugier, “A survey on motion prediction and risk assessment for intelligent vehicles,” *Robomech J.*, vol. 1, p. 1, Jul. 2014.
- [3] B. T. Morris and M. M. Trivedi, “Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2287–2301, Nov. 2011.
- [4] M. Schreier, V. Willert, and J. Adamy, “Bayesian, maneuver-based, long-term trajectory prediction and criticality assessment for driver assistance systems,” in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst.*, Oct. 2014, pp. 334–341.
- [5] G. S. Aoude, V. R. Desaraju, L. H. Stephens, and J. P. How, “Driver behavior classification at intersections and validation on large naturalistic data set,” *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 724–736, Jun. 2012.
- [6] P. Kumar, M. Perrollaz, S. Lefevre, and C. Laugier, “Learning-based approach for online lane change intention prediction,” in *Proc. IEEE Intell. Vehicles Symp.*, Gold Coast, QLD, Australia, Jun. 2013, pp. 797–802.
- [7] C. Liu and M. Tomizuka, “Enabling safe freeway driving for automated vehicles,” in *Proc. Amer. Control Conf. (ACC)*, Jul. 2016, pp. 3461–3467.
- [8] J. Schlechtriemen, F. Wirthmueller, A. Wedel, G. Breuel, and K.-D. Kuhnert, “When will it change the lane? A probabilistic regression approach for rarely occurring events,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jul. 2015, pp. 1373–1379.
- [9] N. Deo and M. M. Trivedi, “Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstm,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1179–1184.
- [10] S. Patel, B. Griffin, K. Kusano, and J. J. Corso. (2018). “Predicting future lane changes of other highway vehicles using RNN-based deep models.” [Online]. Available: <https://arxiv.org/abs/1801.04340>
- [11] W. Yao, H. Zhao, P. Bonnifait, and H. Zha, “Lane change trajectory prediction by using recorded human driving data,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2013, pp. 430–436.
- [12] A. Hounou, P. Bonnifait, V. Cherfaoui, and W. Yao, “Vehicle trajectory prediction based on motion model and maneuver recognition,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 4363–4369.
- [13] C. Guo, C. Sentouh, B. Soualmi, and J.-B. Haué, and J.-C. Poupieul, “Adaptive vehicle longitudinal trajectory prediction for automated highway driving,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2016, pp. 1279–1284.
- [14] C. M. Kang, S. J. Jeon, S. Lee, and C. C. Chung, “Parametric trajectory prediction of surrounding vehicles,” in *Proc. IEEE Int. Conf. Veh. Electron. Saf. (ICVES)*, Jun. 2017, pp. 26–31.
- [15] D. Xu, Z. Ding, H. Zhao, M. Moze, F. Aioun, and F. Guillemard, “Naturalistic lane change analysis for human-like trajectory generation,” in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1393–1399.
- [16] M. K. C. Tay and C. Laugier, “Modelling smooth paths using Gaussian processes,” in *Field Service Robotics*. New York, NY, USA: Springer, 2008, pp. 381–390.
- [17] C. Laugier et al., “Probabilistic analysis of dynamic scenes and collision risks assessment to improve driving safety,” *IEEE Intell. Transp. Syst. Mag.*, vol. 3, no. 4, pp. 4–19, Oct. 2011.
- [18] G. Aoude, J. Joseph, N. Roy, and J. How, “Mobile agent trajectory prediction using Bayesian nonparametric reachability trees,” in *Proc. Infotech Aerosp.*, Jun. 2011, p. 1512.
- [19] J. Wiest, M. Höffken, U. Kreßsel, and K. Dietmayer, “Probabilistic trajectory prediction with Gaussian mixture models,” in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2012, pp. 141–146.
- [20] C. Hermes, C. Wohler, K. Schenk, and F. Kummert, “Long-term vehicle motion prediction,” in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2009, pp. 652–657.
- [21] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, “Social lstm: Human trajectory prediction in crowded spaces,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 961–971.
- [22] F. Althé and A. de La Fortelle, “An LSTM network for highway trajectory prediction,” in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 353–359.
- [23] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi, “Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network,” in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2017, pp. 399–404.
- [24] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2016.
- [25] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, “Structural-RNN: Deep learning on spatio-temporal graphs,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5308–5317.
- [26] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Sep. 2016, pp. 770–778.
- [27] S. Shalev-Shwartz, S. Shammah, and A. Shashua. (2017). “On a formal model of safe and scalable self-driving cars.” [Online]. Available: <https://arxiv.org/abs/1708.06374>

- [28] L. Li, X. Peng, F. Wang, D. Cao, and L. Li, "A situation-aware collision avoidance strategy for car-following," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 5, pp. 1012–1016, Sep. 2018.
- [29] A. Gosain and S. Sardana, "Handling class imbalance problem using oversampling techniques: A review," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2017, pp. 79–85.

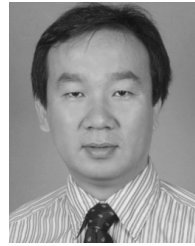


SHENGZHE DAI is currently pursuing the M.Eng. degree with the Department of Automation, Tsinghua University, Beijing, China. His research interests include machine learning, intelligent transportation systems, and intelligent vehicle.



LI LI (S'05–M'06–SM'10–F'17) is currently an Associate Professor with the Department of Automation, Tsinghua University, China, where he is involved in the fields of artificial intelligence, intelligent control and sensing, intelligent transportation systems, and intelligent vehicles. He has published over 80 SCI indexed international journal papers and over 50 international conference papers as a First/Corresponding Author. He serves as a member for the Editorial Advisory Board of

Transportation Research Part C: Emerging Technologies. He serves as an Associate Editor for the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.



ZHIHENG LI (M'05) received the Ph.D. degree in control science and engineering from Tsinghua University, Beijing, China, in 2009, where he is currently an Associate Professor with the Department of Automation. He is also with the Graduate School at Shenzhen, Tsinghua University, Shenzhen, China. His research interests include traffic operation, advanced traffic management systems, urban traffic planning, and intelligent transportation systems. He serves as an Associate

Editor for the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS.

• • •