Interactive Trajectory Prediction of Surrounding Road Users for Autonomous Driving Using Structural-LSTM Network

Lian Hou, Long Xin, Shengbo Eben Li, *Senior Member, IEEE*, Bo Cheng, and Wenjun Wang

***Abstract*— Accurate trajectory prediction of surrounding road users is critical to autonomous driving systems. In mixed traffic flows, road users with different kinds of behaviors and styles bring complexity to the environment, which requires considering interactions among road users when anticipating their future trajectories. This paper presents a long-term interactive trajectory prediction method for surrounding vehicles using a hierarchical multi-sequence learning network. In contrast to non-interactive method which assumes that road users are independent of each other, this method can automatically learn high-level dependencies among multiple interacting vehicles through the proposed structural-LSTM (long short-term memory) network. Specifically, structural-LSTM first assigns one LSTM for each interacting vehicle. Then these LSTMs share their cell states and hidden states with their spatial-neighboring LSTMs by a radial connection, and recurrently analyze the output state of itself as well as the other LSTMs in a deeper layer. Finally based on all output states, the network predicts trajectories for surrounding vehicles. The proposed method is evaluated on the NGSIM dataset, and its results show that satisfyingly accurate prediction performance of long-term trajectories of surrounding vehicles is accessible, e.g., longitudinal and lateral RMS error can be reduced to less than 1.93m and 0.31m over 5s time horizon, respectively.**

***Index Terms*— Autonomous driving, trajectory prediction, interaction, surrounding road user, LSTM.**

I. Introduction

AUOTONOMOUS vehicles (AVs) are believed to have the great potential to improve road safety, reduce traffic congestion and relieve drivers from driving burden. To guarantee driving safety in complex traffic environment, an AV should be able to anticipate the traffic environment in the future and respond to these changes appropriately. However, the motion of road users surrounding the AV is often difficult to predict since it is affected by various factors such as the randomness of driver/bicyclist/pedestrian behaviors, strong interactions with other road users, and spatial constraints of road geometry. In addition, the observation for road users by on-board environment sensors is usually imperfect and highly noisy due to the environment complexity and sensor uncertainty. Therefore, it is challenging to implement accurate online trajectory prediction of road users in complex traffic environment, so as to facilitate the decision making and path planning system for higher level of autonomous driving [1], [2].

Among all kinds of road users, surrounding vehicles (SVs) are the most common and fundamental targets of trajectory prediction in many scenarios, especially in highway. Classic works use vehicle kinematic or dynamic models along with filtering technologies, e.g., Kalman filters, for trajectory prediction [3], [4]. They are computationally efficient and generally effective for short-term prediction (i.e., 1s to 2s in the future), but quickly lose accuracy over a longer time horizon. In order to extend prediction horizon with enough prediction accuracy, researchers make effort on studying driver intention and maneuver [5], [6], and then use prototype trajectories to make predictions based on the recognized driving patterns [7]. The predictions are quite satisfying but constrained by the limited number of prototypes. For more temporal adaptability and trajectory flexibility, Xin *et al.* implements recurrent neural networks (RNNs), which can learn both spatial and temporal structure of long trajectories. Such method outperforms the other methods in long-term prediction (i.e., 3s to 5s in the future) [8].

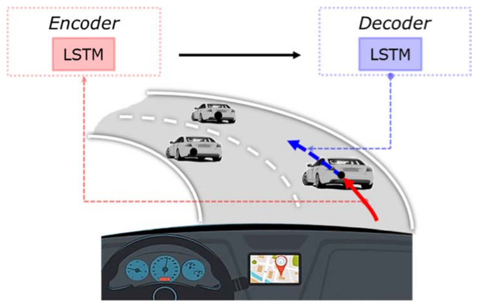
So far, most of pioneering RNN-based works have assumed that the future trajectories of SVs are independent of each other. As Fig. 1(a) illustrated, this kind of method generally observes the historic trajectory of a single SV by the first LSTM (called "encoder") and then passes what it understands to the second LSTM (called "decoder") to predict the future trajectory of this SV. Even though [11] and [12] extract features of multiple SVs and feed all of them to the encoder LSTM, no interactions among the SVs but only a single trajectory of the target SV is analyzed and predicted in the decoder LSTM (see Fig. 1 (b)). In this paper, we aim at predicting long-term trajectories of multiple SVs by designing a two-layer structural-LSTM network and learning interactions among SVs in both the encoder and the decoder as Fig. 1 (c) shown.

LSTMs are able to recognize and predict long sequences, but one single LSTM cannot capture dependencies between

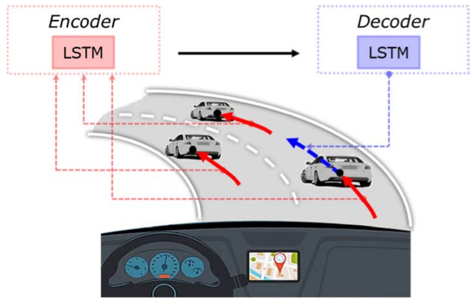
Manuscript received January 21, 2019; revised June 18, 2019; accepted September 12, 2019. This work was supported by the National Key Research and Development Program of China under Grant 2017YFB0102603 and in part by the NSF China under Grant 51575293, Grant U1664263, and Grant 51622504. The Associate Editor for this article was W. Lin. *(Lian Hou and Long Xin contributed equally to this work) (Corresponding author: Shengbo Eben Li.)*

The authors are with the State Key Laboratory of Automotive Safety and Energy, School of Vehicle and Mobility, Tsinghua University, Beijing 100084, China (e-mail: houl16@mails.tsinghua.edu.cn; xin-l13@mails.tsinghua.edu.cn; lishbo@tsinghua.edu.cn; chengbo@tsinghua.edu.cn; wangxiaowenjun@tsinghua.edu.cn).

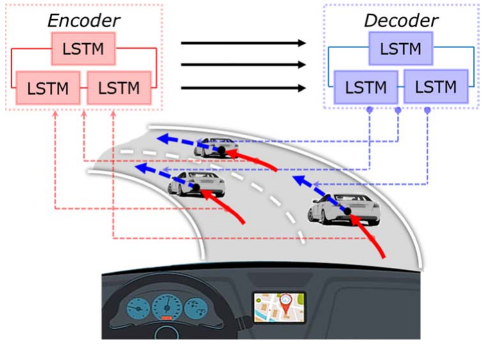
Digital Object Identifier 10.1109/TITS.2019.2942089



(a) Single-input, single-LSTM, and single-output [8][9][10]



(b) Multi-input, single-LSTM, and single-output [7][11][12]



(c) Multi-input, multi-LSTM, and multi-output [This paper]

Fig. 1. Different mechanisms of trajectory prediction using LSTMs.

multiple correlated sequences. In our work, we employ and connect multiple two-layer LSTMs corresponding to spatial-nearby trajectories. In particular, during every state updating step, we assign one LSTM to each SV as well as its neighboring vehicles to extract spatial-temporal features of each historic trajectory, and then connect these neighboring LSTMs, share their hidden states with each other, and analyze their interactions in the deeper LSTM layer, based on which we predict future trajectories. This hierarchical network, which is inspired by structural-RNN [13], is referred to as the structural-LSTM. The proposed method replaces the standard LSTM as structural-LSTM for both the encoder and the decoder. Inputs for the encoder are sequentially transformed coordinates of SVs in the past 10 seconds, while the outputs of the decoder are future locations of SVs in the next 5 seconds.

The contribution of this paper is proposing radial connection structures for state sharing among multiple LSTMs and designing a new trajectory representation method for the two-layer encoder-decoder architecture to learn interactions among vehicles and improve long-term trajectory prediction with this guidance.

The rest of this paper is organized as follows. In Section II, related works about trajectory prediction are summarized. In section III, the problem is formally defined, and the proposed method with its related techniques are introduced. In Section IV, the method application is described and the results are discussed. The paper is concluded in Section V.

II. Related Works

An extensive survey on vehicle motion prediction models has been presented by Lefèvre *et al.* [14], where the models are categorized into physics-based, maneuver-based and interaction-aware models. Since then, many RNN-based methods have been proposed for maneuver recognition, trajectory prediction and interaction modeling. In this section, we give a review on these works to clarify the different mechanisms of trajectory prediction, especially LSTM-based methods.

A. Non-Interactive Predictions

Implemented with object tracking techniques, vehicle motion models such as kinematic models or dynamic models have been used for trajectory prediction [3], [4]. Since the observation is often limited, Kalman filter has been widely used for prediction by taking the uncertainty in vehicle model into consideration [3]. In order to account for more influential factors and improve the prediction accuracy further, Bayesian filtering techniques such as the context-dependent interactive multiple model filter [15] and Monte-Carlo method [4] have been proposed. These methods, while taking into account the physical limitations of a vehicle, are normally effective for short-term trajectory prediction, i.e. one or two seconds in the future, and are not accurate enough for long-term prediction, which will directly affect the performance of decision making and path planning.

One popular alternative to the challenges mentioned above is to take advantage of the prototype trajectory so that prediction can be performed by comparing the current trajectory with the learned motion patterns and using the prototype trajectory as a base model for future motion [14]. These prototype vehicle trajectories can be learned through Gaussian process [16], [17] and Gaussian mixture model [18], or multi-kernel-based shrinkage method [19]. The downside of Gaussian model is the expensive online computation load of calculating the probability similarity of the current trajectory with the prototype one. Besides, it is time-dependent, i.e. trajectories falling into the waiting intervals when a vehicle stops have to be manually dropped out [20], [21]. When motion patterns are represented by a finite set of prototype trajectories, the similarity of a partial trajectory to a motion pattern is measured by metrics such as the modified Hausdorff [22]. One disadvantage of such method is that using a finite set of trajectories would take a very large number of prototypes to model various patterns of the real-world driving trajectories. Another difficulty is the adaption to different road geometry, as the learned prototype trajectory models can only be applied in a similar road layout.

One alternative to trajectory prototypes is to first estimate the maneuver intention of the driver (e.g. waiting at the stop line, following another vehicle, executing a left turn) and then to predict the successive physical states so that they correspond to a possible execution of the identified maneuver [14]. A major advantage over trajectory prototypes is that there is no need to match the partial trajectory with a previously observed trajectory. Instead, higher-level characteristics are extracted and used to recognize maneuvers, which makes it easier to generalize the learned model to arbitrary layouts. For identifying maneuvers in complex scenarios, discriminative learning algorithms are very effective, such as Multi-Layer Perceptrons [23], LSTM [24], Logistic regression [25], Relevance Vector Machines [26], Support Vector Machines [27], [28], Hidden Markov Model [29], [32], or tube-and-droplet-based method [30], [31]. Given the driver's maneuvers, trajectories are predicted to match the identified maneuvers in a deterministic manner [33] or a probabilistic manner such as Gaussian Process [16] or Rapidly-exploring Random Tree [27].

Besides these methods, an alternative end-to-end method, the LSTM-based encoder-decoder architecture has been more and more popular. It is first proposed for machine translation task and has an ability to read and generate a sequence of arbitrary length, i.e., sequence-to-sequence [34]. Many recent researches used a LSTM as an encoder to understand driver's intention from the historic trajectory and then used another LSTM as a decoder to generate the future trajectory as Fig. 1 (a) shown [8]–[10]. However, in practice, the assumption that vehicles are moving independently of each other does not hold. Vehicles share the road with other vehicles, and the maneuvers performed by one vehicle will strongly influence the maneuvers of the other vehicles, especially at road intersections where priority rules force vehicles to take into account the other vehicles. Disregarding these dependencies can lead to erroneous interpretations of the situations, and deteriorates the evaluation of the risk.

B. Interactive Prediction

Interaction-guided motion models represent vehicles as maneuvering entities which interact with each other, i.e. the motion of a vehicle is assumed to be influenced by the motion of the other vehicles in the scene. Taking into account the dependencies between the vehicles leads to a better interpretation of their motion compared to the Intention-guided motion models described in the previous section. As a result, it contributes to a better understanding of the situation and a more reliable evaluation of the risk. Despite this, there are few interaction-guided motion models in the literature. They use Dynamic Bayesian Networks [35], Kalman Neural Networks [36], or LSTM [7], [11], [12], [37]. Nevertheless, such models are usually computationally expensive for the model training and may be not compatible with real-time predicting as the number of interacting objects in the model increases. More problematically, such models understand only the interactions among the historic trajectories of different SVs but consider no interaction when predicting the future trajectories. For example, as shown in Fig. 1 (b), the single-LSTM method inputs the trajectories of multiple SVs but only outputs the trajectory of a single SV. From another perspective,

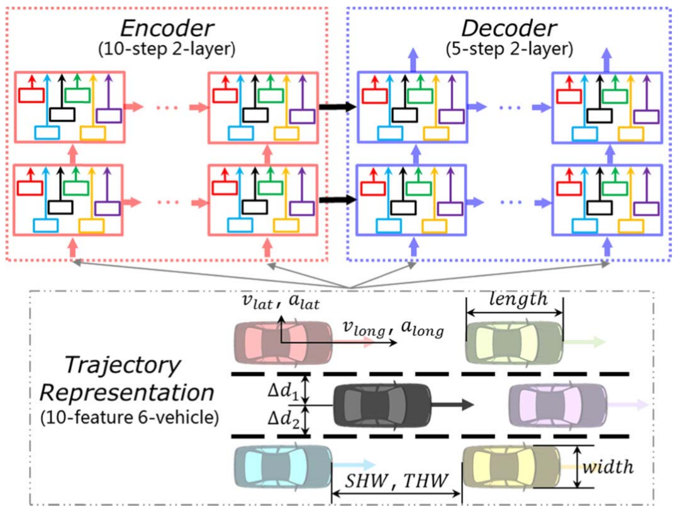


Fig. 2. The overall pipeline of the proposed trajectory predicting method.

such interaction-guided methods model only the influence of the other SVs on the target SV in the encoder whereas ignore the influence of the target SV on the other SVs in the decoder, i.e., the interactions are unidirectional. For a higher reasonability, we proposed the structural-LSTM method to model bidirectional interactions and predict trajectories of multiple SVs as Fig. 1 (c) shown. This idea is consistent with [37] but we use an end-to-end architecture without intention pattern layers.

III. Proposed Method

A. Problem Definition

We select a group of totally six SVs located in three adjacent lanes for each sample in the problem as Fig. 2 shown: (a) target SV, whose trajectory is to be predicted for accuracy evaluation; (b) front SV, which is in front of the target SV and in the same lane; (c) left front SV, which is closest to the target SV in longitudinal distance in front and in the left adjacent lane; (d) left rear SV, which is closest to the target SV in longitudinal distance in rear and in the left adjacent lane; (e) right front SV, which is closest to the target SV in longitudinal distance in front and in the right adjacent lane; (f) right rear SV, which is closest to the target SV in longitudinal distance in rear and in the right adjacent lane. This is similar as the work in [7], [12] except that we delete the rear SV which is behind the target SV. Note that whenever the center of the target SV crosses the lane marking, the IDs of the other SVs change correspondingly. Different from the work in [9], [10] where vehicle trajectories were represented in the form of occupancy grid and the prediction task turned into a classification problem, we consider it as a regression one whose outputs are supposed to be as accurate as actual values.

More formally, our goal is to train a predictor for future trajectory of sequential outputs $Y = \{y_t^k | t \in \mathcal{T}_{\text{pred}}, y^k \in \mathcal{F}\}$ based on a set of observable feature inputs $X = \{x_t^k | t \in \mathcal{T}_{\text{past}}, x^k \in \mathcal{F}\}$, where $\mathcal{T}_{\text{past}} = \{-T_{\text{past}}, \ldots, -1, 0$ and $\mathcal{T}_{\text{pred}} = \{1, 2, \ldots, T_{\text{pred}}$ are respectively the time intervals of the historical input and future predictions, $\mathcal{F}$ is the feature set acquired and calculated simultaneously. For $x^k \in \mathcal{F}$ and $t \in T_{\text{pred}}$, we denote x_t^k by the value of feature x^k observed t time steps earlier. Similarly, we denote y_t^k by the value of output y^k generated t time steps in the future.

B. Structural-LSTM Model

1) LSTM Cell: RNNs are distinguished from traditional feed-forward networks by its internal states and cycles, which are capable of analyzing sequential information and learning temporal features. LSTM, a particular RNN, was developed to avoid vanishing gradients of the loss function over time.

The LSTM has a memory state called "cell" which stores the interpretation of past input data. The cell is updated based on the current input and the previous cell state. Between the input and the cell are different gates, a unique control mechanism that enables the LSTM to learn when to forget past state and update the state when given new input. Let c_t be memory cell state at time step t and h_t be the output hidden state, then c_t and h_t are updated by the following equations:

$$i_t = \sigma(w_{x^i} x_t + w_{h^i} h_{t-1} + b_i) \tag{1}$$
$$f_t = \sigma(w_{x^f} x_t + w_{h^f} h_{t-1} + b_f) \tag{2}$$
$$o_t = \sigma(w_{x^o} x_t + w_{h^o} h_{t-1} + b_o) \tag{3}$$
$$g_t = f(w_{x^c} x_t + w_{h^c} h_{t-1} + b_c) \tag{4}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{5}$$
$$h_t = o_t \odot f(c_t) \tag{6}$$

where $\sigma(x) = 1/1 + e^{-x}$ is the sigmoid function; $f(\cdot)$ is any activation function; i_t, f_t, o_t and g_t are input gate vector, forget gate vector, output gate vector and state update vector, respectively; w_{x^i}, w_{x^f}, w_{x^o}, w_{x^c}, w_{h^i}, w_{h^f}, w_{h^o}, w_{h^c} are the weights for linear combination; b_i, b_f, b_o and b_c are the relative bias; $\odot$ is element-wise production.

2) Encoder-Decoder Architecture: We used a two-layer encoder-decoder architecture for the predicting system as illustrated in Fig. 3. The architecture employs two networks (i.e., structural-LSTMs in our method) called the encoder and the decoder. The encoder processes the input sequence $X = \{x^k_{-T_{\text{past}}}, \ldots, x^k_{-2}, x^k_{-1}$ of the length T_{past} and produces the summary of the past input sequence through the cell state vector c_t (including $c_{t,1}$ for the first layer and for $c_{t,2}$ the second layer)and the hidden state vector h_t (including $h_{t,1}$ for the first layer and for $h_{t,2}$ the second layer). After T_{past} times of recurrent updates by repeating the equations (1) to (6), the encoder has input a T_{past}-long sequence and updates the cell state from initial state c_{init} (equals to zero) to final state c_{-1} as well as the hidden state from initial state (equals to zero) to final state h_{-1}. Then, the encoder passes c_{-1} and h_{-1} to the decoder and the decoder uses them as initial cell state and hidden state (i.e., $c'_{\text{init}} = c_{-1}$, $h'_{\text{init}} = h_{-1}$). The first input of decoding step is the current observed input x_0^k. In every next decoding step, the decoder inputs the output y^k_{t-1} obtained in the previous step, repeats the equations (1) to (6) for T_{pred} times and generates the output sequence $Y = \{y_1^k, y_2^k, \ldots, y^k_{T_{\text{pred}}}\}$.

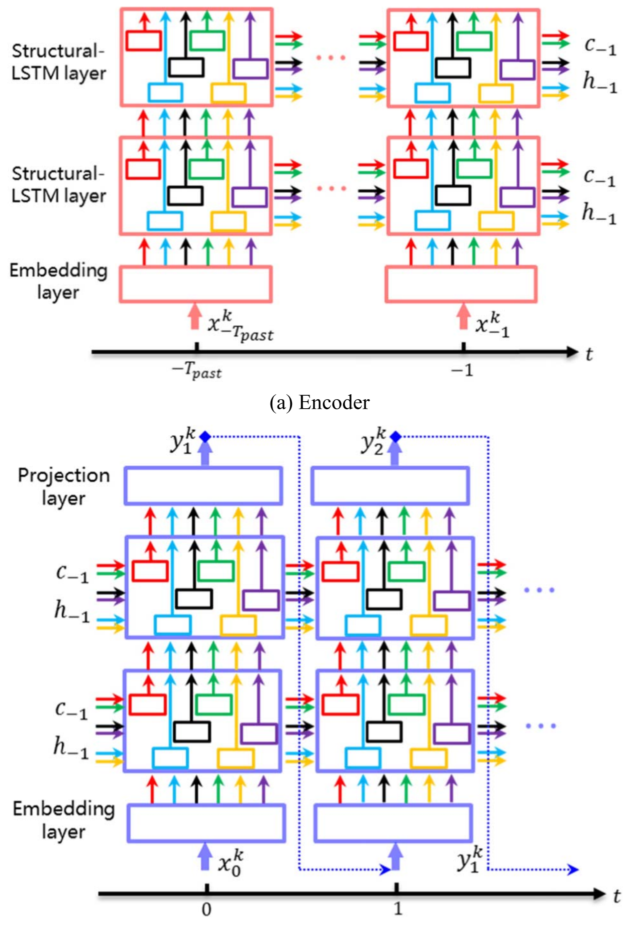


Fig. 3. The two-layer encoder-decoder architecture.

In this architecture, the role of the first network (encoder) is to analyze the interactions among vehicles, and transform the features into cell states and hidden states as the understanding of the network while the second network (decoder) is to extract this understanding result of the sequential inputs and thus generate future trajectories. Note that the outputs of the decoder are derived by applying the affine transformation followed by a projecting function that scale the outputs to suit expected features.

In the encoder (the first Structural-LSTM, for historical trajectory inputting), there may be two part of trajectories of the target SV's surrounding vehicles, but the target SV's lane change trajectory is only one part. In the decoder (the second Structural-LSTM, for future trajectory predicting), only the latter part of trajectories of the target SV's surrounding vehicles as well as the lane change trajectory of the target vehicle is predicted. That is, the predicted trajectories are consistent with the current observed part of trajectories. The former part of trajectories is only used for providing more historical information.

C. Structural-LSTM Network

Instead of simply concatenating the features of all the SVs into one vector and feed it to a single LSTM cell as done in [7], [11], which considers the interactions among SVs as a black box, we extend the single LSTM cell into a structural-LSTM network including multiple LSTM cells to processes the inputs separately and learn the interaction among the SVs hierarchically, as Fig. 4 shown. For the first structural-LSTM as the encoder:

$$\left[x_t^{\mathrm{t}}, x_t^{\mathrm{f}}, x_t^{\mathrm{lf}}, x_t^{\mathrm{lr}}, x_t^{\mathrm{rf}}, x_t^{\mathrm{rr}}\right] = x_t^k \tag{7}$$

$$c_t^{\mathrm{f}}, h_t^{\mathrm{f}} = \mathrm{LSTM}^{\mathrm{f}}\left([x_t^{\mathrm{t}}, x_t^{\mathrm{f}}, x_t^{\mathrm{lf}}, x_t^{\mathrm{rf}}], c_{t-1}^{\mathrm{f}}, h_{t-1}^{\mathrm{f}} | \boldsymbol{W}^{\mathrm{f}}\right) \tag{8}$$

$$c_t^{\mathrm{lf}}, h_t^{\mathrm{lf}} = \mathrm{LSTM}^{\mathrm{lf}}\left([x_t^{\mathrm{t}}, x_t^{\mathrm{lf}}, x_t^{\mathrm{f}}, x_t^{\mathrm{lr}}], c_{t-1}^{\mathrm{lf}}, h_{t-1}^{\mathrm{lf}} | \boldsymbol{W}^{\mathrm{lf}}\right) \tag{9}$$

$$c_t^{\mathrm{lr}}, h_t^{\mathrm{lr}} = \mathrm{LSTM}^{\mathrm{lr}}\left([x_t^{\mathrm{t}}, x_t^{\mathrm{lr}}, x_t^{\mathrm{lf}}], c_{t-1}^{\mathrm{lr}}, h_{t-1}^{\mathrm{lr}} | \boldsymbol{W}^{\mathrm{lr}}\right) \tag{10}$$

$$c_t^{\mathrm{rf}}, h_t^{\mathrm{rf}} = \mathrm{LSTM}^{\mathrm{rf}}\left([x_t^{\mathrm{t}}, x_t^{\mathrm{rf}}, x_t^{\mathrm{f}}, x_t^{\mathrm{rr}}], c_{t-1}^{\mathrm{rf}}, h_{t-1}^{\mathrm{rf}} | \boldsymbol{W}^{\mathrm{rf}}\right) \tag{11}$$

$$c_t^{\mathrm{rr}}, h_t^{\mathrm{rr}} = \mathrm{LSTM}^{\mathrm{rr}}\left([x_t^{\mathrm{t}}, x_t^{\mathrm{rr}}, x_t^{\mathrm{rf}}], c_{t-1}^{\mathrm{rr}}, h_{t-1}^{\mathrm{rr}} | \boldsymbol{W}^{\mathrm{rr}}\right) \tag{12}$$

$$c_t^{\mathrm{t}}, h_t^{\mathrm{t}} = \mathrm{LSTM}^{\mathrm{t}}\left([x_t^{\mathrm{t}}, x_t^{\mathrm{f}}, x_t^{\mathrm{lf}}, x_t^{\mathrm{lr}}, x_t^{\mathrm{rf}}, x_t^{\mathrm{rr}}], c_{t-1}^{\mathrm{t}}, h_{t-1}^{\mathrm{t}} | \boldsymbol{W}^{\mathrm{t}}\right) \tag{13}$$

where LSTM$(\cdot)$ is the state updating of LSTM cells (i.e., short for (1) to (6)); $\boldsymbol{W}$ is the trainable weights of LSTM cells; superscript t, f, lf, lr, rf, rr represents the target SV, front SV, left front SV, left rear SV, right front SV and right rear SV, respectively; $[\cdot,\cdot]$ means concatenating several vectors into one vector; cell states c_t and hidden states h_t of all the LSTM cells are initialized to zero at the start of the recurrent updating.

As shown in Fig. 4, each vehicle is modeled by an LSTM cell, whose state recurrence is described as equation (8) to (13). The number of input arrows in the figure is the same as the number of input vectors in $[\cdot,\cdot]$ in the equations, which depends on the number of neighbor vehicles. All six SV models are in the same types of inputs, outputs and LSTM cells, except for the different length of input. That means the interaction between the SVs are bidirectional and symmetric. But each of six LSTM cells is different with each other in the input size and trainable weights.

It means that the six different LSTMs are expected to be functioning in the same way both for the target SV and other SVs when the SVs' input features are different.

Instead of using a single LSTM as the decoder and predicting trajectories of only the target SV in the work of [12], we make the decoder a structural-LSTM which is similar as the encoder except that inputs are gained from outputs which is predicted in the previous step:

$$\left[x_t^{\mathrm{t}}, x_t^{\mathrm{f}}, x_t^{\mathrm{lf}}, x_t^{\mathrm{lr}}, x_t^{\mathrm{rf}}, x_t^{\mathrm{rr}}\right] = \boldsymbol{V} \cdot \left[h_t^{\mathrm{t}}, h_t^{\mathrm{f}}, h_t^{\mathrm{lf}}, h_t^{\mathrm{lr}}, h_t^{\mathrm{rf}}, h_t^{\mathrm{rr}}\right] \tag{14}$$

where $\boldsymbol{V}$ is the trainable weights of the linear projection function. Note that the trainable weights of LSTM cells in the decoder is also a different set, denoted as $\boldsymbol{W}'$.

During each prediction, we use six LSTMs to independently analyze the interactions between each SV and its neighboring SVs, respectively, as (8) to (13) shown. Each of the LSTMs input the features of a SV and its neighboring SVs. Then the

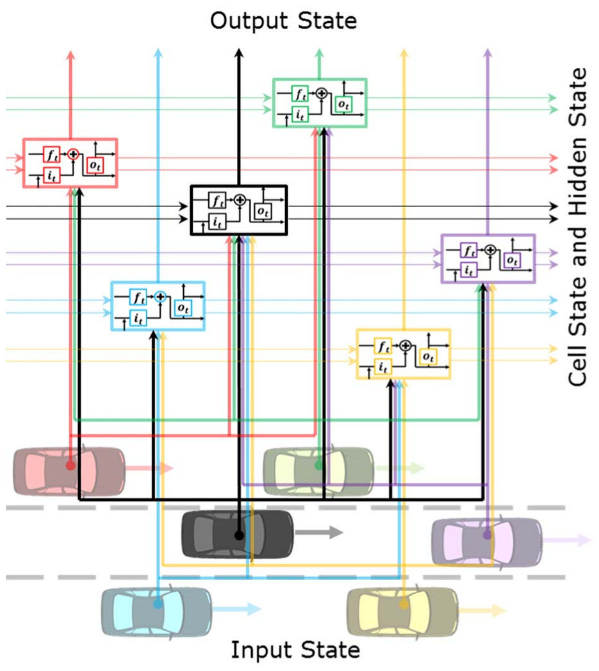


Fig. 4. The spatial-temporal graph of the structural-LSTM. It models the interactions among the SVs by sharing their states with each other in radial connections and passes the cell states and hidden states to the next steps so that it can use the interactions it learned to predict trajectories in the future.

hidden states of all the LSTMs are passed to a deeper layer to extract more features (repeat (8) to (13)) or the projection layer to outputs the prediction results as (14) shown. This is because using only one layer Structural-LSTM is actually still a "Multi-input, single-LSTM, and single-output" type of trajectory prediction as Fig. 1(b) shows. In another word, there is no difference between one-layer Structural-LSTM and six independent multi-input LSTMs. Only by stacking multiple (at least 2) layers of Structural-LSTM can these LSTM cells share their hidden states thus connecting with each other. In addition, the more complicated the environment is, the more layers of Structural-LSTM should we stack.

By the design of Structural-LSTM network, at each time step, all six SVs' trajectories are predicted and each SV's trajectory is predicted by inputting its neighbor SV's historical trajectory (i.e., considering the influence of other SVs on the target SV as well as the influence of the target SV on other SVs). That means our proposed method models the bidirectional interactions between the target SV and the other SVs rather than the unidirectional influence of the other SVs on the target SV.

There are two ways to predict trajectories for all six vehicles: (1) use one Structural-LSTM to predict all of them; (2) use six Structural-LSTMs to predict each of them as the target SV, respectively. The former costs less computational resource but leads to larger error for the other five SVs. The latter provides smaller error but costs more computational resource. We use the second way because these six Structural-LSTMs can be computed in a distributed way, which is suitable for online application. The target SV is for outputting final result, so its interactions with the other vehicles are all considered. The other five SVs are for predicting target SV in the next time step, so it is feasible if their interactions are partially considered.

D. Training the Model

Since our goal is the high predicting accuracy of the target SV and meanwhile an acceptable predicting accuracy of the other SVs, we use weighted RMSE (root mean squared error) of the output features at all predicted time-steps as the loss function:

$$J = \sqrt{\frac{1}{T_{pred}} \sum_{k=1}^{K} \boldsymbol{w}_k \sum_{t=1}^{T_{pred}} (y_t^k - \hat{y}_t^k)^2} \tag{15}$$

where $\boldsymbol{w}_k$ is the weight on error of the k-th feature y_t^k related to the observed ground truth $\hat{y}_t^k$.

The trainable weight set θ includes $\boldsymbol{W}$, $\boldsymbol{W}'$, $\boldsymbol{V}$. In each training step, the loss is calculated and weights are updated through BPTT (backpropagation through time) to minimize the loss:

$$\theta \leftarrow \theta - \eta \cdot \frac{\partial J}{\partial \theta} \tag{16}$$

IV. Evaluation

A. Data

The dataset used in this paper is from the Next Generation Simulation (NGSIM) [38]. Collected and published by the US Federal Highway Administration in 2005, the NGSIM is one of the largest open datasets of naturalistic driving and has been widely studied in the literature, e.g. [7], [8], [11], [12], [39].

More specifically, the area of interest is the I-80 freeway in Emeryville, California, of which the covered segment is approximately 500m in length and 6 lanes (3.66m or 12ft each) in width (see Fig. 5). The 45-minute trajectory data were collected from 4:00pm to 4:15pm (transition period) and from 5:00pm to 5:30pm (rush hour), reflecting different traffic characteristics during transitional and congested traffic period, respectively.

The dataset contains more than 5000 trajectories of individual vehicles, with a sampling rate at 10 Hz. Each sample in one trajectory includes the information such as instantaneous speed, acceleration, longitudinal and lateral positions (both local and global), vehicle length and width, vehicle type, lane ID, vehicle ID, etc. The local coordinates is set at the up-left point of the study area, where x is the lateral position of the vehicle relative to the leftmost edge of the road, and y its longitudinal position to the entry edge.

B. Features

Totally 48 features are extracted for the trajectory predicting on I-80 data, eight features for each of the six SVs in our Structural-LSTM model. We use 9999 as the input feature value for the missing SV before training the network to learn such kind of symbolization about missing SVs. This symbolization works because all the normal values are smaller than 100 and an input vector with extreme large element numbers will be recognized as a signal that there is no vehicle. Once such symbolization of missing vehicles is learned, Structural-LSTM shall perform well even though there are less than six vehicles in some scenarios.

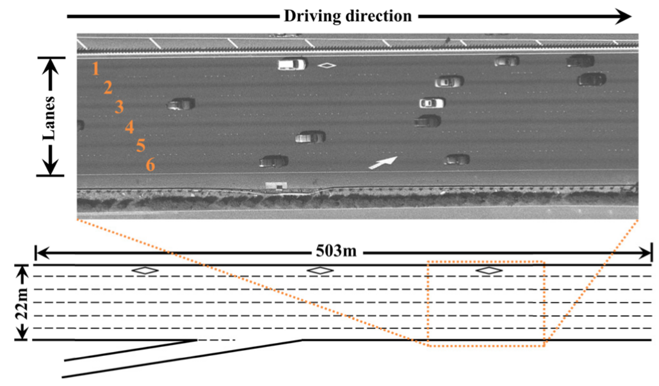


Fig. 5. Birdview of naturalistic traffic recorded within I-80 freeway study area.

1) Target SV: Relative Lateral Position (Two Features). Lateral distance from the center of the target SV to the left lane marking and right lane marking.

Relative Longitudinal Position (Two Features). Time headway (THW) and space headway (SHW) related to the front SV, gained from the raw data of THW and SHW in the dataset. If there is no front SV, the value is denoted as 9999. The calculation of THW is based on current instantaneous velocity:

$$t_{\mathrm{THW}} = \frac{x - x_{\mathrm{f}}}{v_{\mathrm{x}}} \tag{17}$$

where v_{x} is the current instantaneous longitudinal velocity of target SV, x is the longitudinal position of the front center of target SV, x_{f} is the longitudinal position of the front center of front SV.

Lateral and Longitudinal Velocity. Gained from the differentiation of the local lateral and longitudinal position instead of the raw data of velocity in the dataset.

Lateral and Longitudinal Acceleration. Gained from the differentiation of the lateral and longitudinal velocity.

2) Other SVs: Relative Lateral and Longitudinal Position. Distance from the center of each SV to the center of the target SV.

Relative Lateral and Longitudinal Velocity. The difference between the velocity of each SV and that of the target SV.

Relative Lateral and Longitudinal Acceleration. The difference between the acceleration of each SV and that of the target SV.

Relative Width and Length. The difference between the width and length of each SV and that of the target SV.

In case of the missing of any SV in the samples, all its eight features are all denoted as 9999. The weights on the 48 features of the weighted RMSE loss function (i.e., $\boldsymbol{w}_k$ in (15)) is set as follows: 10 for the lateral and longitudinal velocity of the target SV; 5 for the other features of the target SV; 1 for all the features of the other SVs.

C. Implementation Details

1) Data Preprocessing: We sampled the target SV trajectories from the dataset with a horizon of 15s (10s for understanding, 5s for prediction) every 5s. Then we searched and matched the other SVs' ID and trajectories. Then these trajectories were smoothed by the first order Savitzky-Golay filter. Finally, we downsampled the data at 1Hz and got a preprocessed dataset with totally 6381 lane changing samples and 53216 lane keeping samples. In addition, we got a training set with totally 49613 samples and a testing set with 9984 samples by random selecting.

2) Training Details: We used an embedding dimension of 32 for the aforementioned features before feeding them to Structural-LSTM to scale the different features into similar magnitude. We used the two-layer LSTM cell for all the LSTM cells in the Structural-LSTM with a fixed hidden state dimension of 128. Layer Normalization was implemented before each layer [40]. SeLU was used as the activation function for all the LSTM cells [41]. A batch size of 256 is used and the network is trained for 200 epochs using Adam optimizer with an initial learning rate of 0.0001 [42]. The global norm of gradients is clipped at a value of 1 to ensure stable training. The Structural-LSTM model was trained on a single TITAN-Xp GPU with a Tensorflow-1.6 implementation [43].

3) Predicting Details: We firstly used quadratic interpolation to sample the lateral and longitudinal velocity of the target SV at 10Hz from the 1Hz outputs of the model. Then we gained the lateral and longitudinal position from the integral of the velocity with the initiate local position.

D. Results

We evaluate our results with two different metrics:

Average Displacement Error. The root mean square error (RMSE) over all the predicted position points of a trajectory in five different predicting horizons (0s to 1.0s, 0s to 2.0s, 0s to 3.0s, 0s to 4.0s and 0s to 5.0s).

Final Displacement Error. The distance between predicted final position point and true final position point at ends of the five predicting horizons (1.0s, 2.0s, 3.0s, 4.0s and 5.0s).

In Table I, II and III, we compare the performance of our model with state-of-art methods as well as multiple control settings:

Structural-LSTM. This is our proposed model that two Structural-LSTMs composing in encoder-decoder (sequence-to-sequence, seq2seq) architecture. One high-level LSTM and five low-level LSTMs input features of the six SVs, respectively.

Seq2seq-6. This is a simplified setting of our model where we use naive LSTM instead of Structural-LSTM and simply input the concatenated features of the six SVs.

Seq2seq-1. This is a simplified setting of our model where we use naive LSTM instead of Structural LSTM and simply input the features of the target SV.

Dual LSTMs. Proposed by [8], this model uses one LSTM to recognize driver's lane changing intention and another LSTM to generate trajectories based on the intention outputs.

TABLE I
PREDICTION TIME

Model	Prediction Time
Structural-LSTM	0.12s
Seq2seq-6	0.11s
Seq2seq-1	0.09s
Dual LSTMs	0.09s
Single LSTM	0.05s
Linear model	$<$0.01s

TABLE II
AVERAGE DISPLACEMENT ERROR

Model	Prediction Horizon				
	0~1s	***0~2s***	***0~3s***	***0~4s***	***0~5s***
Structural-LSTM	0.20	0.50	0.70	0.87	0.99
Seq2seq-6	0.20	0.51	0.73	0.93	1.08
Seq2seq-1	0.21	0.52	0.76	1.04	1.31

(a) Longitudinal position error (m)

Model	Prediction Horizon				
	0~1s	***0~2s***	***0~3s***	***0~4s***	***0~5s***
Structural-LSTM	0.07	0.15	0.19	0.20	0.21
Seq2seq-6	0.07	0.15	0.19	0.21	0.22
Seq2seq-1	0.06	0.15	0.19	0.21	0.23

(b) Lateral position error (m)

Single LSTM. Proposed by [11], this model uses one naive LSTM to input features of ten SVs and generate the trajectory of the SV in the center.

Linear Model. We extrapolate the trajectories with assumption of a constant acceleration.

The time cost for the online prediction is shown as TABLE I.

For longitudinal average displacement error, results show that Structural-LSTM model produces the best prediction while Seq2seq-1 model produces the highest error, no matter in case of lane changing or lane keeping as Fig. 6 shown. In addition, there is no significant difference the longitudinal error between lane changing and lane keeping for all the three models.

For lateral average displacement error, results show that Structural-LSTM model produces the best prediction while Seq2seq-1 model produces the highest error, no matter in case of lane changing or lane keeping as Fig. 6 shown. In addition, the lateral position error is significantly lower in case of lane keeping rather than lane changing for all the three models.

For longitudinal final displacement error, results show that the Single LSTM model produces the highest error in each prediction horizon. The Dual LSTMs model and the Linear

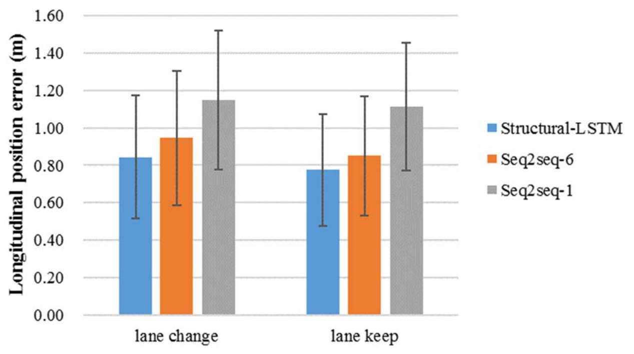
TABLE III
FINAL DISPLACEMENT ERROR

Model	**Prediction Horizon**				
	1s	*2s*	*3s*	*4s*	*5s*
Structural-LSTM	0.57	**1.05**	**1.35**	**1.65**	**1.93**
Seq2seq-6	0.59	1.08	1.46	1.85	2.23
Seq2seq-1	0.60	1.10	1.71	2.51	3.48
Dual LSTMs	**0.47**	1.39	2.57	4.04	5.77
Single LSTM	0.71	1.98	3.75	5.96	9.00
Linear model	0.50	1.43	2.63	4.1	5.8

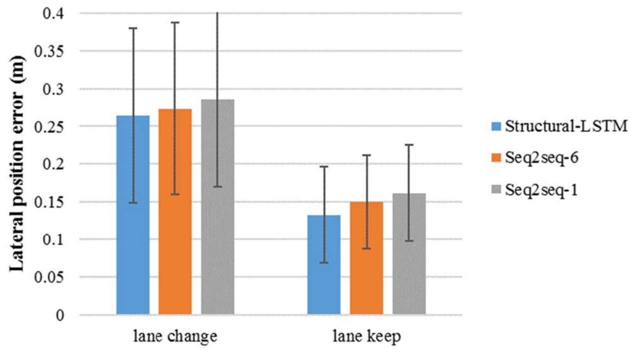
(a) Longitudinal position error (m)

Model	**Prediction Horizon**				
	1s	*2s*	*3s*	*4s*	*5s*
Structural-LSTM	0.19	0.28	**0.28**	**0.29**	**0.31**
Seq2seq-6	0.19	0.28	0.29	0.32	0.37
Seq2seq-1	0.18	0.28	0.30	0.34	0.40
Dual LSTMs	0.15	0.26	0.38	0.45	0.49
Single LSTM	**0.11**	**0.25**	0.33	0.40	0.47
Linear model	0.16	0.38	0.60	0.82	1.05

(b) Lateral position error (m)



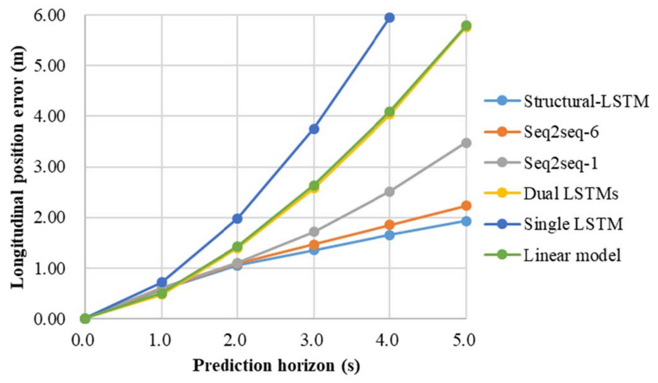
(a) Longitudinal position error



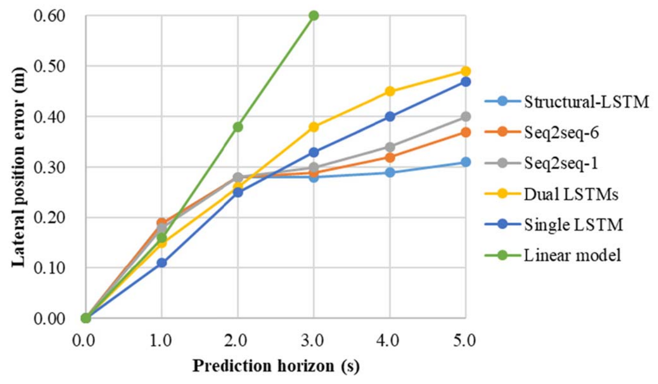
(b) Lateral position error

Fig. 6. Average displacement error of trajectory prediction over 5s.

model produce better prediction, especially the Dual LSTMs model producing the lowest error in prediction horizon of 1.0s.



(a) Longitudinal position error



(b) Lateral position error

Fig. 7. Final displacement error of trajectory prediction.

In case of prediction horizon longer than 1.0s, the Seq2seq-1, Seq2seq-6 and Structural-LSTM models produce much lower error. We note that the Seq2seq-6 model produces better prediction than the Seq2seq-1 model by inputting the features of the other five SVs. We also note that the Structural-LSTM model produces better prediction than the Seq2seq-6 model by hierarchically learning the interaction between the target SV and the other SVs.

For lateral final displacement error, results show that the Linear model produces the highest error in prediction horizon longer than 1.0s. The Single LSTM model and Dual LSTMs produce better prediction, especially the Single LSTM model producing the lowest error in prediction horizon shorter than 2.0s. In case of prediction horizon longer than 1.0s, the Seq2seq-1, Seq2seq-6 and Structural-LSTM models produce lower error. Similar as the longitudinal position predicting, the more input features and the inter-vehicle interactions contribute to a higher accuracy of lateral position predicting in long-term horizon.

The improvement of using interaction is shown in Fig. 7. The orange line is Seq2seq-6 method which simply inputs trajectories of six SVs and outputs trajectories of six SVs in one LSTM cell. The blue line is our proposed method which models interaction using connected six LSTM cells. It shows that the proposed method reduced error over 13% and 16% for longitudinal and lateral prediction, respectively.

Trajectory predictions in three different scenarios are presented in Fig. 8. The trajectories of center SV, front SV,

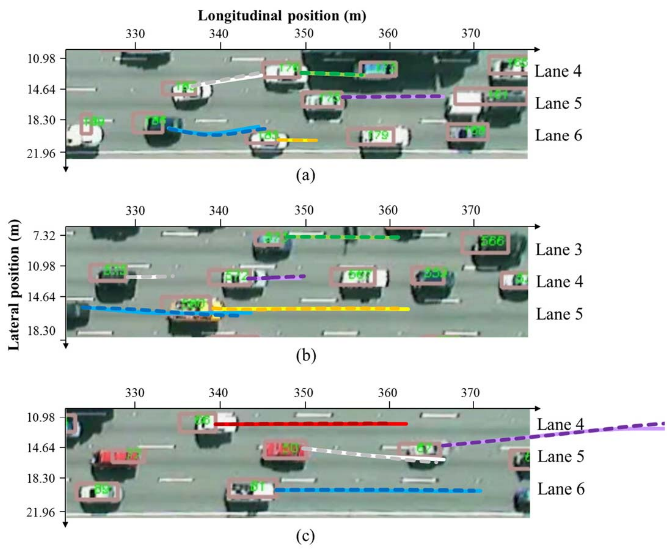


Fig. 8. Examples of typical trajectory prediction under multi-interactive scenario using Structural-LSTM model: (a) left lane changing and following left lane changing; (b) following lane keeping and braking; (c) left lane changing and following right lane changing (Center SV in white line, front SV in purple line, left front SV in green line, left rear SV in red line, right front SV in yellow line, right rear SV in blue line; true trajectoriy in solid line, predicted trajectoriy in dashed line).

left front SV, left rear SV, right front SV and right rear SV are represent by white lines, purple lines, green lines, red lines, yellow lines and blue lines, respectively, with predicted trajectories in dashed lines and ground truth in solid lines. Note that all the six SVs here are the target SV, i.e., the trajectories of the six SVs are predicted from six different test samples, respectively, where each of them is the target SV, instead of being predicted from a single sample where only the center SV is the target SV. In Fig. 8(a), a left lane changing trajectory of the center SV (white), a following left lane changing trajectory of the right rear SV (blue), lane keeping trajectories of the left front SV (green), front SV (purple) and the right front SV (yellow) are successfully predicted. The right rear SV (blue) is trying for a right lane changing from Lane 5 to Lane 6 at first but soon moves back to Lane 5. This is because the low-speed right front SV (yellow) leaves rare safety area for the right rear SV (blue) to stay in Lane 6 while the center SV (white) changing from Lane 5 to Lane 4 provides enough safety area for the right rear SV (blue) to change back to Lane 5. By understanding such interaction, the proposed Structural-LSTM model can predict the overtake-like trajectory of the right rear SV (blue). In Fig. 8(b), lane keeping trajectories of the five SVs are successfully predicted. By understanding the effect of the decelerating SVs in front and the constraints of SVs in the adjacent lanes, the proposed Structural-LSTM model can predict low-speed trajectories of the center SV (white) and the front SV (purple). In addition, a high-speed trajectory of the right front SV (yellow) is also predicted as no SV is blocking in front of it. In Fig. 8(c), a left lane changing trajectory of the front SV (purple), a following right lane changing trajectory of the center SV (white), lane keeping trajectories of the left rear SV (red) and the right rear SV (blue) are successfully predicted. The center SV (white) is changing to Lane 5 because the front SV (purple) changing from Lane 5 to Lane 4 provides enough safety area for the center SV (white) to change to Lane 5. The proposed Structural-LSTM model understands this interaction and predict the trajectories.

V. Conclusion

In this paper, a multi-sequence learning network called structural-LSTM is proposed to learn interaction patterns among surrounding vehicles and thus predicting their long-term (5s in the future) trajectories. The evaluation of this method on NGSIM dataset shows better prediction accuracy indicated by smaller RMS error for both longitudinal and lateral position prediction over different time horizons. Compared to the state-of-art studies, the proposed network has the following advantages: (1) adaptability to various road geometry due to the special representation of lateral deviation from the center of the target lane; (2) consider multiple SVs as a whole which reflects interaction among these SVs; and (3) make use of the interactions not only in understanding historic trajectories (in encoder) but also in predicting future trajectories (in decoder).

This study provides a promising way to handle motion prediction of SVs for autonomous driving systems. Due to the accessibility of data, we use only lane-keeping and lane changing maneuvers for evaluation. But the basic procedure is the same for other scenarios. The preliminary research opens various perspectives for future research: (1) to generalize the proposed model in different scenarios, e.g. intersections and unstructured roads; (2) to address the motion prediction as a stochastic problem which requires distributions and confidence intervals instead of just a single value; (3) to capture spatial information using visual- and map-based cues in a CNN model.

References

[1] J. Chen, C. Tang, L. Xin, S. E. Li, and M. Tomizuka, "Continuous decision making for on-road autonomous driving under uncertain and interactive environments," in *Proc. IEEE 29th Intell. Veh. Symp.*, Jun. 2018, pp. 1651–1658.

[2] R. Li, Y. Li, S. E. Li, E. Burdet, and B. Cheng, "Driver-automation indirect shared control of highly automated vehicles with intention-aware authority transition," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2017, pp. 26–32.

[3] S. Ammoun and F. Nashashibi, "Real time trajectory prediction for collision risk estimation between vehicles," in *Proc. IEEE 5th Int. Conf. Intell. Comput. Commun. Process.*, Aug. 2009, pp. 417–422.

[4] N. Kaempchen, K. Weiss, M. Schaefer, and K. C. Dietmayer, "IMM object tracking for high dynamic driving maneuvers," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2004, pp. 825–830.

[5] Y. Liao, G. Li, S. E. Li, B. Cheng, and P. A. Green, "Understanding driver response patterns to increased mental workload," *IEEE Access*, vol. 6, pp. 35890–35900, Mar. 2018.

[6] G. Li, S. E. Li, B. Cheng, and P. Green, "Estimation of driving style in naturalistic highway traffic using maneuver transition probabilities," *Transp. Res. C, Emerg. Technol.*, vol. 74, pp. 113–125, Jan. 2017.

[7] N. Deo and M. M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based LSTMs," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2018, pp. 1179–1184.

[8] L. Xin, P. Wang, C. Chan, J. Chen, S. E. Li, and B. Cheng, "Intention-aware long horizon trajectory prediction of surrounding vehicles using dual LSTM networks," in *Proc. IEEE 21st Int. Conf. Intell. Transp. Syst.*, Nov. 2018, pp. 1441–1446.

[9] 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.*, Oct. 2017, pp. 399–404.

[10] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, "Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2018, pp. 1672–1678.

[11] F. Altché 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.

[12] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1468–1476.

[13] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-RNN: Deep learning on spatio-temporal graphs," in *Proc. IEEE CVPR*, Jun. 2016, pp. 5308–5317.

[14] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH J.*, vol. 1, no. 1, pp. 1–14, 2014.

[15] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using Monte Carlo sampling," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 137–147, Mar. 2008.

[16] 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.

[17] Q. Tran and J. Firl, "Online maneuver recognition and multimodal trajectory rediction for intersection assistance using non-parametric regression," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2014, pp. 918–923.

[18] J. Wiest, U. Höffken, M. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with Gaussian mixture models," in *Proc. IEEE Intell. Vehicles Symp.*, Oct. 2012, pp. 141–146.

[19] H. Xu, Y. Zhou, W. Lin, and H. Zha, "Unsupervised trajectory clustering via adaptive multi-kernel-based shrinkage," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4328–4336.

[20] J. Joseph, F. Doshi-Velez, A. S. Huang, and N. Roy, "A Bayesian nonparametric approach to modeling motion patterns," *Auton. Robot.*, vol. 31, no. 4, p. 383, Nov. 2011.

[21] 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.

[22] S. Atev, G. Miller, and N. P. Papanikolopoulos, "Clustering of vehicle trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 647–657, Sep. 2010.

[23] M. G. Ortiz, J. Fritsch, F. Kummert, and A. Gepperth, "Behavior prediction at multiple time-scales in inner-city scenarios," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2011, pp. 1068–1073.

[24] A. Zyner, S. Worrall, J. Ward, and E. Nebot, "Long short term memory for driver intent prediction," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2017, pp. 1484–1489.

[25] S. Klingelschmitt, M. Platho, H. Gross, V. Willert, and J. Eggert, "Combining behavior and situation information for reliably estimating multiple intentions," in *Proc. IEEE Intell. Veh. Symp.*, Jun. 2014, pp. 388–393.

[26] B. Morris, A. Doshi, and M. Trivedi, "Lane change intent prediction for driver assistance: On-road design and evaluation," in *Proc. 4th IEEE Intell. Vehicles Symp.*, Baden-Baden, Germany, Jun. 2011, pp. 895–901.

[27] G. S. Aoude, B. D. Luders, K. K. H. Lee, D. S. Levine, and J. P. How, "Threat assessment design for driver assistance system at intersections," in *Proc. IEEE 13th Int. Conf. Intell. Transp. Syst.*, pp. 1855–1862, Sep. 2010.

[28] 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.

[29] 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.

[30] Y. Zhou, W. Lin, H. Su, J. Wu, J. Wang, and Y. Zhou, "Representing and recognizing motion trajectories: A tube and droplet approach," in *Proc. 22nd ACM Int. Conf. Multimedia*, Nov. 2014, pp. 1077–1080.

[31] W. Lin *et al.*, "A tube-and-droplet-based approach for representing and analyzing motion trajectories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1489–1503, Aug. 2016.

[32] T. Streubel and K. H. Hoffmann, "Prediction of driver intended path at intersections," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2014, pp. 134–139.

[33] A. Tamke, T. Dang, and G. Breuel, "A flexible method for criticality assessment in driver assistance systems," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2011, pp. 697–702.

[34] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, Doha, Qatar, Oct. 2014, pp. 1724–1734.

[35] G. Agamennoni, J. I. Nieto, and E. M. Nebot, "Estimation of multi-vehicle dynamics by considering contextual information," *IEEE Trans. Robot.*, vol. 28, no. 4, pp. 855–870, Aug. 2012.

[36] C. Ju, Z. Wang, and X. Zhang, "Socially aware Kalman neural networks for trajectory prediction," 2018, *arXiv:1809.05408*. [Online]. Available: https://arxiv.org/abs/1809.05408

[37] Y. Ma, X. Zhu, S. Zhang, R. Yang, W. Wang, and D. Manocha, "TrafficPredict: Trajectory prediction for heterogeneous traffic-agents," 2018, *arXiv:1811.02146*. [Online]. Available: https://arxiv.org/abs/1811.02146

[38] U.S. Department of Transportation. (2008). *NGSIM: Next Generation Simulation*. Accessed: Jun. 6, 2017. [Online]. Available: http://www.ngsim.fhwa.dot.gov

[39] J. Morton, T. A. Wheeler, and M. J. Kochenderfer, "Analysis of recurrent neural networks for probabilistic modeling of driver behavior," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 5, pp. 1289–1298, May 2017.

[40] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," Jul. 2016, *arXiv:1607.06450*. [Online]. Available: https://arxiv.org/abs/1607.06450

[41] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Proc. NIPS*, Dec. 2017, pp. 971–980.

[42] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent.*, May 2015, p. 13.

[43] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX conf. OSDI*, Nov. 2016, pp. 265–283.

Lian Hou received the B.S. degree in vehicle engineering from Tsinghua University, Beijing, China, in 2016, where he is currently pursuing the master's degree in mechanical engineering. He is currently with the State Key Laboratory of Automotive Safety and Energy, School of Vehicle and Mobility, Tsinghua University. His research interests include driving behavior analysis and vehicle trajectory prediction.

Long Xin received the B.S. degree in mathematics and physics from Tsinghua University, Beijing, China, in 2013, where he is currently pursuing the Ph.D. degree with the Department of Automotive Engineering. He was a Visiting Student Researcher with the Department of Mechanical Engineering and California PATH, University of California at Berkeley, USA. He is currently with the State Key Laboratory of Automotive Safety and Energy, School of Vehicle and Mobility, Tsinghua University. His active research interests include autonomous driving, intelligent transportation, and machine learning for intelligent vehicles.

Shengbo Eben Li (SM'16) received the M.S. and Ph.D. degrees from Tsinghua University in 2006 and 2009, respectively. He was with Stanford University, University of Michigan, and University of California at Berkeley. He is currently a tenured Associate Professor with Tsinghua University. He is the author of over 100 journal/conference articles and the co-inventor of over 20 Chinese patents. His active research interests include intelligent vehicles and driver assistance, reinforcement learning and distributed control, and optimal control and estimation.

He was a recipient of the Best Paper Award in 2014 IEEE ITS Symposium, the Best Paper Award in 14th ITS Asia Pacific Forum, the National Award for Technological Invention in China in 2013, the Excellent Young Scholar of NSF China in 2016, and the Young Professorship of Changjiang Scholar Program in 2016. He serves as an Associated Editor for IEEE ITSM and IEEE TRANSACTIONS ON ITS.

Bo Cheng received the B.S. and M.S. degrees in automotive engineering from Tsinghua University, Beijing, China, in 1985 and 1988, respectively, and the Ph.D. degree in mechanical engineering from The University of Tokyo, Tokyo, Japan, in 1998. He is currently a Professor with the School of Vehicle and Mobility, Tsinghua University, where he is the Dean of the Suzhou Automotive Research Institute. He is the author of more than 100 peer-reviewed journal/conference articles and the co-inventor of 40 patents. His active research interests include autonomous vehicles, driver-assistance systems, active safety, and vehicular ergonomics, among others. He is also the Chairman of the Academic Board of SAE-Beijing, a member of the Council of the Chinese Ergonomics Society, and a Committee Member of National 863 Plan, among others.

Wenjun Wang received the M.S. and Ph.D. degrees from The University of Tokyo, Tokyo, Japan, in 2005 and 2008, respectively. From 2008 to 2011, he was a Visiting Researcher with the Toyota Central Research and Development Labs., Inc. He is currently an Associate Professor with the School of Vehicle and Mobility, Tsinghua University, Beijing, China. His active research interests include vehicle system dynamics and control and driver behavior.