RECUP Net: RECUrsive Prediction Network for Surrounding Vehicle Trajectory Prediction with Future Trajectory Feedback

Sanmin Kim[1], Dongsuk Kum[1*] and Jun won Choi[2], *Member, IEEE*

Abstract— **In order to predict the behavior of human drivers accurately, the autonomous vehicle should be able to understand the reasoning and decision process of motion generation of human drivers. However, most of the conventional prediction methods overlook this and focus on improving the prediction results using the given data, the historical information. In contrast, human drivers not only depend on the historical motion but also consider future predictions when handling interactions with other vehicles. In this paper, we propose a novel recursive RNN encoder-decoder prediction model that takes the initial future prediction results as inputs of second prediction computation. This feedback mechanism can be interpreted as a *network sharing*, which allows the model to refine or correct the predicted results iteratively. We use two encoders to analyze both of the historical information and future information, and the attention mechanism is employed to interpret interaction. Our experimental results with the NGSIM dataset demonstrate the recursive structure enhances prediction results effectively compare to the baselines based on the ablation study and state-of-the-art methods. Furthermore, we observe that the results improve successively as the model iterates.**

I. Introduction

Autonomous vehicles have to perceive other drivers' intention or planning strategy before it happens to react to the danger in the future proactively. Otherwise, it can cause unsafe and unexpected motion, such as sudden braking. However, predicting the future motion of other vehicles is a challenging task due to the complicated characteristics of vehicle motion. The high complexity of vehicle trajectories, imperfect sensor data, and interaction between traffic participants make the prediction of the motion of surrounding vehicles difficult.

In order to tackle these problems, following the success of deep learning, various deep learning-based prediction models have been employed. In particular, the Recurrent Neural Network (RNN) has recently been used to analyze the temporal structure and mutual interactions between the multiple trajectories of the surrounding vehicles. [1]-[4] use RNN for the pedestrian trajectory prediction and [5]-[10] use RNN for the vehicle motion prediction. Even though RNN-based prediction models have achieved significant improvement over the previous methods, their performance is limited in predicting the reasoning and intention of human drivers. It might be attributed to the fact that they only focus on past information and overlook the importance of future interaction. As depicted in Fig. 1, when human drivers plan their future path, they empirically predict the motion of their surrounding vehicles in advance. They then make their decision on which way to go taking into account both past and future information of their surrounding vehicles. In here, what autonomous vehicles should predict are these behaviors of human drivers. Therefore, autonomous vehicles also must able to use future information to predict the future motion of human drivers. However, to the best knowledge of authors, all of the previous methods do not consider future interaction explicitly in their prediction models. Additionally, most of the previous methods are implemented with the first-person view of a target vehicle. However, this perspective of view needs the information of surrounding vehicles around the target vehicle, which is infeasible from the sensors of the ego vehicle.

In this paper, we address this issue by proposing a novel trajectory prediction architecture, which can consider not only past information but also future information that roles as an empirical prediction of human drivers. In particular, we introduce a "Recursive Structure" of an encoder-decoder architecture, which reuse the output of the decoder into the input of the encoder and makes the model to work iteratively. In this way, the encoder takes both future information (the output of the decoder) and past information together to generate accurate future prediction results. The main contributions of our work are as follows:

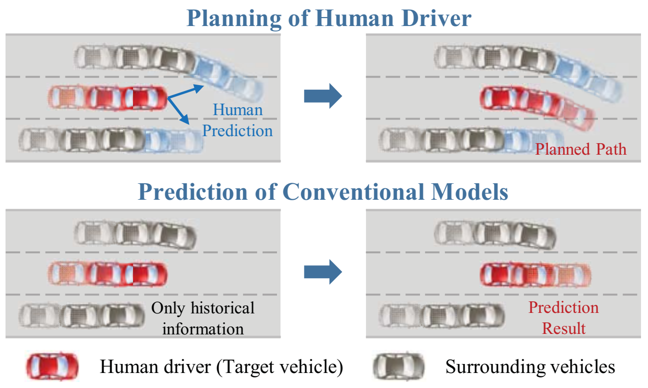
Planning of Human Driver
Human Prediction
Planned Path
Prediction of Conventional Models
Only historical information
Prediction Result
Human driver (Target vehicle)
Surrounding vehicles

Figure 1. Illustration for the prediction of conventional prediction models and planning of human drivers. **(Top)** Human drivers predict motion of other vehicles empirically and then combine the historical information and prediction results for the planning. Autonomous vehicles should predict this kind of behavior of human drivers. **(Bottom)** Conventional prediction models use only historical information to predict future motion.

This research was supported and the KAIST-KU Joint Research Center, KAIST, Korea and the Technology Innovation Program (No. 10083646) funded by the Ministry of Trade, Industry & Energy (MOTIE, Korea)

* corresponding author

[1] S. Kim and D. Kum are with the Cho Chun Shik Graduate School for Green Transportation, KAIST, Daejeon 34051, Republic of Korea. E-mail: {sm0312, dskum}@ kaist.ac.kr.

[2] J. Choi is with the Department of Electrical Engineering, Hanyang University, Seoul Republic of Korea. E-mail: {junwchoi}@ hanyang.ac.kr

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on October 21,2021 at 08:19:34 UTC from IEEE Xplore. Restrictions apply.

- We propose a novel recursive RNN encoder-decoder prediction model (RECUP Net), which improves the prediction accuracy successively by recycling the output of the prediction results.
- The prediction is implemented with the first-person perspective of an autonomous vehicle (ego vehicle) instead of the perspective of a target vehicle. This perspective of view is directly applicable to real-world technology.

II. Related Work

A. *Independent prediction*

Independent prediction models solve prediction problems in the simplest way. They focus on a single target vehicle and only use the information of that vehicle. [11], [12], [13] use a Constant Yaw Rate and Acceleration model, which assumes vehicles keep current motion. [14] and [15] use the Interacting Multiple Model (IMM) to predict the target lane of the vehicle. In other ways, [16] use Variational Gaussian Mixture Model to take into account uncertainty of the future position and [17] use Gaussian Process Regression and Unscented Kalman Filter for non-linearity of trajectory. However, independent prediction models are not appropriate in the multi-vehicle situation because it does not consider the influence of other vehicles, which is one of the most critical factors that determine future motion.

B. *Interaction-aware prediction*

Interaction-aware prediction models [18] solve problems of independent prediction through modeling interaction between vehicles. [19] and [20] use the Dynamic Bayesian Network for modeling the interaction between agents and [21] use Hidden Markov Model (HMM). In [22], they use Deep Neural Network and Gaussian Mixture Model together. These methods show improved prediction accuracy over independent prediction models. However, they are not enough to model sequential interaction between vehicles. Therefore, a lot of Recurrent Neural Network (RNN) based prediction models have been proposed. [23] propose a combined LSTM and CNN architecture using Occupancy Grid Map (OGM) to consider interaction in a traffic scene implicitly. [1] propose social LSTM for pedestrian motion prediction and introduce social pooling, which effectively models the interaction between pedestrians in crowds. To apply social pooling at the vehicle application, [5] propose a vehicle interaction model with CNN-based social pooling. [24] use past trajectories of multiple agents and scene context together. Even the effectiveness of modeling interaction using RNN is proved, they still have some problems. When modeling the interaction with surrounding vehicles, the different influence of each surrounding vehicle to the target vehicle is ignored.

C. *Attention-based prediction*

The main idea of attention mechanism is giving more weights to more critical inputs. This simple idea showed outstanding performance in the image process [25], [26], and the translation task [27], while the attention mechanism applied in a few prediction models. When interaction-aware models predict the motion of a vehicle, the influence of surrounding vehicles is considered, but they do not care about the degree of influence of each surrounding vehicle. Through the attention mechanism, each surrounding vehicle is treated differently according to its degree of influence. [28] and [3] propose pedestrian prediction models using the attention mechanism. In the vehicle application, [29] apply the attention mechanism for each lane to predict lane-change maneuver. Even though the attention mechanism solves one of the problems of prediction models, there are still some limitations exist. The conventional prediction models only rely on historical information, ignoring future interactions.

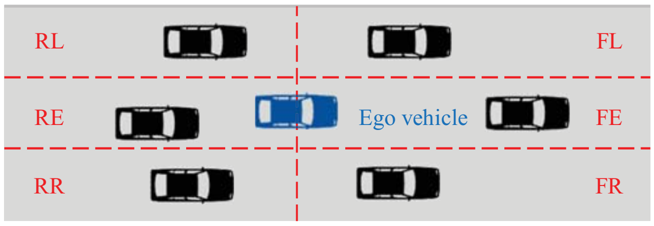

Figure 2. Surrounding vehicles around the ego vehicle. We use maximum 6 vehicles (Rear Left, Front Left, Rear Ego, Front Ego, Rear Right, and Front Right) that within 100 meters (longitudinal distance) from the ego vehicle. The maximum number of vehicles can be changed.

In this paper, we overcome this limitation and propose a novel prediction model. Our model takes into account future interactions in advance through the recursive structure that feedbacks the predicted future trajectories recursively. Additionally, the use of a first-person view of the ego vehicle ensures feasibility compared to others.

III. Problem Definition

We formulate the future trajectory prediction as a regression problem, where the objective is to simultaneously predict future trajectories of multiple vehicles around an ego vehicle (autonomous vehicle) with the prediction time horizon t_p, using historical information of surrounding vehicles and the ego vehicle with observation time horizon time t_h.

The input $\boldsymbol{X} = \{\boldsymbol{X}_1, \cdots, \boldsymbol{X}_N, \boldsymbol{X}_E\}$ is time-serial historical information of surrounding vehicles and the ego vehicle, where N is the number of vehicles and subscript E stands for the ego vehicle. N is not a fixed number but cannot exceed a designated maximum value. In this paper, we defined the maximum value of N as described in Fig. 2. The historical input of the i-th vehicle is defined as $\boldsymbol{X}_i = \{\mathbf{x}_i^{t-t_h}, \cdots, \mathbf{x}_i^{t-1}, \mathbf{x}_i^t\}$. Here, each element of $\boldsymbol{X}_i$ at time t, $\mathbf{x}_i^t$ is a vector in $\mathbb{R}^m$ representing states of the ith vehicle at time t. In this paper, we choose five states ($m = 5$) $\mathbf{x}_i^t = \{x_i^t, y_i^t, v_i^t, d_i^t, l_i^t\}$, where x_i^t and y_i^t are the longitudinal and lateral position, v_i^t is the absolute velocity in the direction of the x-axis, d_i^t is the distance from the lane center line and l_i^t is the lane ID relative to the ego vehicle.

The output of the network is time-serial future trajectories of all surrounding vehicles $\hat{\mathbf{Y}} = \{\hat{\boldsymbol{Y}}_1, \hat{\boldsymbol{Y}}_2, \cdots, \hat{\boldsymbol{Y}}_N\}$. The predicted future trajectory of the i-th vehicle is defined as $\hat{\boldsymbol{Y}}_i = \left\{\hat{\mathbf{y}}_i^{t+1}, \cdots, \hat{\mathbf{y}}_i^{t+t_p-1}, \hat{\mathbf{y}}_i^{t+t_p}\right\}$. Each element of $\hat{\boldsymbol{Y}}_i$ is a vector in $\mathbb{R}^2$ representing the position of i-th vehicle at a future time $\hat{\boldsymbol{y}}_i^t = \{\hat{x}_i^t, \hat{y}_i^t\}$. Where $\hat{x}_i^t$ and $\hat{y}_i^t$ are the predicted longitudinal and lateral position.

Recursive Prediction Network (RECUP Net) operates iteratively to reuse the output of the model as input again, and

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on October 21,2021 at 08:19:34 UTC from IEEE Xplore. Restrictions apply.

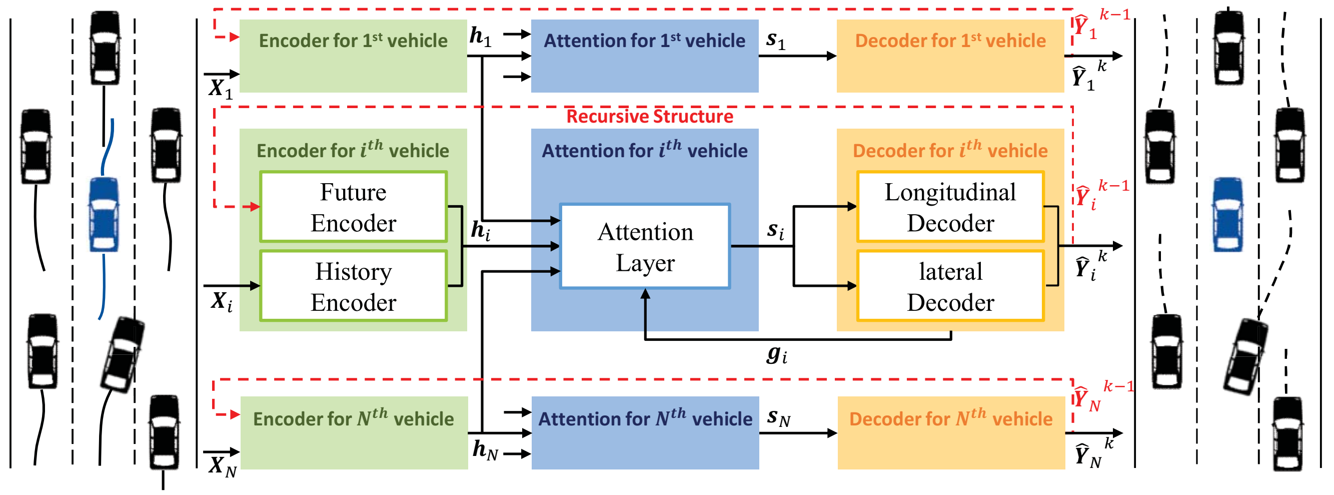

Figure 3. The overall architecture of RECUP Network. Our model predicts trajectories of surrounding vehicles around the ego vehicle simultaneously. Whole network can work iteratively by network-sharing. As the model iterates, it refines the prediction results. It consists of three parts, 1) Motion encoder: encoders takes information of historical motion and roughly predicted trajectories then generate feature vector for each vehicle. 2) Attention layer: this layer takes into account interaction of vehicles using all feature vectors of surrounding vehicles and makes weighted feature vector. 3) Trajectory decoder: decoders generate the future trajectories of surrounding vehicles.

the first iteration and other iteration use different inputs. In the first iteration, there is no output of the model to reuse, so the model only uses the historical information $\hat{\mathbf{Y}}^1 = f(\boldsymbol{X};\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is parameters in the model. However, in the second or higher iterations, the model reuses the output of the previous iteration $\hat{\mathbf{Y}}^k = f(\boldsymbol{X}, \hat{\mathbf{Y}}^{k-1};\boldsymbol{\theta})$, where k means the iteration number.

IV. RECUP NET: RECURSIVE PREDICTION NETWORK

RECUP Net generates the prediction of trajectories of all surrounding vehicles simultaneously. The overall architecture is shown in Fig. 3. We adopt an LSTM encoder-decoder to generate time-serial future trajectories from time-serial information. The same network that shares parameters is used to predict each surrounding vehicle.

A. Recursive Structure

In order to use the historical information and the future information of surrounding vehicles together, we predict future motion first and then use predicted motion as inputs again. Thereby the overall network works iteratively, and

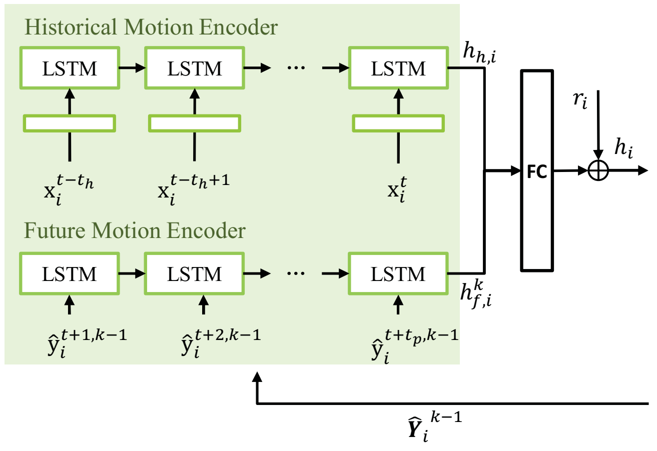

Figure 4. The motion encoder. There are two encoders: the historical motion encoder extracts features of the past motion, and the future motion encoder extracts features of the predicted trajectory from the previous iteration. Then a relative position and existence information is concatenated.

every iteration is a network-sharing process. In the first iteration, the model uses only historical information and gives roughly predicted trajectories of surrounding vehicles $\hat{\mathbf{Y}}^1$. In the second or higher iterations, the model takes $\hat{\mathbf{Y}}^1$($\hat{\mathbf{Y}}^{k-1}$ if the iteration number is k) as the input and then generates the final predicted trajectories $\hat{\mathbf{Y}}^k$. It is done by the recursive structure that feedbacks $\hat{\mathbf{Y}}^{k-1}$ into the future motion encoder of the k-th iteration. In this way, the network can recognize what will happen in the future approximately and then predict future trajectories again with features of both the historical information and future interaction. By making a loop of the network, the model recursively refines and improves prediction results as it repeats.

B. Motion encoder

We use LSTM encoders to extract features from the time-serial historical and future motion of each vehicle. Fig. 4 shows the structure of the encoder. RECUP Net has two encoders: a historical motion encoder and a future motion encoder. The historical motion encoder takes historical inputs $\boldsymbol{X}_i$ and extracts a historical feature vector $\boldsymbol{h}_{h,i}$. The future motion encoder takes the predicted trajectory from the previous iteration $\hat{\boldsymbol{Y}}_i^{k-1}$ and extracts a future feature vector $\boldsymbol{h}_{f,i}^k$ (Eq. (1) and Eq. (2))

$$\boldsymbol{h}_{h,i}^t, \boldsymbol{c}_{h,i}^t = LSTM_{hist}\left(\boldsymbol{W}_{emb}\boldsymbol{X}_i, \boldsymbol{h}_{h,i}^{t-1}, \boldsymbol{c}_{h,i}^{t-1}, ; \boldsymbol{W}_{hist}\right) \quad (1)$$

$$\boldsymbol{h}_{f,i}^{t,k}, \boldsymbol{c}_{f,i}^{t,k} = LSTM_{fut}\left(\hat{\boldsymbol{Y}}_i^{k-1}, \boldsymbol{h}_{f,i}^{t,k-1}, \boldsymbol{c}_{f,i}^{t,k-1}; \boldsymbol{W}_{fut}\right) \quad (2)$$

$\boldsymbol{W}_{emb}$ is the weight matrix of the embedding layer, $\boldsymbol{W}_{hist}$, $\boldsymbol{W}_{fut}$ and $\boldsymbol{c}_{h,i}^t$, $\boldsymbol{c}_{f,i}^{t,k}$ are the weight matrix and cell states of historical and future motion encoders, respectively. Outputs of each encoder have the meaning of compressed information of past motion and future motion of the vehicle.

Then feature vectors $\boldsymbol{h}_{h,i}$ and $\boldsymbol{h}_{f,i}^k$ is concatenated and fed into a fully connected layer. Up to here, the model only deals with the feature of the independent vehicle. To consider interaction with other vehicles, it should use the relative position of vehicles. Therefore, the relative position

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on October 21,2021 at 08:19:34 UTC from IEEE Xplore. Restrictions apply.

of other surrounding vehicles and the indicator of existence are concatenated with the output of the fully connected layer.

$$\boldsymbol{h}_i = [ReLU(\boldsymbol{W}_{fc}[\boldsymbol{h}_{h,i};\boldsymbol{h}_{f,i}]);\boldsymbol{r}_i] \tag{3}$$

$\boldsymbol{W}_{fc}$ is the weight matrix in the fully connected layer and $\boldsymbol{r}_i$ is the vector in $\mathbb{R}^3$, which contains relative position and indicator of the existence of vehicles in that position. For example, if there is no vehicle at i-th position, $\boldsymbol{r}_i = [0,0,0]$, otherwise $\boldsymbol{r}_i = [1, x_e^t - x_i^t, y_e^t - y_i^t]$.

Since there are no predicted trajectories $\hat{\mathbf{Y}}^{k-1}$ at the first iteration, we mask inputs for the future motion encoder as zeros. However, in the case of the ego vehicle, we assume that there is a planned trajectory for the ego vehicle since the ego vehicle would be an autonomous vehicle. Therefore, we fed the ground truth of future trajectory $\boldsymbol{Y}_E$ as the input of the future motion encoder in the first iteration.

C. *Attention layer*

The key idea in the attention mechanism is a selective focus. For example, when a human driver plans a lane-change maneuver, she focuses on the behavior of vehicles in the lane that she plans to go instead of other irrelevant lanes. The attention layer allows the model to imitate this kind of reasoning of human drivers by shifting focus towards certain vehicles more critical. In particular, since RECUP Net uses the first-person perspective of the ego vehicle, vehicles that do not directly interact with the target, e.g., a front vehicle of the front vehicle from the target vehicle, can be included. For this reason, the attention layer is more effective in RECUP Net than in others.

To calculate the different influence of each vehicle in predicting i-th vehicle at time $t+1$, the attention layer takes all feature vectors $[\boldsymbol{h}_1, \cdots, \boldsymbol{h}_N, \boldsymbol{h}_E]$ from encoders of individual vehicles and hidden states of i-th vehicle's decoders at the previous time step $[\boldsymbol{g}_{lat,i}^t; \boldsymbol{g}_{long,i}^t]$. Then calculates the score $e_{j,i}$ that means the degree of influence of j-th vehicle to i-th vehicle (Eq.(4)). The attention weight $\alpha_{j,i}$ is computed by normalizing $e_{j,i}$ with softmax function (Eq. (5)).

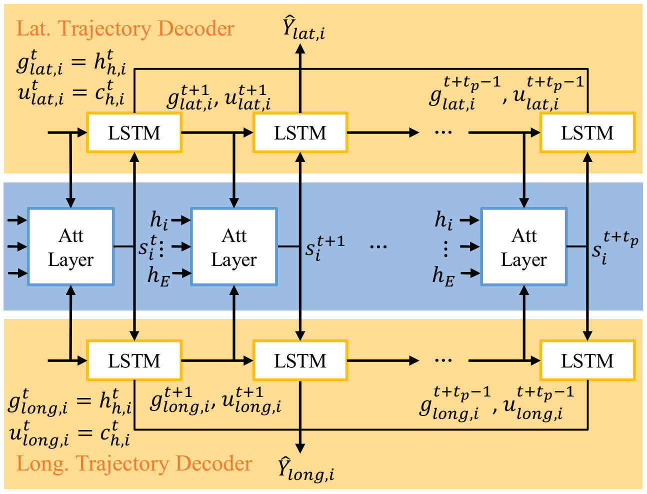
Lat. Trajectory Decoder
$\hat{Y}_{lat,i}$
$g_{lat,i}^t = h_{h,i}^t$
$u_{lat,i}^t = c_{h,i}^t$
$g_{lat,i}^{t+1}, u_{lat,i}^{t+1}$
$g_{lat,i}^{t+t_p-1}, u_{lat,i}^{t+t_p-1}$
LSTM
Att Layer
h_i
s_i^t
h_E
s_i^{t+1}
$s_i^{t+t_p}$
$g_{long,i}^t = h_{h,i}^t$
$u_{long,i}^t = c_{h,i}^t$
$g_{long,i}^{t+1}, u_{long,i}^{t+1}$
$g_{long,i}^{t+t_p-1}, u_{long,i}^{t+t_p-1}$
Long. Trajectory Decoder
$\hat{Y}_{long,i}$

Figure 5. The trajectory decoder. There are two decoders for lateral and longitudinal trajectory. Each decoder uses the same output of attention layer and generates the future trajectory for the target vehicle. In attention layer, degree of influence is calculated and represented as weights $\boldsymbol{\alpha}$. Then weighted feature vector $\mathbf{s}_i$ is generated.

$$e_{j,i} = \boldsymbol{W}_a tanh(\boldsymbol{W}_h\boldsymbol{h}_j + \boldsymbol{W}_g[\boldsymbol{g}_{lat,i}^t; \boldsymbol{g}_{long,i}^t]) \tag{4}$$

$$\alpha_{j,i} = \frac{exp(e_{j,i})}{\sum exp(e_{j,i})} \tag{5}$$

$\boldsymbol{W}_a, \boldsymbol{W}_h$, and $\boldsymbol{W}_g$ are weight matrices. The feature vector $\boldsymbol{h}_j$ is multiplied with $\alpha_{j,i}$ and the weighted feature vector is calculated over all vehicles. The result of the attention layer is the concatenated vector of all $\alpha_{j,i}\boldsymbol{h}_j$ (Eq.(6)).

$$\boldsymbol{s}_i = [\alpha_{1,i}\boldsymbol{h}_1; \alpha_{2,i}\boldsymbol{h}_2; \cdots; \alpha_{N,i}\boldsymbol{h}_N; \alpha_{E,i}\boldsymbol{h}_E] \tag{6}$$

D. *Trajectory decoder*

As shown in Fig. 5, we construct the decoder with LSTMs to interpret the feature vector $\boldsymbol{s}_i$, which is from the attention layer and generate future trajectories. There are two decoders for each vehicle: lateral and longitudinal trajectory decoder. Each decoder only generates lateral or longitudinal prediction independently. We separate decoders to improve prediction accuracy by specializing decoders for each task. Hidden and cell states of both decoders are initialized with the final hidden and cell states from the encoder, in which all features of the target vehicle are contained. By initializing with them, decoders can analyze the dynamic property of the target vehicle and generate future trajectory. Then the output of the attention layer $\mathbf{s}_i$ is fed as the input of decoders.

V. Experimental Evaluation

A. *Dataset*

We use NGSIM I-80 [31] and US101 [32] datasets in this paper. The NGSIM is a publicly available dataset that one of the most extensive naturalistic trajectory datasets and widely used in the prediction models. Trajectories of all the vehicles that pass a fixed area within 45 minutes are annotated at 10Hz. We split the dataset into a train (75%) set and a test (25%) set. The test set divided into half again into a validation set and a test set.

We divide the trajectories into a sequence of 8 seconds, and each sequence consists of 3 seconds of the historical horizon and 5 seconds of the prediction horizon. We downsample each sequence at 5Hz to reduce the complexity of the model. The total number of sequences is 610,304 and among them, 448,512 for training, 80,896 for each validation and test. To prevent overfitting, the train set and test set have different acquisition times.

B. *Implementation details*

All LSTMs in the model are 256 units, and the embedding size is 64. We use ReLU in the encoder and Tanh in the attention layer and the decoders as the activation function. We trained the model with Adam optimizer and a multi-step learning rate. The learning rate for the model is 0.001 for the first 10 epochs and 0.0001 for the rest of 190 epochs. As the loss function of the training, we adopt mean square error (MSE) between the predicted trajectory $\hat{\mathbf{Y}}^k$ and the ground truth trajectory $\mathbf{Y}$. In the loss function, only the final predicted trajectories $\hat{\mathbf{Y}}^k$ are included. Even though the predicted trajectories of the previous iteration $\hat{\mathbf{Y}}^{k-1}$ do not involve in the loss function, they still have the form of trajectories because they are generated from the same network. The model is implemented using PyTorch [30].

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on October 21,2021 at 08:19:34 UTC from IEEE Xplore. Restrictions apply.

TABLE I. RMSE IN METERS AND DECREASE RATE OF ERROR OF TRAJECTORY PREDICTION

Prediction Horizon (s)	Independent V-LSTM (Baseline)	Independent V-LSTM (CS-LSTM)	Interaction V-LSTM	Recursive LSTM	CS-LSTM (Δ)	MATF (Δ)	RECUP Net *k*=2 (Δ)	RECUP Net *k*=3 (Δ)
1	0.64	0.68	0.61	0.71	0.61 (10.29%)	0.66 (-)	0.56 (12.50%)	**0.55 (14.06%)**
2	1.64	1.65	1.70	1.40	1.27 (23.03%)	1.34 (-)	1.27 (22.56%)	**1.23 (25.00%)**
3	2.93	2.91	2.67	2.15	2.09 (28.18%)	2.08 (-)	2.05 (30.03%)	**2.03 (30.03%)**
4	4.49	4.46	3.77	3.07	3.10 (30.49%)	2.97 (-)	2.98 (33.63%)	**2.95 (34.30%)**
5	6.23	6.27	5.14	4.32	4.37 (30.30%)	4.13 (-)	**4.03** (35.31%)	**4.03 (35.31%)**

k means the number of iteration

C. Baselines and Metrics

Baselines are generated based on the ablation study. We also compare RECUP Net with state-of-the-art models.

Independent vanilla LSTM (Baseline): An LSTM encoder- decoder network that uses only historical trajectory information of the target vehicle

Independent vanilla LSTM (CS-LSTM) [1]: The same network architecture with the independent vanilla LSTM in baselines, but produced in [1]

Interaction-aware vanilla LSTM: The baseline model that does not have the attention layer and the recursive structure

Recursive LSTM: The baseline model that does not have the attention layer but have the recursive structure

CS-LSTM [1]: The LSTM encoder-decoder prediction model with convolutional social pooling

MATF (Multi-agent tensor fusion) [21]: The model that applied the convolutional fusion of the LSTM encoder and scene context of multi-agent

We use two performance metrics in this study: Root Mean Square (RMSE) in meters and Error Decrease Rate (EDR) in percentages. RMSE compares the prediction results with the ground truth at each time step $RMSE(t) = \frac{1}{n}\sum_{i=1,2\cdots,n}\{(y_{lat,i}^t - \hat{y}_{lat,i}^t)^2 + (y_{long,i}^t - \hat{y}_{long,i}^t)^2\}$, where n is the number of total vehicles in the test set. It shows the physical distance difference between the prediction and real trajectory intuitively.

However, apple-to-apple comparisons based on RMSE are difficult due to different data split methods, even in the same dataset. To vindicate the evaluation with other methods impartially, the Error Decrease Rate (EDR) is adopted in this paper. EDR is an indicator of how much error is decreased from the simplest baseline of each model. By comparing EDR, we can evaluate prediction performance from different data composition indirectly.

$$EDR(t) = \frac{RMSE_{base}^i(t) - RMSE_{model}^i(t)}{RMSE_{base}^i(t)} \times 100\ \% \qquad (6)$$

where $RMSE_{base}^i$ means RMSE of model i's most simple baseline, and $RMSE_{model}^i$ means RMSE of the model i. Therefore, the simplest baseline form each model should have the same structure, and Independent Vanilla LSTM is used in this paper. EDR is denoted as Δ in Table I.

D. Results

The results in Table I show that the proposed model outperforms both baselines and SOTA methods, such as CS-LSTM and MATF. In particular, we can verify the effect of the recursive structure from RMSE results of baselines. Interaction-aware V-LSTM and Recursive LSTM have the only difference in the existence of the recursive structure, but Recursive LSTM shows better performance (0.82m of RMSE improved at 5 sec). It means that the recursive structure, which makes the model to combine the past and future information, improves the prediction ability. Furthermore, RECUP Net shows better prediction results relative to CS-LSTM and MATF. Moreover, we also verified that the prediction performance is better with the iteration number $k = 3$ than $k = 2$, which means as the model iterates, it can use more accurate future information and improves the prediction results.

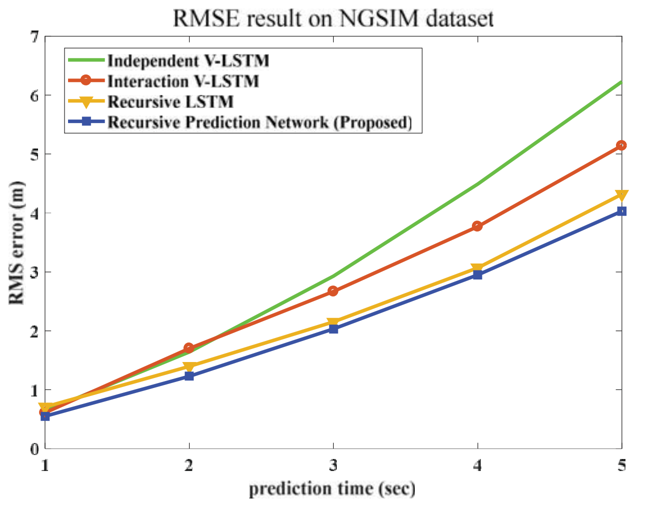

Figure 6. RMS error graph of proposed method and baselines on NGSIM dataset. The graph shows outstanding improvement in RMS error by using the recursive structure.

From the results of EDR, RECUP Net outperforms CS-LSTM (5.01% higher than CS-LSTM at 5sec), which means RECUP Net has more improvement from vanilla LSTM than CS-LSTM. Consequentially, RECUP Net performs better than CS-LSTM in both RMSE and EDR.

Fig. 7. shows the initial-predicted trajectories $\hat{\mathbf{Y}}^1$ and the second-predicted trajectories $\hat{\mathbf{Y}}^2$ simultaneously in some scenes. As shown in Fig. 7(a), one of $\hat{\mathbf{Y}}^2$ is different from $\hat{\mathbf{Y}}^1$ and closer to the ground truth, which means $\hat{\mathbf{Y}}^2$ correct the wrong parts of $\hat{\mathbf{Y}}^1$. Moreover, as shown in Fig. 7(b), $\hat{\mathbf{Y}}^2$ refine trajectories more precisely from $\hat{\mathbf{Y}}^1$. By comparing two trajectories $\hat{\mathbf{Y}}^1$ and $\hat{\mathbf{Y}}^2$, we can find that feeding the predicted trajectories as inputs for future interactions improves prediction accuracy.

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on October 21,2021 at 08:19:34 UTC from IEEE Xplore. Restrictions apply.

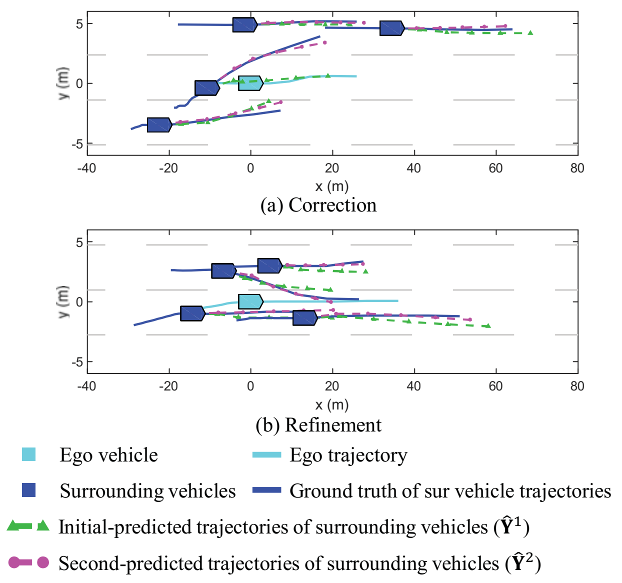

Figure 7. Illustration of initial-predicted trajectories and second-predicted trajectories. (a) shows the refinement effect and (b) shows the correction effect of the final prediction result relative to the initial prediction output.

VI. Conclusion

In this paper, we propose a trajectory prediction model for vehicles on highways. To improve prediction accuracy, we introduced the recursive structure that feeds roughly predicted trajectory into the encoder of the next iteration of the same model recursively. It allows the model to consider the past and future information together as human drivers do. Additionally, our model uses the first-person perspective of the ego vehicle, which is more feasible than other SOTA models. Therefore, our model can be applied to autonomous driving directly. As a result, the proposed model outperforms the reported state-of-the-art on the public dataset of vehicle trajectories.

References

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in IEEE Conference on Computer Vision and Pattern Recognition, Proceedings, 2016, pp. 916–971.

[2] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks," in IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2255–2264.

[3] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, S. H. Rezatofighi, and S. Savarese, "SoPhie: An Attentive GAN for Predicting Paths Compliant to Social and Physical Constraints," in IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 1349–1358.

[4] P. Zhang, W. Ouyang, P. Zhang, J. Xue, and N. Zheng, "SR-LSTM: State Refinement for LSTM towards Pedestrian Trajectory Prediction," in IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 12085–12094.

[5] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2018, pp. 1549–1557.

[6] W. Ding, J. Chen, and S. Shen, "Predicting Vehicle Behaviors Over An Extended Horizon Using Behavior Interaction Network," in IEEE International Conference on Robotics and Automation, 2019, pp. 8634–8640.

[7] F. Altche and A. De La Fortelle, "An LSTM network for highway trajectory prediction," in IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2018, pp. 353–359.

[8] 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 IEEE Intelligent Vehicles Symposium, Proceedings, 2018, pp. 1672–1678.

[9] S. Su, K. Muelling, J. Dolan, P. Palanisamy, and P. Mudalige, "Learning Vehicle Surrounding-aware Lane-changing Behavior from Observed Trajectories," in IEEE Intelligent Vehicles Symposium, 2018, pp. 1412–1417.

[10] L. Xin, P. Wang, C. Y. Chan, J. Chen, S. E. Li, and B. Cheng, "Intention-aware Long Horizon Trajectory Prediction of Surrounding Vehicles using Dual LSTM Networks," in IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2018, pp. 1441–1446.

[11] A. Polychronopoulos, M. Tsogas, A. J. Amditis, and L. Andreone, "Sensor fusion for predicting vehicles' path for collision avoidance systems," IEEE Trans. Intell. Transp. Syst., vol. 8, no. 3, pp. 549–562, 2007.

[12] P. Lytrivis, G. Thomaidis, and A. Amditis, "Cooperative path prediction in vehicular environments," in IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2008, pp. 803–808.

[13] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, "Vehicle trajectory prediction based on motion model and maneuver recognition," in IEEE International Conference on Intelligent Robots and Systems, 2013, pp. 4363–4369.

[14] R. Toledo-Moreo and M. A. Zamora-Izquierdo, "IMM-based lane-change prediction in highways with low-cost GPS/INS," IEEE Trans. Intell. Transp. Syst., vol. 10, no. 1, pp. 180–185, 2009.

[15] J. Kim and D. Kum, "Collision Risk Assessment Algorithm via Lane-Based Probabilistic Motion Prediction of Surrounding Vehicles," IEEE Trans. Intell. Transp. Syst., vol. 19, no. 9, pp. 2965–2976, 2018.

[16] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with Gaussian mixture models," in IEEE Intelligent Vehicles Symposium, Proceedings, 2012, pp. 141–146.

[17] Q. Tran and J. Firl, "Modelling of traffic situations at urban intersections with probabilistic non-parametric regression," in IEEE Intelligent Vehicles Symposium, Proceedings, 2013, pp. 334–339.

[18] 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.

[19] T. Gindele, S. Brechtel, and R. Dillmann, "A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments," in IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2010, pp. 1625–1631.

[20] D. S. González, V. Romero-Cano, J. S. DIbangoye, and C. Laugier, "Interaction-aware driver maneuver inference in highways using realistic driver models," in IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2018, pp. 1–8.

[21] T. Streubel and K. H. Hoffmann, "Prediction of driver intended path at intersections," in IEEE Intelligent Vehicles Symposium, 2014, pp. 134–139.

[22] D. Lenz, F. Diehl, M. T. Le, and A. Knoll, "Deep neural networks for Markovian interactive scene prediction in highway scenarios," in IEEE Intelligent Vehicles Symposium, Proceedings, 2017, pp. 685–692.

[23] H. S. Jeon, D. S. Kum, and W. Y. Jeong, "Traffic Scene Prediction via Deep Learning: Introduction of Multi-Channel Occupancy Grid Map as a Scene Representation," in IEEE Intelligent Vehicles Symposium, Proceedings, 2018, pp. 1496–1501.

[24] T. Zhao et al., "Multi-Agent Tensor Fusion for Contextual Trajectory Prediction," in IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 12126–12134.

[25] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, "Image Captioning with Semantic Attention," in IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4651–4659.

[26] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent Models of Visual Attention," in Advances in Neural Information Processing Systems, 2014, pp. 2204–2212.

[27] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in International Conference on Learning Representations, proceedings, 2015, pp. 1–15.

[28] A. Vemula, K. Muelling, and J. Oh, "Social Attention: Modeling Attention in Human Crowds," in IEEE International Conference on Robotics and Automation, Proceedings, 2018, pp. 4601–4607.

[29] O. Scheel, N. S. Nagaraja, L. Schwarz, N. Navab, and F. Tombari, "Attention-based Lane Change Prediction," in IEEE International Conference on Robotics and Automation, 2019, pp. 8655–8661.

[30] A. Paszke et al., "Automatic differentiation in PyTorch," in Conference on Neural Information Processing Systems, 2017.

[31] J. Colyar and J. Halkias, "US Highway 101 Dataset," 2007.

[32] J. Colyar and J. Halkias, "Interstate 80 Freeway Dataset," 2006.

Authorized licensed use limited to: Southern University of Science and Technology. Downloaded on October 21,2021 at 08:19:34 UTC from IEEE Xplore. Restrictions apply.