2018 IEEE Intelligent Vehicles Symposium (IV)
Changshu, Suzhou, China, June 26-30, 2018

Learning Vehicle Surrounding-aware Lane-changing Behavior from Observed Trajectories

Shuang Su, Katharina Muelling, John Dolan, Praveen Palanisamy, Priyantha Mudalige

***Abstract*—Predicting lane-changing intentions has long been a very active area of research in the autonomous driving community. However, most of the literature has focused on individual vehicles and did not consider both the neighbor information and the accumulated effects of vehicle history trajectories when making the predictions. We propose to apply a surrounding-aware LSTM algorithm for predicting the intention of a vehicle to perform a lane change that takes advantage of both vehicle past trajectories and their neighbor's current states. We trained the model on real-world lane changing data and were able to show in simulation that these two components can lead not only to higher accuracy, but also to earlier lane-changing prediction time, which plays an important role in potentially improving the autonomous vehicle's overall performance.**

***Keywords*—LSTM, lane-change intention.**

I. INTRODUCTION

Lane changing has been regarded as one of the major factors causing traffic accidents [1]. To enable autonomous vehicles to drive on highways, it is important to predict other vehicles' lane-changing intention to prevent potential collisions. There has been a lot of work attempting to model drivers' lane-changing behaviors, which can be divided into two types: rule-based algorithms [2], [3], [4], machine-learning-based algorithms [5], [6], [7], and knowledge-representation algorithms [8], [9], [10].

Rule-based algorithms define a set of rules to model lane changing. The most representative one is the 'gap acceptance model' [2], which assumes that drivers' lane-changing maneuvers are based on the lead and lag gaps in the target lane. This method assumes that the driver tends to make a lane change if the gap has attained a minimum acceptable value. Although straightforward and robust in simple scenarios, such methods requires a lot of parameter fine-tuning, which can be tedious and time-consuming.

Machine-learning-based algorithms create a mathematical model for the problem: given the vehicle-related features as input, and the vehicle's lane-changing intention as output, the methods attempt to infer a mapping function such as to obtain the best prediction results. A large number of classifiers such as logistic regression [5] and SVM classifiers [6] have been adopted to formulate this model.

Knowledge-representation algorithms develop a network model to imitate the human reasoning process. The applications of knowledge-based algorithms in driving scenarios include reinforcement learning [11], curriculum learning [12], and learning through Bayesian networks [8], [9]. However, it may take a long time for the model to generalize the basic rules in an unknown environment during the learning process.

A human-driven vehicle's intention to initiate a lane-changing maneuver is not only based on the vehicle's own state such as heading angle and acceleration, but also on its relationship to neighboring vehicles, such as its distance from the front vehicle. Recently, several references have explored the impact of neighboring traffic on the ego vehicle. Sadigh et al. proposed an algorithm that enables an ego vehicle to model the intentions of an other vehicle in its planning process to purposefully change the other cars' behavior [13]. Furthermore, it requires the knowledge of the other cars' intention in form of a reward function and does not model cooperative behavior that is usually found in humans. Recently, Alahi et al. proposed the idea of a social LSTM, which encodes the pedestrians past trajectory as well as neighbor agents' past trajectory information to predict the future trajectory of the pedestrian in a cooperative manner [14]. The authors suggested the use of LSTMs [15] to model the sequential nature of the problem together with a social pooling layer to model the interactions between the pedestrians. As a result, they were able to model trajectories of pedestrians in a crowded where each of the pedestrian adapted their future trajectories cooperatively. However, the method is not straightforward to apply to the lane-changing problem due to the small number of cooperative interactions in the data.

In this paper, we propose to also use a LSTM network structure to include the vehicles past positions into the prediction algorithm and enable the system to extract the relevant information of the past. To model the effect of other cars on the decision process while keeping the problem tractable, we include information about the neighboring cars into the input features of the network.

To get a full understanding of naturalistic driving behaviors, a large number of driving and lane-changing trajectories are required for the learning process, which is why we chose the NGSIM data set [16] to train and verify our algorithms. We also adopted a julia-based NGSIM [17] platform to extract input features for the network and visualize the traffic

This work was supported by GM funding.

S. Su is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, 15213 USA, E-mail: shuangs@andrew.cmu.edu.

K. Muelling and J. Dolan are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 15213 USA, E-mail: kmuelling@nrec.ri.cmu.edu, jmd@cs.cmu.edu.

P. Palanisamy and P. Mudalige are with General Motors Company.

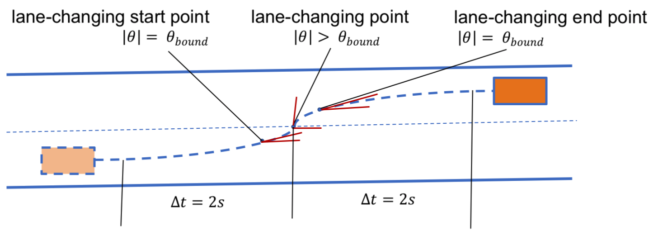

lane-changing start point
$|\theta| = \theta_{bound}$
lane-changing point
$|\theta| > \theta_{bound}$
lane-changing end point
$|\theta| = \theta_{bound}$
$\Delta t = 2s$
$\Delta t = 2s$

Fig. 1. The start point, lane-changing point, and end point of a lane-changing trajectory.

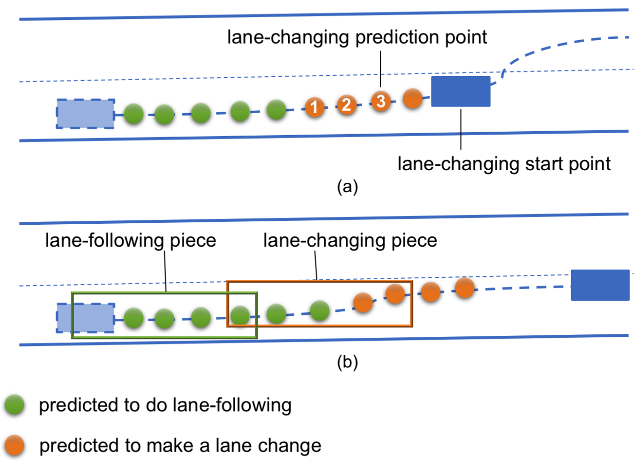

lane-changing prediction point
1
2
3
lane-changing start point
(a)
lane-following piece
lane-changing piece
(b)
predicted to do lane-following
predicted to make a lane change

Fig. 2. (a) A lane-changing prediction point is settled if a vehicle is predicted to make a lane change for 3 consecutive time steps. The lane-changing prediction time is defined to be the time gap between the lane-changing point and the lane-changing prediction point. (b) n continuous time steps were packed into one trajectory piece. If the n th time step of a trajectory piece was a lane-following time step, then the piece was a lane-following piece, otherwise it was labeled as a lane-changing piece.

scenarios.

Section II describes the procedures of extracting and pre-processing data from the NGSIM data set. In Section III, we introduce the input features and explain the specific network structure in detail. We then illustrate and compare our results to other methods in Section IV. Section V summarizes the results of our work and proposes potential future work.

II. Data extraction and process

The open source Federal Highway Administration's Next Generation Simulation (NGSIM) data set [16], which has been adopted in numerous previous studies [5], [6], was picked to extract vehicle trajectories and build the lane-changing prediction model. At 0.1-second intervals, the data set recorded the location, speed, acceleration, and headway information for each vehicle on U.S. Highway 101 [18] and the Interstate 80 (I-80) Freeway [19]. Both locations contain 45 minutes. of vehicle trajectory data. Highway 101 is 640 meters long with five main lanes and a sixth auxiliary lane, while I-80 is about 500 meters in length, with 6 main lanes.

We extracted 6 sequences of vehicle trajectory data, 10 minutes each, from NGSIM. We removed the first 5 minutes from each 15-min sequence to ensure there are enough number of vehicles involved in each frame. For each sequence, the first 2 minutes were defined as the test set, and the remaining 8 minutes as the training set. Since the data were recorded at 10 frames per second, we could obtain 1200 test time steps and 4800 training time steps in total.

A vehicle is labeled as 'intend to change lane to the left', 'intend to do lane following', or 'intend to change lane to the right' at each time step. The way we labeled the vehicle status is as follows.

As depicted in Fig. 1, we first gathered all of the lane-changing points, i.e., the points where the vehicle gravity point crossed the dashed line dividing the lanes, for each vehicle. If a vehicle was on a lane-changing point at time step t, we checked its trajectories in $[t-\delta t, t+\delta t] (\delta t = 2s)$, and calculated its heading orientation θ during that time period. We then marked the starting point and ending point of this lane-changing trajectory when θ has reached a bounding value θ_{bound}: $|\theta| = \theta_{bound}$.

Fig. 2 (b) depicts the way we labeled the trajectory pieces. n consecutive time steps were packed into one trajectory piece for each vehicle. If the n th time step of a trajectory piece was a lane-changing time step, then the piece was a lane-changing piece, otherwise it was labeled as a lane-following piece. In this paper, we set n to be 6, 9, and 12 to determine the impact of length of the history trajectories on the final results.

We could then get around $60,000$ lane changing pieces plus $400,0000$ car following pieces in total for training. This clearly involves a data-imbalance problem, where there are far more lane following pieces than lane-changing pieces for training, which will result in over-fitting in the training process. To deal with this problem, we randomly selected the same number of pieces, N, from the lane-changing-left pool, the lane-following pool and the lane-changing-right pool, mixing them together for the training data set. To make the most use of the data, N is set to be the number of pieces in the lane-changing-right pool $(30,000)$.

Each vehicle's lane-changing intention was then predicted at each time step given its previous $(n-1)$-time-step history trajectories and neighbor information in the test set. The lane-changing prediction time was also calculated after filtering the results. Specifically, a lane-changing prediction point is settled if a vehicle is predicted to make a lane change for 3 continuous time steps, and the lane-changing prediction time is defined to be the time gap between the lane-changing point and the lane-changing prediction point, as depicted in Fig. 2 (a).

III. Methodology

In this paper, we attempt to predict whether a car changes lanes and in which lane it will merge. We use a LSTM to enable the agent to reason over the vehicles history trajectory information. However, since the human decision behavior will depend also on its surrounding vehicles, we include also the vehicles neighbors information as input to the network.

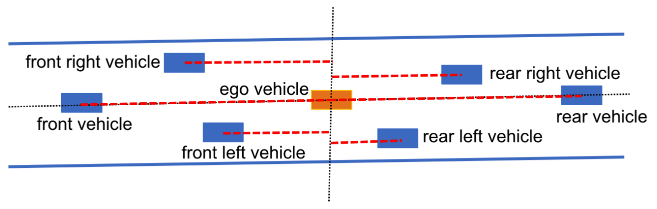


Fig. 3. Neighbor information collection. We first divided the neighbor space into four parts based on the ego vehicle's orientation and center position, and defined the corresponding neighbor vehicles based on their relative positions to the ego vehicle. We then collected the longitudinal distances between these neighbor vehicles and the ego vehicle to be the neighbor features. If there are no neighbors in the corresponding position, we defined the distances to be 500 m to infer infinite distances.

In the following, we will describe the input features that are provided to the prediction algorithm and then shortly describe the network architecture used here.

A. Input features

We use two types of input features for the prediction algorithm: (a) the vehicle's own information and (b) the vehicle's neighbor information. The vehicle's own information include:

1) vehicle acceleration
2) vehicle steering angle with respect to the road
3) the global lateral vehicle position with respect to the lane
4) the global longitudinal vehicle position with respect to the lane

The vehicle's neighbor information (see Fig. 3, "ego vehicle" here refers to the vehicle whose lane-changing intention we are estimating) are provided through the following features:

1) the existence of left lane (1 if it exists, 0 if not)
2) the existence of right lane (1 if exists, 0 if not)
3) the longitudinal distance between ego vehicle and left-front vehicle
4) the longitudinal distance between ego vehicle and front vehicle
5) the longitudinal distance between ego vehicle and right-front vehicle
6) the longitudinal distance between ego vehicle and left-rear vehicle
7) the longitudinal distance between ego vehicle and rear vehicle
8) the longitudinal distance between ego vehicle and right-rear vehicle

B. Network structure

We adopted the LSTM network structure, as depicted in Fig. 4, to deal with this lane-changing intention prediction problem. The embedding dimension chosen for the vehicle's own features as well as its neighbor features was 64, and the hidden dimension for the LSTM network was 128. We selected the learning rate to be 0.000125, using soft-max cross entropy loss as the training loss: $loss = -\sum_{i=1} y_i^{'} log(y_i)$, where $y_i^{'}$ is the real label for the i th lane-changing intention($y_i^{'} = 1$, the intention exists, $y_i^{'} = 0$, the intention does not exist. $i \in \{1,2,3\}$, $y_1^{'}$ is the lane-changing left intention, $y_2^{'}$ is the lane-following intention, and $y_3^{'}$ is the lane-changing right intention). y_i is the predicted output probability from the model for the i th lane-changing intention after going through a soft-max layer.

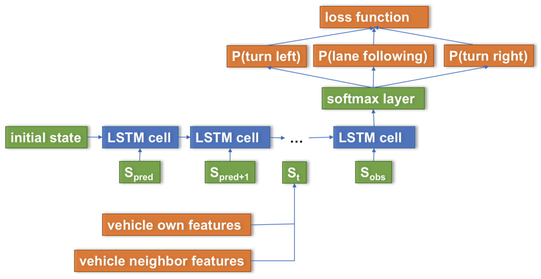


Fig. 4. LSTM network structure for lane-changing intention prediction.

	Real Predict	Left	Following	Right
SA-LSTM	Left	87.40%	12.34%	0.26%
	Following	7.47%	85.33%	7.20%
	Right	2.94%	11.22%	85.84%
Feedforward Neural Network	Left	84.6%	15.40%	0%
	Following	2.61%	83.78%	13.61%
	Right	2.44%	12.91%	79.65%
Logistic Regression	Left	64.91%	35.03%	0.06%
	Following	9.88%	82.87%	7.25%
	Right	0.05%	36.30%	63.65%

TABLE I. LANE-CHANGING PREDICTION ACCURACY COMPARISON

IV. RESULTS

A. Comparison with other network structures

We obtained and compared our results with feedforward neural network, logistic regression and LSTM without neighbor feature inputs to show the advantages of adding history trajectories and environmental factors.

Table I and Fig. 5 show the classification accuracy rate calculated by our algorithm, feedforward neural network, and logistic regression. Our approach to which we will refer as Surrounding Aware (SA)-LSTM and which is based on the benefits of history trajectory information and neighbor information, outperforms the other two methods in all classification types (lane-changing left, lane-following, and lane-changing right) in terms of prediction accuracy.

B. Comparison among different trajectory lengths

We then obtained and compared the predicted accuracy rate for different trajectory lengths. Specifically, we set the history trajectory length of the LSTM structure to be 6, 9, and 12, comparing them against each other. The results are shown in Table II, and visualized in Fig. 6. We compared the results on five different trajectory sequences to help get a general understanding of the curve changing trend. The prediction accuracy increases as the history length grows in all prediction scenarios (lane-changing left, lane following,

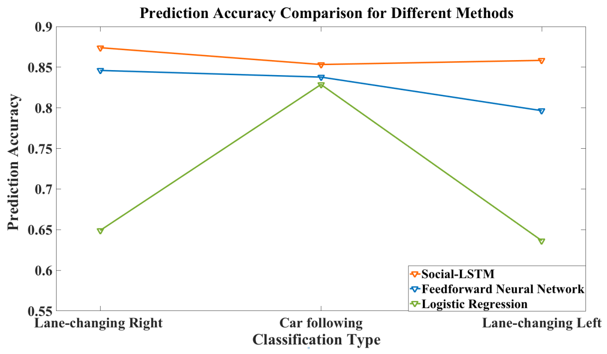


Fig. 5. Prediction accuracy comparison for different methods. SA-LSTM outperforms the other two in all classification types including lane-changing right, lane-following, and lane-changing left.

and lane-changing right). From common sense, the longer history trajectory length we have, the more information we will get from the previous trajectories, and the more accuracy will be obtained in the final results. However, there's a trade-off between length and computation time. The increase in accuracy also slows down as the length grows longer. As we could see from the figures, length=12 has only obtained a slightly higher accuracy rate compared to length=9. What is more, the history trajectory length could not be too long. Otherwise some factors which are non-relative to the current lane-changing intention will be introduced into the input. Based on the above analysis, it is reasonable for us to set length=12 where the accuracy increase rate slows down in this NGSIM scenario to obtain an accurate prediction while saving computation power at the same time. The analysis could also be adopted in real-world applications, and such trade-off between accuracy and computation power should always be considered when setting the proper history length parameter.

C. Comparison between with- and without- neighbor scenarios

Table III depicts the lane-changing prediction time generated by with- and without- neighbor-input-feature LSTM models, which is defined to be the time gap between the time at which the model predicts there will be a lane-change and the time at which the vehicle has actually reached the lane-changing point. The longer the time gap is, the more useful the prediction is. It can be seen from Fig. 7 that adding neighbor features prolongs the lane-changing prediction time in most cases.

SA-LSTM, History Length=12				
	Real Predict	Left	Following	Right
Sequence 1	Left	87.69%	11.72%	0.60%
	Following	11.80%	84.55%	3.65%
	Right	0.00%	12.66%	87.34%
Sequence 2	Left	84.85%	15.00%	0.15%
	Following	7.49%	88.71%	3.79%
	Right	4.79%	7.56%	87.66%
Sequence 3	Left	98.17%	1.83%	0.00%
	Following	14.32%	81.42%	4.26%
	Right	13.04%	0.00%	86.96%
Sequence 4	Left	90.76%	8.61%	0.63%
	Following	3.94%	81.59%	14.46%
	Right	0.58%	12.14%	87.28%
Sequence 5	Left	92.11%	7.89%	0%
	Following	2.71%	90.13%	7.16%
	Right	7.16%	17.65%	75.19%

(a)

SA-LSTM, History Length=9				
	Real Predict	Left	Following	Right
Sequence 1	Left	86.94%	12.56%	0.49%
	Following	12.87%	83.34%	0.79%
	Right	0.00%	14.50%	85.50%
Sequence 2	Left	83.13%	16.80%	0.07%
	Following	7.94%	88.01%	4.04%
	Right	4.65%	7.82%	87.53%
Sequence 3	Left	98.17%	1.83%	0.00%
	Following	14.99%	80.54%	4.48%
	Right	14.24%	0.00%	85.76%
Sequence 4	Left	89.82%	9.69%	0.49%
	Following	4.41%	80.04%	15.55%
	Right	2.69%	11.78%	85.53%
Sequence 5	Left	90.05%	9.95%	0.00%
	Following	3.53%	89.60%	6.86%
	Right	5.61%	21.08%	73.30%

(b)

SA-LSTM, History Length=6				
	Real Predict	Left	Following	Right
Sequence 1	Left	84.27%	15.44%	0.29%
	Following	23.49%	72.20%	4.31%
	Right	3.30%	28.66%	68.04%
Sequence 2	Left	77.21%	20.82%	1.98%
	Following	16.80%	78.72%	4.48%
	Right	8.55%	13.06%	78.38%
Sequence 3	Left	95.12%	4.88%	0.00%
	Following	24.84%	70.33%	4.84%
	Right	14.42%	12.50%	73.08%
Sequence 4	Left	78.47%	20.21%	1.32%
	Following	8.90%	72.40%	18.70%
	Right	2.15%	16.76%	81.09%
Sequence 5	Left	77.78%	22.22%	0.00%
	Following	13.92%	81.87%	4.21%
	Right	0.00%	30.01%	69.99%

(c)

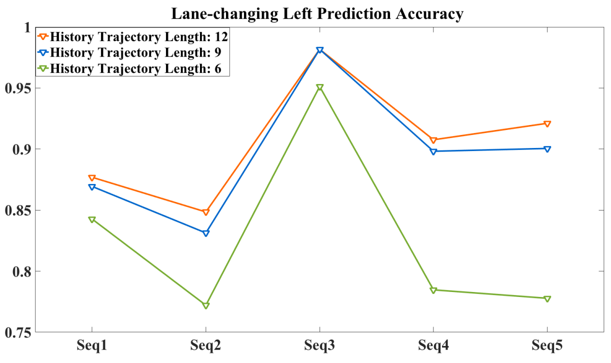
TABLE II. LANE-CHANGING PREDICTION ACCURACY COMPARISON AMONG DIFFERENT TRAJECTORY LENGTHS

	Lane-changing Left	Lane-changing Right
Sequence 1	1.17s/1.08s	1.00s/0.70s
Sequence 2	1.34s/1.31s	1.39s/1.31s
Sequence 3	1.59s/1.58s	1.48s/1.50s
Sequence 4	1.66s/1.50s	1.32s/1.33s
Sequence 5	1.66s/1.55s	0.75s/0.73s
Average	1.44s/1.31s	1.14s/1.03s

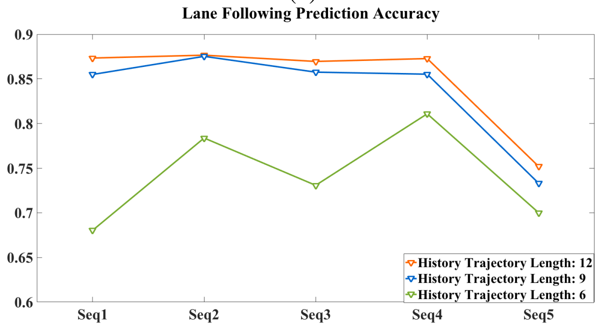
TABLE III. LANE-CHANGING PREDICTION TIME COMPARISON BETWEEN WITH-NEIGHBOR AND WITHOUT-NEIGHBOR SCENARIOS

V. CONCLUSIONS

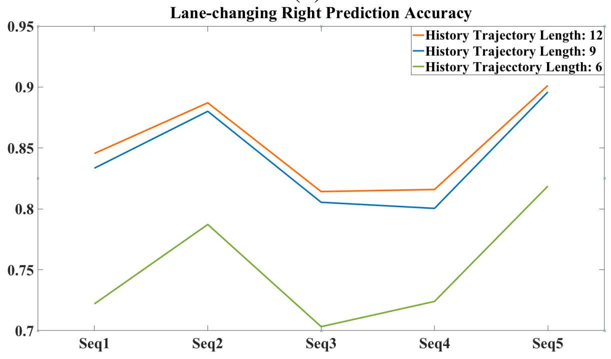
This paper proposes a LSTM network structure with the introduction of neighbor vehicles' features to make lane-changing intention predictions for each individual vehicle. We compared our methods with different network structures such as feed-forward neural network and logistic regression, as well as with LSTM structures without neighbor features to show the advantages of adding time and space information. We also compared the structure among different history trajectory lengths, and saved their influence on the final prediction results. Future work will mainly focus on extending the algorithm into practical scenarios, and seeing if the trained



(a)

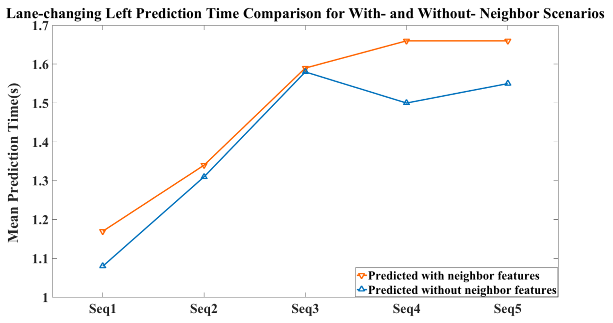


(b)

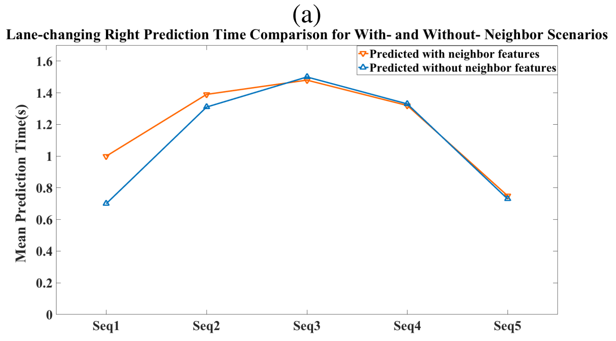


(c)

Fig. 6. We compared the prediction accuracy for different history trajectory lengths, the history time step length in the LSTM network structure. For each test sequence, the prediction accuracy increased as the history trajectory length grew.



(a)



(b)

Fig. 7. Prediction time comparison for with- and without- neighbor scenarios. Both lane-changing left and lane-changing right predictions show a increase, if not maintained, in prediction time after adding neighbor features.

network can be adopted on a real autonomous-driving car. The outstanding performance of the LSTM network also suggests the potential of attempting other recurrent network structures to further improve the prediction results in traffic scenarios.

ACKNOWLEDGMENTS

The authors would like to thank General Motors Company for their sponsorship and great help.

REFERENCES

[1] D. Clarke, J. W. Patrick, and J. Jean, "Overtaking road-accidents: Differences in manoeuvre as a function of driver age," *Accident Analysis & Prevention*, vol. 30, no. 4, pp. 455–467, 1998.

[2] Q. Yang and N. K. Haris, "A microscopic traffic simulator for evaluation of dynamic traffic management systems," *Transportation Research Part C: Emerging Technologies*, vol. 4, no. 3, pp. 113–129, 1996.

[3] C. R. Baker and J. M. Dolan, "Traffic interaction in the urban challenge: Putting boss on its best behavior," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008.

[4] J. Wei, J. M. Dolan, and B. Litkouhi, "A prediction and cost function-based algorithm for robust autonomous freeway driving." in *Intelligent Vehicles Symposium (IV)*. IEEE, 2010.

[5] C. Oh, C. J, and P. S, "In-depth understanding of lane changing interactions for in-vehicle driving assistance systems," *International journal of automotive technology*, vol. 18, no. 2, pp. 357–363, 2017.

[6] Y. Dou, Y. Fengjun, and F. Daiwei, "Lane changing prediction at highway lane drops using support vector machine and artificial neural network classifiers," in *IEEE International Conference on Advanced Intelligent Mechatronics*. IEEE, 2016, pp. 901–906.

[7] A. Kuefler, J. Morton, and T. Wheeler, "Imitating driver behavior with generative adversarial networks," in *IEEE Transactions on Intelligent Vehicles*. IEEE, 2017, pp. 204–211.

[8] D. Kasper, G. Weidl, T. Dang, G. Breuel, A. Tamke, A. Wedel, and W. Rosenstiel, "Object-oriented bayesian networks for detection of lane change maneuvers," *IEEE Intelligent Transportation Systems Magazine*, vol. 4, no. 3, pp. 19–31, 2012.

[9] G. Weidl, A. L. Madsen, V. Tereshchenko, W. Zhang, S. Wang, and D. Kasper, "Situation awareness and early recognition of traffic maneuvers," in *9th EUROSIM Congress on Modelling and Simulation*. IEEE, 2016.

[10] C. Dong, J. M. Dolan, and B. Litkouhi, "Intention estimation for ramp merging control in autonomous driving," in *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2017, pp. 1584–1589.

[11] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforce-

ment learning framework for autonomous driving." in *IST Electronic Imaging, Autonomous Vehicles and Machines*. IEEE, 2017, pp. 70–76.

[12] Z. Qiao, K. Muelling, J. Dolan, P. Palanisamy, and P. Mudalige, "Automatically generated curriculum based reinforcement learning for autonomous vehicles in urban environment." in *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2018.

[13] D. Sadigh, S. Sastry, S. A. Seshia, and A. Dragan, "Planning for autonomous cars that leverage effects on human actions," in *Robotics: Science and Systems*. IEEE, 2016.

[14] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, F. Li, and S. Savarese, Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2016, pp. 961–971.

[15] S. Hochreiter and S. Jurgen, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] "Next generation simulation fact sheet," in *ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm*. Federal Highway Administration, 2011.

[17] T. Wheeler. A julia package for handling the next generation simulation (ngsim) traffic dataset. [Online]. Available: https://github.com/sisl/NGSIM.jl

[18] J. Colyar and J. Halkias, "Us highway 101 dataset," *Federal Highway Administration(FHWA)*, vol. Tech, no. Rep, 2007.

[19] ——, "Us highway 80 dataset," *Federal Highway Administration(FHWA)*, vol. Tech, no. Rep, 2006.