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

Learning-based Multiple-Path Prediction for Early Warning

Ikuro Sato[1] and Liu Guoqing[1]

***Abstract*— It has been claimed that development of an in-vehicle early warning system is important to maximize collision avoidance capability. This study aims to provide a method that can predict the true ego-vehicle motion long enough so that early warning could be feasible, using a monocular forward-view camera. Based on an observation that several-second future position of a vehicle is probabilistic rather than deterministic, our approach utilizes a machine learning algorithm that can probabilistically model human driving behavior from data to predict multiple possible paths, such as going straight, turning left, and turning right. Experimental results on KITTI dataset suggest that our probabilistic learning approach could suffice for 4-second early warning, while an orthodox, deterministic learning approach and a simple motion extrapolation approach do not satisfy the requirement.**

I. INTRODUCTION

Time-To-Collision (TTC) is one of the important measures for collision avoidance systems [1], [6], [8]. A recent study by Wan, *et al*. concluded that early warning is especially important for reducing both kinetic energy at collision and collision rate [1]. They conducted experiments on a driving-simulator that verbally warns a driver of an out-of-sight hazardous object ahead under an assumption that some infrastructure sensing and communication systems are available. Their experimental results indicate that a warning of 4-8 seconds prior to a potential collision maximizes safety. In contrast, a typical Advanced Driver Assistance System (ADAS) available today warns much later, say 2-3 seconds prior to a potential collision. Such a late warning is far from optimal in terms of collision avoidance according to [1].

To have a reliable early warning functionality for a collision avoidance system, we at least need a method to predict an accurate position of the ego vehicle, say 4 seconds after the current time. To realize such a system, at least two types of technical difficulties as stated below must be resolved. The time period of 4 seconds could be long enough to mispredict future ego-vehicle position when multiple distinct paths are possible to follow, *e.g*., a situation of just entering an intersection. Possible positions of a vehicle that is approaching an intersection can vary by as much as 4 seconds, because it may go straight, turn left, or turn right. Therefore, if a target object is at position after a right turn, the system should be able to compute the time to reach that point based on a right-turn hypothesis. Conventional approaches to compute such a lead time (which we refer to as TTR: Time-To-Reach) based on simple kinematical extrapolation methods [7], [11] including Kalman Filter [5] do not deal with the multiple-path hypotheses. Recently proposed, learning-based methods [14], [15], [16] mimicking human controls also lack the capability of calculating multiple prediction. Another type of difficulty is that estimation of 4-second integration of change in motion of the ego vehicle can be highly non-trivial even when following a single path. Various environmental factors such as the visibility, road type, traffic situation, object locations, *etc.* and human factors such as personal driving preference and mercuriality affect driving behavior. Simple kinematical models, such as constant velocity or acceleration model, may be able to obtain a decent TTR prediction accuracy for a very short time interval [5], [7], [11]; however, there is no guarantee that such models work well for longer-time prediction. Frameworks of probabilistic motion prediction [8], [9], [10], [12], [13] could potentially deal with the problem, but it is almost impossible to take all the kinds of factors into account to thoroughly design by hand how a driver drives in real-world situations.

The aim of this work is to provide a framework to overcome the aforementioned difficulties. Using a monocular forward-view camera mounted on an ego vehicle, we deal with a problem of predicting the detailed motion of the ego vehicle to estimate TTR to a given point on a road.

(a) The proposed multiple-path predictor

(b) A single-path predictor

Fig. 1. Illustration of learning-based path-prediction methods for a validation data sample: (a) the proposed multiple-path predictor based on a probabilistic neural network and (b) a single-path predictor based on a non-probabilistic counterpart. Left column: front camera images. Right column: height images mapped from the Lidar point-cloud. All images show the true driving paths (white) and predicted paths (red) up to 6 seconds in the future. Our stochastic learning method can potentially predict exhaustive driving patterns (here only two out of many patterns are shown for visibility). In this case, one of the predicted paths successfully overlaps the true future path (the right turn). In contrast, an orthodox learning method can only predict a single pattern; therefore, misprediction can easily happen especially for a longer-time prediction as the figure shows.

[1]Authors are with Denso IT Laboratory, Inc., Tokyo, Japan. isato@d-itlab.co.jp

Our approach utilizes machine learning that can implicitly model human driving behavior from data in a probabilistic manner to predict multiple possible paths. The top row of Fig. 1 illustrates a typical, selected output of our method, while the bottom row illustrates a typical limitation of a non-probabilistic learning approach. The contributions of this paper are stated below.

- We propose a path-prediction framework based on a Stochastic Convolutional Neural Network (Stochastic CNN), designed to probabilistically return the possible paths of an ego-vehicle. From these, an expected TTR to a point on a path is calculated.
- Experimental results imply that 4-second early warning could be feasible in intersection scenes with the proposed method. On the other hand, a non-probabilistic learning-based method and a simple kinematical model fail to predict 4-second future position most of the time, so early warning is infeasible.

In this work we focus on non-hazardous scenes. Driving behavior prediction with a hazardous object visible from the driver is a harder problem, from the point of view of data collection. Still, we believe the proposed framework itself is beneficial because it provides a way of probabilistic modeling of possible driving patterns from data, working effectively for longer-time path predictions.

II. METHOD

A. Assumption on Human Driving and Modeling Strategy

Before discussing the method we propose, we state an underlying assumption about the probabilistic nature of human driving. We assume a driving behavior at an intersection is discretely probabilistic, because either going straight or turning left/right is possible. We further assume that, due to human nature, there is a continuous probability distribution in vehicle positions even when driving along a single path.

Our strategy is to use a learning method that can model an arbitrary multimodal probability distribution as a predictor for human driving behavior. Such an approach inherently deals with the difficulties described in the previous section. If one trains a model that can only return a deterministic driving pattern from the data of general driving scenes, including straight-going and turning behaviors, it would merely return a sort of average driving pattern, which may not always be meaningful, as is explained in detail in Section III.

One might think it is infeasible to reconstruct a probability distribution of driving at a particular location from a dataset with a finite numer of samples. Nonetheless, we assume that even if the number of data samples from a particular location is quite limited, there are relatively many data samples from *semantically* identical scenes. Similarity among *features* from such data samples would be high so that reconstruction of probabilistic driving behavior is feasible.

B. Stochastic CNN for Ego-Vehicle Behavior Prediction

We extend Stochastic Feedforward Neural Networks (SFNN) [2] to a convolutional architecture. We briefly describe SFNN first. SFNN are types of networks that can associate a given input x^i with probabilistic output y^i for all $i \in \mathcal{D}$, where $\mathcal{D}$ is the set of training data indices. The network has a set of "stochastic neurons", which have sigmoid activations $s \in [0,1]^{N_s}$ and binary values $h \in \{0,1\}^{N_s}$, where N_s is the total number of stochastic neurons in the network. The binary h_k takes 1 at a probability of s_k and 0 at a probability of $1-s_k$ for $k = 1, \cdots, N_s$. Note that an SFNN potentially returns 2^{N_s} distinct binary patterns. The objective function Q_θ, that is a lower bound on the complete data log-likelihood $\mathbb{E}_{i\in\mathcal{D}} \ln p(y^i|x^i)$, is

$$Q_\theta = \mathop{\mathbb{E}}_{i\in\mathcal{D}} \sum_{all\ h} w^i_{\theta'}(h) \left[\ln p_\theta(y^i|h,x^i) + \ln p_\theta(h|x^i)\right], \quad (1)$$

$$w^i_{\theta'}(h) = \frac{p_{\theta'}(y^i|h,x^i)p_{\theta'}(h|x^i)}{\sum_{all\ h'} p_{\theta'}(y^i|h',x^i)p_{\theta'}(h'|x^i)}, \quad (2)$$

where θ and θ' are the network parameters at the current and previous iteration, respectively. Output patterns returned by SFNN are associated with probabilities given by $p_\theta(h|x)$. For example, $p_\theta((1,1,\cdots)^\top|x) = \prod_{k=1}^{N_s} s_k$. The h-conditional likelihood function, $p_\theta(y|h,x)$, which represents the fitting quality of the binary pattern h, needs to be defined prior to training. Mathematical forms of Eq. (1, 2) are slightly modified from [2] in such a way that *all possible* (not sampled) binary patterns are evaluated. As we discuss in the next section, from our experience $N_s \simeq 10$ works fine on KITTI dataset. In this case, the computational cost for the exhaustive binary-pattern training is not dominantly high when an SFNN is attached to a preceding convolutional architecture, as the convolutional part generally demands the most intensive computations.

TABLE I
THE STOCHASTIC CNN ARCHITECTURE USED IN THIS WORK.

operation	RGB images	odometry
-	18(126, 446)	-
C, P	64(62, 222)	-
C, P	96(30, 110)	-
C, P	128(14, 54)	-
C, P	160(6, 26)	-
C, P	196(2, 12)	18(1, 1)
F	128(1, 1)	
F	(128+5)(1, 1)	
F	(128+5)(1, 1)	
F	128(1, 1)	
F	180(1, 1)	

Notations
C: 3×3 convolution
P: 2×2-grid max-pooling
F: full connection
A layer is represented by $n_m(m_y, m_x)$ with n_m the number of maps and (m_y, m_x) the 2D map size.
Numbers of stochastic units are underlined.

Table I shows the stochastic convolutional neural network architecture that we use in this work. The input to the network is: $x = (x_I, x_V, x_\Omega)$, where x_I is a set of RGB images, and x_V is a set of 2D translation vectors in meters and x_Ω is a set of yaw rates of the ego vehicles between frames in radians. In the experiment we used the most current six consecutive frames for x_I, x_V and x_Ω. We do not assume image x_I and odometry (x_V, x_Ω) are acquired at the same time, though they are in the experiment. The odometry (x_V, x_Ω) is assumed to be available by some means, say GPS/IMU. The output of the network is: $y = (y_V, y_\Omega)$, where y_V is a set of 2D translation vectors in meters and y_Ω

is a set of yaw rates of the ego vehicle between frames in radians for a predefined time duration T (in the experiment our model predicts up to 60 frames = 6 seconds). The h-conditional likelihood of the proposed method is defined as

$$p_\theta(y|h,x) = \prod_{t=1}^{T} \mathcal{N}(y_{V,t}(x,h;\theta)|g_{V,t},\sigma_V^2)\times \mathcal{N}(y_{\Omega,t}(x,h;\theta)|g_{\Omega,t},\sigma_\Omega^2), \quad (3)$$

where $g_{V,t}$ and $g_{\Omega,t}$ are the ground truths for the t-frame-future predictions $y_{V,t}$ and $y_{\Omega,t}$, respectively. The quantity $p_\theta(y|h,x)$ tells how good the path prediction by the binary vector h is. Note that a binary vector h yields a single path; therefore, there are at most 2^{N_s} paths for total. The likelihood definition in Eq. (3) may look oversimplified because i.i.d. assumption is made on consecutive odometry information; nevertheless, it turns out that the neural network does acquire appropriate time-series coordination. The variances σ_V^2 and σ_Ω^2 are hyperparameters fixed before training. Having defined the likelihood, Q_θ in Eq. (1) can be maximized by an EM-like algorithm, where calculation of the binary-pattern weights w_θ and gradient ascent for the weighted log-probabilities in Eq. (1) are alternately repeated.

III. Evaluation

Below, we describe baseline methods, metric, dataset, training details, and evaluation results.

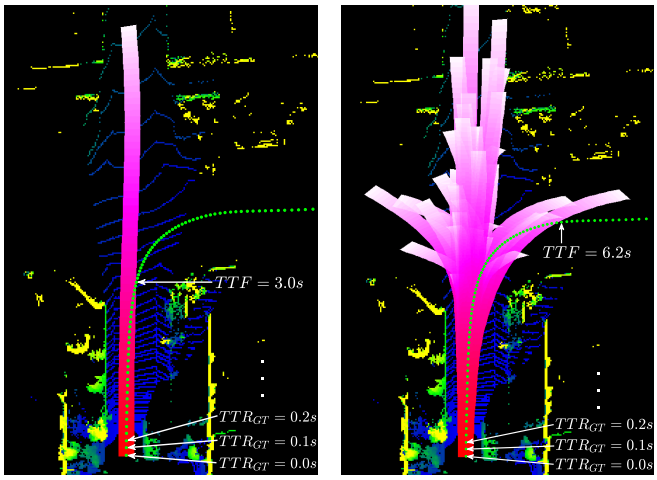
A. Baseline Methods

Baseline-1. This baseline serves as a non-machine-learning based method. The method simply performs future-odometry extrapolation from the most current odometry information. It assumes the rear wheels maintain a circular motion with a constant angular velocity perpendicular to the ground plane about a rotational center, through which the extended line of the rear axle passes. If the initial yaw rate is zero, then the model just assumes the vehicle goes forward at a constant velocity.

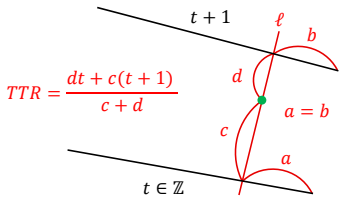
Baseline-2. This baseline serves as a neural-network based method that predicts *deterministic* future odometry, *i.e.*, it predicts a single path. The network architecture is the same as in Table I except that the number of stochastic units is zero. The objective function for baseline-2 is given by Eq. (3) without h-dependence.

B. Metric

In this work we evaluate quality of predicted paths by two metrics: Time-To-Fail (TTF) and Time-To-Reach (TTR) with respect to the ground points, through which the ego vehicle will actually pass. Before defining TTF and TTR, let us first give definitions of the ground-truth points and predicted paths. A *ground-truth point* at t-th frame ($t \in \{1,2,\cdots\}$) is defined as the middle point of the front wheels on the road at t-frame future. A *predicted path* is represented by a 2D polygon whose vertices are given by the t-th predicted positions of the left-front and right-front wheels on the road for all $t \in \{0,1,\cdots,T\}$. In Fig. 2(a, b), ground-truth points



(a) Single-path predictor (b) Multiple-path predictor



(c) The basic way to compute a TTR

Fig. 2. Descriptions of TTF and TTR. Ground-truth points (green dots) and predicted paths (red-white polygon) covered by the front axle with estimated TTR (red-white: 0-6s) are shown in two cases: (a) single-path predictor and (b) multiple-path predictor. A few ground-truth TTRs are marked as TTR_{GT}. The TTFs are also shown. Note that in (b) the *expected* TTR map is shown. (c) The basic way to interpolate TTR at a ground-truth point. Black line segments represent predicted positions of the front axle at t and $t+1$ in the future. A straight line ℓ passing through both line segments is drawn so that $a = b$. Then TTR at the ground-truth point is calculated by a linear interpolation using c and d as denoted in the figure.

(green dots) and predicted paths (red-white colored regions) are illustrated. In this work we regard the prediction for t-th ground-truth point as a success if this point is inside the polygon of the predicted path, and as a failure otherwise. Let us define *first-failure-point* as the first ground-truth point that is marked as a failure. As we will see in detail, a first-failure-point tends to come early for a single-path predictor as in Fig. 2(a), compared to the multiple-path predictor as in Fig. 2(b).

Time-To-Fail (TTF). TTF is the *true* (not predicted) time $t \in \{1,2,\cdots\}$ associated with the first-failure-point, as depicted in Fig. 2(a, b). Suppose 4-second early warning is required, then TTF needs to be at least 4 seconds long steadily. Note that even if a prediction looks successful (as the polygon has all the corresponding ground-truth points inside in an ideal fashion) a first-failure-point does exist because the polygon has a finite area, meaning there is always a ground-truth point outside the polygon. TTF in such a case may be greater than T.

Time-To-Reach (TTR). TTR is defined as a time duration for the ego vehicle to reach a given point on a road. For a predicted path, the TTR is interpolated on each of the

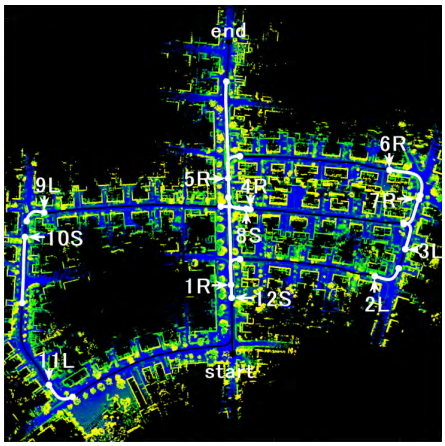


Fig. 3. The global map of the test dataset. White curves with dotted endpoints indicate evaluated paths, passing through selected intersections.

ground-truth points marked as successes in a particular way depicted in Fig. 2(c). In our method a ground-truth point can be covered by more than one path. In such a case, a *probability-weighted expectation value* is computed from the interpolated TTRs for a given ground-truth point, *i.e.*, $TTR_{pred} = \sum_{h\in\mathcal{H}} p(h|x)TTR(h) / \sum_{h'\in\mathcal{H}} p(h'|x)$, where $\mathcal{H}$ is a set of binary vectors that yield paths covering the point and $TTR(h)$ is the interpolated TTR produced by the network with binary vector h.

We evaluate TTR accuracies by comparing the true TTRs of ground-truth points. Obviously, TTR is not always computable, because the predicted path(s) may not successfully overlap with the ground-truth point. We evaluate TTR accuracies based on samples with computable TTRs, *i.e.*, we discard samples with TTR computations that failed.

Lastly, we emphasize that a method needs to show good performance in *both* TTF *and* TTR metrics in order for establishing a reliable early warning system.

TABLE II

THE DATASET USED IN THE EXPERIMENTS.

Data-set	Road type	#seq-uences	#frames	Remarks
Train-ing	Residen-tial	18	23,694	Excluded: 2011_09_26_drive_0020, 2011_09_26_drive_0079, 2011_09_26_drive_0093, other 529 frames with bad GPS data
	City	27	6,588	Excluded: 2011_09_26_drive_048, other 1,268 frames with bad GPS data
Valid-ation	Residen-tial	1	1,106	Used: 2011_09_30_drive_0027
Test	Residen-tial	1	See text.	Used: 2011_09_30_drive_0018

C. Dataset and Training Details

We used a part of the raw dataset in the KITTI dataset [4], as summarized in Table II. We chose the "residential" and "city" sequences because they contain scenes with frequent left and right turns in various environments. We excluded frames that appeared to us to have incorrect GPS data and movie sequences that either had too few frames or showed non-exemplary behavior of running onto a sidewalk. The GPS/IMU information that KITTI provides was used to generate the ego-vehicle's odometry for training and testing.

The test dataset was constructed in the following manner, aiming to extract scenes at intersections. The global map of the test dataset is shown in Fig. 3. There, each of the paths marked by white curves with dotted endpoints indicates a set of the ego-vehicle's positions, at which TTF and TTR are evaluated. To keep things simple, we excluded those intersections where the ego vehicle comes to a complete stop. The white curves are marked as 1R, 2L, 3L, 4R, 5R, 6R, 7R, 8S, 9L, 10S, 11L, and 12S, where S, L, and R mean straight, left- and right-turn, respectively. So, we have 3 straight, 4 left-turn, and 5 right-turn sequences. The corresponding numbers of frames are 370, 284, and 382. We extracted those frames by using the following rules. We first picked those frames at which the front axle just enters *mathematical* intersections of two roads (≈rectangles). Let us call these frames $\tau_1 < \tau_2 < \cdots < \tau_{12}$. We then defined $[\tau_k - 40, \tau_k + 30]$ to be the set of frames for evaluation, for every k except 6 and 12, *e.g.*, $[\tau_1 - 40, \tau_1 + 30]$ corresponds to 1R. The 6R curve experiences successive right-turns, thus we just extended the frame interval by hand so that it adequately includes right-turn behavior. The 12S curve likewise experiences successive straight-going behaviors passing through multiple intersections, thus we extended the frame interval in a similar manner.

Training details are as follows. The validation dataset is used to figure out a good early-stopping point. It turns out that 10 epochs are good enough for both our method and baseline-2. The final training is done on both training and validation datasets. AdaMax [3] optimizer is employed for both methods. Hyperparameters are: $\sigma_V = 0.01$(m) and $\sigma_\Omega = 0.01$(rad). In the proposed method, we set $N_s = 10$; *i.e.*, 10 stochastic neurons are used as in Table I.

Lastly, we explain about the heuristics that we adopted in data sampling. We found that a driving dataset such as KITTI is imbalanced, in a sense that the histogram of yaw rates has a strong peak in a very small vicinity around zero. Training such a dataset with uniform data sampling likely takes a very long time until it obtains adequate turning behavior. To avoid this, we adopt the following heuristics in data sampling. We first divide the entire training dataset into 4 subsets based on the magnitudes of the yaw change in 6 seconds. Threshold values are chosen so that subset sizes show a roughly 8:4:2:1 proportion, where the smallest subset corresponds to the group of the largest yaw changes. To construct a minibatch for gradient ascent, we repeat uniform sampling of a data subset and uniform sampling of a datum from the subset.

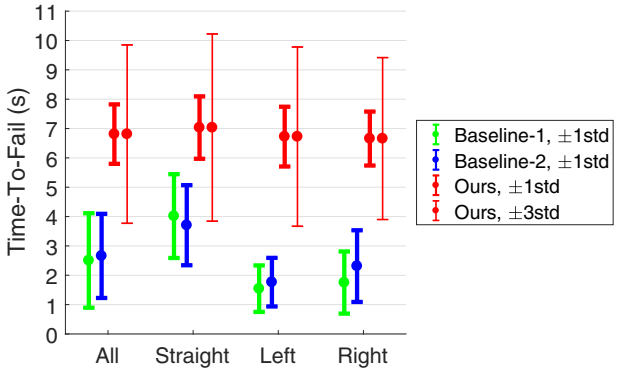


Fig. 4. Quantitative results of TTF evaluation. Dots indicate the mean TTF values. Bold (thin) error bars indicate ranges of ± 1 (± 3) standard deviations about the corresponding mean values.

D. Results

Time-To-Fail. Figure 4 shows the quantitative results of TTF evaluation among the proposed method and other two methods, baseline-1 and baseline-2. The mean TTF values of the proposed method are well above those of the baseline methods. The ± 1 standard deviation (std) error bars of the proposed method do not touch the error bars of the baseline methods. From this observation it is safe to conclude that the predicted paths of the proposed method more successfully overlaps with the true future paths of the ego vehicle. It is against our naive intuition that baseline-2 (the non-probabilistic CNN) works almost as poorly as baseline-1 (the motion extrapolation method base on a simple motion model). It can not be simply concluded that a learning-based approach is always advantageous compared to non-learning-based ones in the path prediction problem.

Time-To-Reach. Figure 5 shows the quantitative results of TTR evaluation among the proposed method, baseline-1 and 2. TTR accuracies of baseline-1 are quite low for left and right-turn sequences, where true TTRs are outside the ± 1 standard deviation range for $t > 4$ seconds. Both machine-learning methods, on the other hand, demonstrate good accuracies, *i.e.*, the mean values of predicted TTRs are more or less within 10% error range, and std values are less than or about 20% of the corresponding true TTRs. It is observed that the proposed method consistently has smaller std in all-sequence evaluation. From these observations, it can be concluded that a probabilistic learning approach is slightly advantageous compared to a non-probabilistic learning approach in the TTR estimation problem.

Remembering Wan's statement [1] that at least 4-second early warning is necessary for optimal collision avoidance, both baseline methods obviously do not suffice due to poor TTF, which is less than 4 seconds most of the time. The proposed method could suffice, from two experimental observations: 1) the mean TTF minus three std approximately equals 4 seconds, and 2) good TTR accuracy is obtained up to 5 seconds, in any of the straight, left and right sequences.

Limitations. We have found that the predicted paths by our method include some meaningless driving paths, such as running onto sidewalks or hitting walls, as illustrated in Fig. 6. One of the reasons behind such poor behavior is probably in the shortage of data samples. Another reason is in the likelihood formulation, in which only similarities in the odometry space are measured, regardless of whether the path is drivable or not. It will be our future work to eliminate such unwanted driving paths.

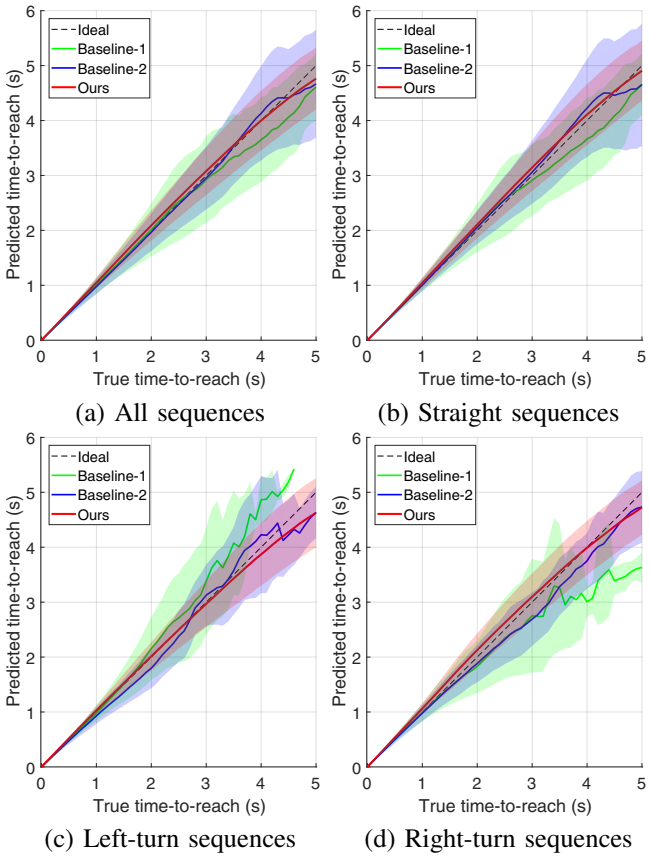


Fig. 5. Quantitative results of TTR evaluation. Mean values are plotted by bold lines and regions within $\pm$ one standard deviations are denoted by corresponding colors. Note that the number of samples with computable TTR monotonically decreases as the true TTR increases; *i.e.*, the number of samples used to compute the mean values and standard deviations monotonically decreases also in each method. Evaluation is performed for an interval where true TTR is less than 5 seconds. When true TTR is close to 6 seconds, predictor's TTRs are subject to be under-estimated statistically because they are upper-bounded by 6 seconds.

Baseline-2 often exhibits a peculiar behavior at the beginning of a turning, as shown in Fig. 7, where the predicted path indicates neither the straight nor the right turn, but somewhere in between. We think this is a moment at which the driver switches control from forward going to right turning. There are many training samples that are similar to this environment but driven differently; namely, some samples are of right-turning and the others are of straight-going. It can be said that a multimodal distribution in driving behavior resides in this type of scene. When a non-probabilistic CNN is trained on data with multimodally distributed output, the CNN most likely grows up to return a sort of the "averaged" values. We suspect this causes the peculiar behavior.

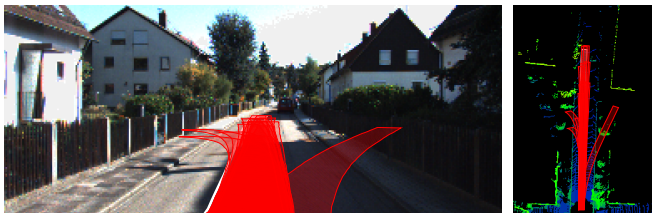


Fig. 6. Poor behavior produced by the proposed method for a validation data sample. Left: front camera images. Right: height images mapped from the Lidar point-cloud. Both images show the true driving paths (white) and all predicted paths (red) with probability larger than 0.01. A few meaningless paths that run onto sidewalks and even a residence are observed.

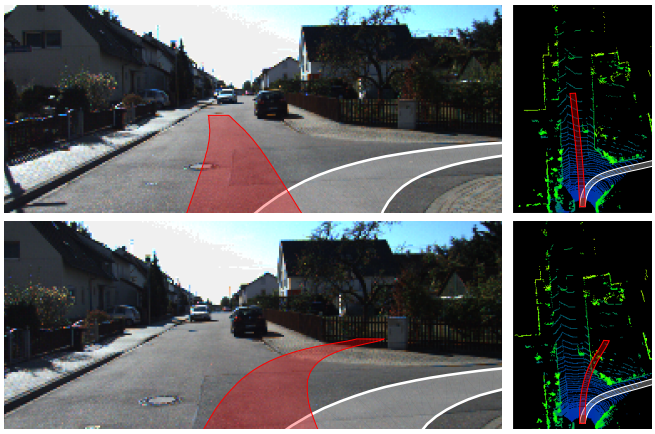


Fig. 7. A poor behavior produced by baseline-2, the non-probabilistic CNN, for a validation data sample. Left column: front camera images. Right column: height images mapped from the Lidar point-cloud. All images show the true driving paths (white) and predicted paths (red). The top row shows a scene where the vehicle is about to start to turn right. The predicted path looks quite ordinary, though it is mispredicted. The bottom row shows the scene after 0.3 seconds. Here, the predicted path indicates neither the straight nor the right turn, but indicates somewhere in between.

IV. Discussion

This study aims to develop a computer-vision algorithm that can predict the ego-vehicle's position accurately enough so that early warning for collision avoidance is feasible. Human driving behavior is in general considered to be probabilistic, *i.e.*, a discrete probability distribution resides in the choice of going straight/left/right at intersections, and a continuous probability distribution resides in a finer-scale maneuver. Our claim is that a learning method that can cope with probabilistic output can effectively model general human driving behavior. The proposed method that meets the above requirement can predict true paths accurately enough so that 4-second early warning could be feasible from experiments. A non-probabilistic counterpart and a simple kinematical model fail to yield reliable paths; thus the early warning is infeasible.

In contrast to our probabilistic learning approach, some previous studies employ deterministic learning approaches to mimic human controls for an autonomous driving purpose [14], [15], [16]. They attempt to train a deterministic CNN in an end-to-end fashion so that the CNN directly returns a steering angle from a front image input. The authors of [14] reported that the "behavior reflex ConvNet" (to use their term) obtained a very poor driving ability compared to a normal human driver. The authors of [15] successfully built a self-driving car with steering controlled by the learned network. The dataset they used only consists of data where the vehicle stays in a lane. Similar practice is also seen in [16]. We speculate that data selection to eliminate probabilistic behaviors would be crucial for a deterministic learning approach. Such an approach would be easily fooled once general driving scenes such as turns or lane changes are included in the training dataset. Unlike a typical image classification problem where solid ground truths are given, there is no solid ground truth in a problem of human driving behavior prediction. We believe that the probabilistic learning method proposed in this work serves as an appropriate tool for modeling general driving behavior.

References

[1] J. Wan, C. Wu, and Y. Zhang, The Effects of Lead Time of Verbal Collision Warning Messages on Driving Performance, Journal of Safety Research, Volume 58, pp. 89–98, Sep. 2016.

[2] Y. Tang and R. Salakhutdinov, Learning Stochastic Feedforward Neural Networks, Advances in Neural Information Processing Systems 26, 2013, pp. 530–538.

[3] J. Ba and D. Kingma, Adam: A Method for Stochastic Optimization, 3rd International Conference on Learning Representations, 2015.

[4] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, Vision meets Robotics: The KITTI Dataset, International Journal of Robotics Research, vol. 32, pp. 1231–1237, Sep. 2013.

[5] L. Thomas, S. T. Panicker, Jerry Daniel J, and T. Tony, DSRC based collision warning for vehicles at intersections, 3rd International Conference on Advanced Computing and Communication Systems, 2016, pp. 1–5.

[6] M. Sieber, and B. Färber, Driver Perception and Reaction in Collision Avoidance: Implications for ADAS Development and Testing, IEEE Intelligent Vehicles Symposium, 2016, pp. 239–245.

[7] E. Dagan, O. Mano, G. P. Stein, and A. Shashua, Forward Collision Warning with a Single Camera, IEEE Intelligent Vehicles Symposium, 2004, pp. 37–42.

[8] A. Berthelot, A. Tamke, T. Dang, and G. Breuel, A novel approach for the probabilistic computation of Time-To-Collision, IEEE Intelligent Vehicles Symposium, 2012, pp. 1173–1178.

[9] A. Broadhurst, S. Baker, and T. Kanade, Monte Carlo road safety reasoning, IEEE Intelligent Vehicles Symposium, 2005, pp. 319–324.

[10] J. Hillenbrand and K. Kroschel, A Study on the Performance of Uncooperative Collision Mitigation Systems at Intersection-like Traffic Situations, IEEE Conference on Cybernetics and Intelligent Systems, 2006, pp. 1-6.

[11] I. Gat, M. Benady, and A. Shashua, A Monocular Vision Advance Warning System for the Automotive Aftermarket, SAE Tech Pap Ser, 2005.

[12] N. Kaempchen, B. Schiele, and K. Dietmayer, Situation Assessment of an Autonomous Emergency Brake for Arbitrary Vehicle-to-Vehicle Collision Scenarios, IEEE Transactions on Intelligent Transportation Systems, vol. 10, no. 4, pp. 678–687, Dec. 2009.

[13] A. Lambert, D. Gruyer, G. S. Pierre, and A. N. Ndjeng, Collision Probability Assessment for Speed Control, 11th International IEEE Conference on Intelligent Transportation Systems, 2008, pp. 1043–1048.

[14] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, DeepDriving: Learning Affordance for Direct Perception in Autonomous Driving, Proceedings of the 2015 IEEE International Conference on Computer Vision, 2015, pp. 2722–2730.

[15] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, End to End Learning for Self-Driving Cars, arXiv:1604.07316, 2016.

[16] Z. Chen and X. Huang, End-to-end learning for lane keeping of self-driving cars, IEEE Intelligent Vehicles Symposium, 2017, pp. 1856–1860.