

# Predicting the Time Until a Vehicle Changes the Lane Using LSTM-Based Recurrent Neural Networks

Florian Wirthmüller , Marvin Klimke , Julian Schlechtriemen , Jochen Hipp , and Manfred Reichert 

**Abstract**—To plan safe and comfortable trajectories for automated vehicles on highways, accurate predictions of traffic situations are needed. So far, a lot of research effort has been spent on detecting lane change maneuvers rather than on estimating the point in time a lane change actually happens. In practice, however, this temporal information might be even more useful. This letter deals with the development of a system that accurately predicts the time to the next lane change of surrounding vehicles on highways using long short-term memory-based recurrent neural networks. An extensive evaluation based on a large real-world data set shows that our approach is able to make reliable predictions, even in the most challenging situations, with a root mean squared error around 0.7 seconds. Already 3.5 seconds prior to lane changes the predictions become highly accurate, showing a median error of less than 0.25 seconds. In summary, this article forms a fundamental step towards downstreamed highly accurate position predictions.

**Index Terms**—Intelligent Transportation Systems, AI-Based Methods, Automated Driving, Vehicle Motion Prediction.

## I. INTRODUCTION

**A**UTOMATED DRIVING is on the rise, making traffic safer and more comfortable already today. However, handing over full control to a system still constitutes a particular challenge. To reach the goal of fully automated driving, precise information about the positions as well as the behavior of surrounding traffic participants needs to be gathered. Moreover, an estimation about the development of the traffic situation, i. e. the future motion of surrounding vehicles, is at least as important. Only if the system is taught to perform an anticipatory style of driving similar to a human driver, acceptable levels of comfort and safety can be achieved. Therefore, every step towards improved predictions of surrounding vehicles' behavior in terms of precision as well as wealth of information is valuable.

Manuscript received October 14, 2020; accepted January 24, 2021. Date of publication February 11, 2021; date of current version March 15, 2021. This letter was recommended for publication by Associate Editor J. Kim and Editor Y. Choi upon evaluation of the reviewers' comments. (*Corresponding author: Florian Wirthmüller.*)

Florian Wirthmüller is with the Mercedes-Benz AG, 71034 Böblingen, Germany, and also with the Institute of Databases and Information Systems (DBIS), Ulm University, 89081 Ulm, Germany (e-mail: florian.wirthmueller@daimler.com).

Marvin Klimke is with the RWTH Aachen University, 52062 Aachen, Germany (e-mail: marvin.klimke@rwth-aachen.de).

Julian Schlechtriemen is with the Mercedes-Benz AG, 71034 Böblingen, Germany, and also with the Institute of Realtime Learning Systems, University of Siegen, 57076 Siegen, Germany (e-mail: julian.slechtriemen@daimler.com).

Jochen Hipp is with the Mercedes-Benz AG, 71034 Böblingen, Germany (e-mail: jochen.hipp@daimler.com).

Manfred Reichert is with the Institute of Databases and Information Systems (DBIS), Ulm University, 89081 Ulm, Germany (e-mail: manfred.reichert@uni-ulm.de).

Digital Object Identifier 10.1109/LRA.2021.3058930

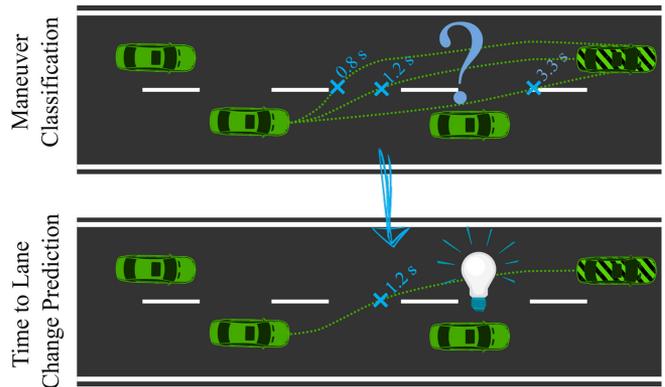


Fig. 1. A lot of previous works investigated systems that classify whether or not a lane change is going to take place. Instead, the proposed approach estimates the time to the next lane change directly. This information is more useful and covers the classification information implicitly.

Although many works in the field of motion prediction focus on predicting whether or not a lane change maneuver will take place, predictions on the exact point in time the lane changes will occur have not been well investigated. This temporal information, however, is extremely important, as emphasized by Fig. 1. Hence, this letter deals with the development of a system that predicts the time to upcoming lane changes of surrounding vehicles precisely. The system is developed and thoroughly evaluated based on a large real-world data set, which is representative for highway driving in Germany. As methodical basis, the state-of-the-art technique of long short-term memory (LSTM)-based recurrent neural networks (RNNs) is applied. Therefore, we form the basis for downstreamed highly accurate position predictions. The novelty and main contribution of our article results from using and thoroughly investigating known techniques with the special perspective of (vehicle) motion prediction rather than from developing completely new learning methods. Therefore, we changed the learning paradigm from classification to regression and obtained a significant gain in knowledge. In addition, to the best of our knowledge, there is no other article comparing an approach for time to lane change regression with a maneuver classification approach.

The remainder of this paper is structured as follows: Section II discusses related work. Section III then describes the proposed approach, followed by its evaluation based on real-world measurements in Section IV. Finally, Section V concludes the article with a short summary and an outlook on future work.

## II. RELATED WORK

An overview of motion prediction approaches is presented in [1], which distinguishes three categories: physics-based, maneuver-based, and interaction-aware approaches. Maneuver-based approaches, which are most relevant in the context of our work, typically define three fundamental maneuver classes: lane change to the left *LCL*, lane change to the right *LCR*, and lane following *FLW* [2]–[4]. These maneuver classes are used to simplify modeling the entirety of highway driving and its multimodality. Based on this categorization, the prediction problem is interpreted as a classification task with the objective to estimate the upcoming maneuver or the maneuver probabilities based on the current sensor data.

An approach that decomposes the lane change probability into a situation- and a movement-based component is presented in [2]. As a result, an  $F_1$ -score better than 98 %, with the maneuvers being detected approximately 1.5 s in advance, can be obtained. The probabilities are modeled with sigmoid functions as well as a support vector machine.

In [3], the problem of predicting the future positions of surrounding vehicles is systematically investigated from a machine learning point of view using a non-public data set. Among the considered approaches and techniques, the combination of a multilayer perceptron (MLP) as lane change classifier and three Gaussian mixture regressors as position estimators in a mixture of experts shows the best performance. The mixture of experts approach can be seen as a divide and conquer manner enabling to master modeling the complex multimodalities during highway driving. In order to achieve this, the probabilities of all possible maneuvers are estimated. The latter are used to aggregate different position estimates being characteristic for the respective maneuvers. In [4], the approach of [3] has been adopted to the publicly available highD data set [5], showing an improved maneuver classification performance with an area under the receiver operating characteristic curve of over 97 % at a prediction horizon of 5 s. Additionally, [4] studies the impact of external conditions (e.g. traffic density) on the driving behavior as well as on the system's prediction performance.

The highD data set [5] has evolved into a defacto standard data set for developing and evaluating such prediction approaches since its release in 2018. The data set comprises more than 16 hours of highway scenarios in Germany that were collected from an aerial perspective with a statically positioned drone. The recordings cover road segments ranging 420 m each. Compared to the previously used NGSIM data set [6], the highD data set contains less noise and covers a higher variety of traffic situations.

In opposition to the so far mentioned machine-learning based approaches, [1] introduced the notion 'physics-based' approaches. Such approaches mostly depend on the laws of physics and can be described with simple models such as constant velocity or constant acceleration [7]. Two well-known and more advanced model-based approaches are the 'Intelligent Driver Model' (IDM) [8] and 'Minimizing Overall Braking Induced by Lane Changes' (MOBIL) approach [9]. Such approaches are known to be more reliable even in rarely occurring scenarios.

Therefore, it is advisable to use them in practice in combination with machine learning models, which are known to be more precise during normal operation, to safeguard the latter's estimates.

Approaches understanding the lane change prediction problem as a regression task instead of a classification task and that are more interested in the time to the next lane change are very rare though. Two such approaches can be found in [10], [11].

In [10], an approach predicting the time to lane change based on a neural network that consists of an LSTM and two dense layers is proposed. Besides information about the traffic situation which can be measured from each point in the scene, the network utilizes information about the driver state. Therefore, the approach is solely applicable to predict the ego-vehicle's behavior, but not to predict the one of surrounding vehicles. Nevertheless, the approach performs well showing an average prediction error of only 0.3 s at a prediction horizon of 3 s when feeding the LSTM with a history of 3 s. To train and evaluate the network, a simulator-based data set covering approximately 1000 lane changes to each side is used.

An approach based on quantile regression forests, which constitute an extension of random decision forests, is presented in [11]. It uses features that describe the relations to the surrounding traffic participants over a history of 0.5 s and produces probabilistic outputs. The approach is evaluated with a small simulation-based as well as a real-world data set with 150 and 50 situations per lane change direction, respectively. The evaluation shows that the root mean squared error (*RMSE*) falls below 1.0 s only 1.5 s before a lane change takes place. In [12], this work is extended utilizing the time to lane change estimates to perform trajectory predictions using cubic polynomials. An approach based on quantile regression forests, which constitute an extension of random decision forests, is presented in [11]. It uses features that describe the relations to the surrounding traffic participants over a history of 0.5 s and produces probabilistic outputs. The approach is evaluated with a small simulation based as well as a real-world data set with 150 and 50 situations per lane change direction, respectively. The evaluation shows that the root mean squared error (*RMSE*) falls below 1.0 s only 1.5 s before a lane change takes place. In [12], this work is extended utilizing the time to lane change estimates to perform trajectory predictions using cubic polynomials.

Other approaches try to infer the future position or a spatial probability distribution [3], [4], [13]–[16]. As [13] shows, it is promising to perform the position prediction in a divide and conquer manner. Therefore, a system exclusively producing time to lane change estimates remains reasonable even though approaches directly estimating the future positions also determine that information as by-product.

The approach presented in [13] uses a random forest to estimate lane change probabilities. These probabilities serve as mixture weights in a mixture of experts predicting future positions. This approach has been extended by the above-mentioned works [3], [4], which have replaced the random forest by an MLP. The evaluations presented in [4] show a median lateral prediction error of 0.18 m on the highD data set at a prediction horizon of 5 s.

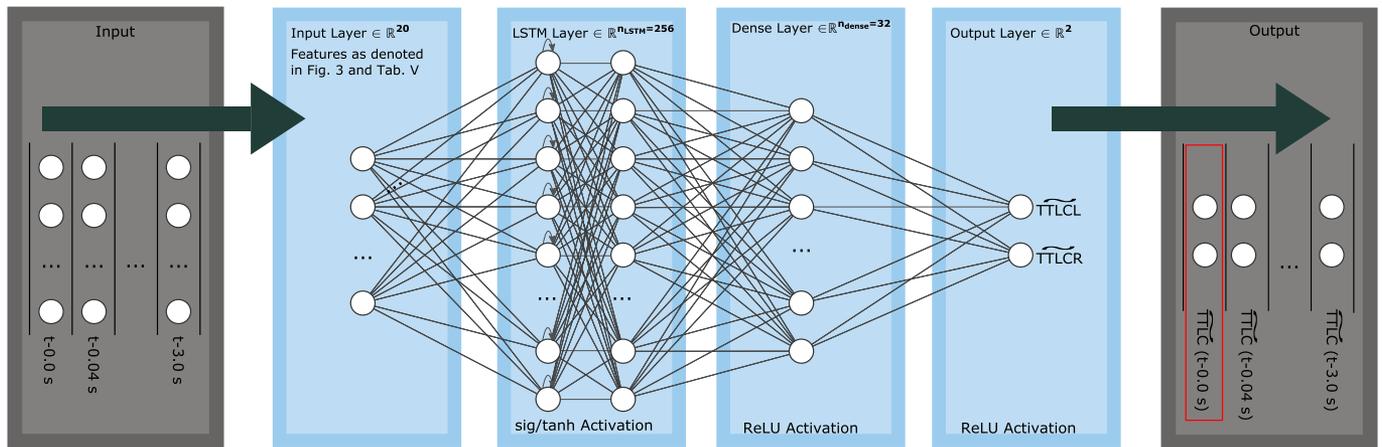


Fig. 2. Architecture of the used LSTM-based RNN together with its inputs and outputs. As the illustration indicates, it is necessary to feed the network with several consecutive measurements in order to take advantage of the recursive nature of the LSTM units. The relevant output is the most recently produced one, highlighted in red, as it is influenced by all previous measurements.

A similar strategy is applied by [14]. In this work, an MLP for maneuver classification as well as an LSTM network for trajectory prediction are trained using the NGSIM data set. In turn, the outputs of the MLP are used as one of the inputs of the LSTM network. The evaluation yields an *RMSE* of only 0.09 m at a prediction horizon of 5 s for the lateral direction when using a history of 6 s.

The approach presented in [15] uses an LSTM-based RNN, which predicts single shot trajectories rather than probabilistic estimates. The network is trained using the NGSIM data set. [15] investigates different network architectures. Among these architectures, a single LSTM layer followed by two dense layers using *tanh*-activation functions shows the best performance, i. e., an *RMSE* of approximately 0.42 m at a prediction horizon of 5 s.

[16] uses an LSTM-based encoder-decoder architecture to predict spatial probability distributions of surrounding vehicles. The used architecture is enabled to explicitly model interactions between vehicles. Thereby, the LSTM-based network is used to estimate the parameters of bivariate Gaussian distributions, which model the desired spatial distributions. Evaluations based on the NGSIM and highD data sets show *RMSE* values of 4.30 m and 2.91 m, respectively, at a prediction horizon of 5 s.

As our literature review shows, many approaches, and especially the most recent ones, use long short-term memory (LSTM) units. An LSTM unit is an artificial neuron architecture, which is used for building recurrent neural networks (RNNs). LSTMs have been firstly introduced by Hochreiter and Schmidhuber in 1997 [17].

The key difference between RNNs and common feedforward architectures (e.g. convolutional neural networks) results from feedback connections that allow for virtually unlimited value and gradient propagation, making RNNs well suited for time series prediction. To efficiently learn long-term dependencies from the data, the LSTM maintains a cell and a hidden state that are selectively updated in each time step. The information flow is guided by three gates, which allow propagating the cell memory without change. The latter contributes to keep the

problem of vanishing and exploding gradients, classic RNNs suffer from [18, Ch. 10], under control.

### III. PROPOSED APPROACH

The present work builds upon the general approach we described in [3], [4] but follows a fundamentally different idea. We replaced the previously used multilayer perceptron (MLP) for lane change classification by a long short-term memory (LSTM)-based recurrent neural network (RNN) predicting the time to an upcoming lane change. Consequently, the classification task becomes a regression task. For the moment of the lane change, we are using the point in time when the vehicle center has just crossed the lane marking [3]. Transforming the classification problem to a regression problem has in fact also the benefit, that the labeling is simplified, as it is no longer necessary to define the start and the end of the lane change maneuver. The latter is a really challenging task. Fig. 2 illustrates the proposed model architecture together with the inputs and outputs. The architecture consists of one LSTM layer followed by one hidden dense layer and an output layer. The dimensionality of the output layer is two, with the two dimensions representing the predicted time to a lane change to the left  $\widetilde{TTLCL}^1$  and to the right  $\widetilde{TTLCR}$ , respectively. In accordance with [17], the LSTM layer uses sigmoid functions for the gates and *tanh* for the cell state and outputs. By contrast, in the following dense layers rectified linear units (ReLU) are used. ReLUs map negative activations to a value of zero. For positive values, in turn, the original activation is returned. ReLUs have to be favored against classical neurons, e.g., using sigmoidal activation functions as they help to prevent the vanishing gradient problem. The use of ReLUs instead of linear output activations for a regression problem can be justified with the fact that negative  $\widetilde{TTLC}^2$  values cannot occur in the

<sup>1</sup>Symbols which are overlined with a tilde denote estimated values in contrast to the actual ones.

<sup>2</sup> $\widetilde{TTLC}$  stands for time to lane change values in general no matter to which direction the lane change is performed.

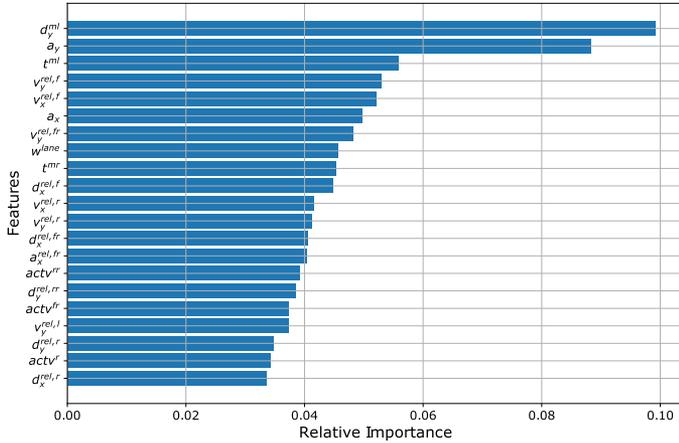


Fig. 3. Visualization of the relative feature importance values. As the relative importance values are derived from the model weights, they are related to a fitted model. In this case we chose the one with the optimal hyperparameters (see Table I). The feature identifiers are in accordance with the ones from [3], where the feature selection was carried out. A short overview can be found in the Appendix (Table V).

given context. While designing our approach, we also considered model architectures featuring two LSTMs stacked on top or using a second dense layer. Both variants provided no significant performance improvement. This observation is in line with the findings described in [15].

The used feature set is the same as in [4] and is based on the highD data set. The selection of the features is taken from [3], where data produced by a testing vehicle fleet is used to thoroughly investigate different feature sets. As opposed to [3], however, our approach omits the yaw angle as it is not available in the highD data set. Moreover, the transformation to lane coordinates is not needed as the highD data set solely contains straight road segments. The relative feature importance values are depicted in Fig. 3.

For each feature  $f$ , the importance value  $\iota(f)$  is calculated according to (1) as the sum of all weights  $w$  connecting that feature to the  $n_{LSTM}$  neurons of the LSTM layer:

$$\iota(f) = \sum_{n=1}^{n_{LSTM}} w(f, n) \quad (1)$$

The relative importance is calculated by normalization. As Fig. 3 indicates, the distance to the left lane marking  $d_y^{ml}$  and the lateral acceleration  $a_y$  play superior roles, whereas the importance of the other features is lower and quite similar.

In order to use the recursive nature of the LSTM units, one has to feed not only the current measurement values, but also a certain number of previous measurement values to the network. Although the network is able to output estimates each time an input vector has been processed, we are only interested in the last output. This is due to the fact that only in the last iteration, all measurements and especially the most recent ones are utilized for generating the prediction. This input/output interface of the network is illustrated in Fig. 2. The gray box on the left depicts a set of past measurements that are fed to the RNN as the input time series for a prediction at point  $t$ . The LSTM layer

TABLE I  
HYPERPARAMETERS IN THE GRID SEARCH

Hyper-Parameter	Values	Description
$n_{LSTM}$	{64, 128, <b>256</b> }	Output dimensionality of the LSTM layer
$n_{dense}$	{16, <b>32</b> , 64}	Number of neurons in the dense layer
$t_h$	{1 s, <b>3 s</b> , 5 s}	Length of the time period that is fed to the RNN
$\alpha$	{0.001, <b>0.0003</b> }	Learning rate for the Adam optimizer

continuously updates its cell state, which can be used to derive a model output at any time. This is indicated by the time series of  $TTL C$  estimates in the gray box on the right. The relevant final estimate is framed in red. In case a prediction is required for every time step, the LSTM is executed with largely overlapping input time series and reset in between.

The remaining hyperparameters, namely the dimensionality of the LSTM cell and the hidden dense layer, as well as the number of time steps provided and the learning rate are tuned using a grid search scheme [19, p. 7f]. Table I lists the hyperparameter values to be evaluated, yielding 54 possible combinations. This hyperparameter tuning scheme is encapsulated in a 5-fold cross validation to ensure a robust evaluation of the model's generalization abilities [3].

More precisely, for each possible combination of hyperparameters a model is trained based on 4 folds. Subsequently, the model is evaluated using the remaining fifth fold. This procedure is iterated so that each fold is used once for evaluation. Afterwards, the results are averaged and used to indicate the fitness of this hyperparameter set. As evaluation metric the loss function of the regression problem is used.

Given the aforementioned grid definition (see Table I), the following hyperparameter setup has proven to be optimal in the context of the present study: The output dimensionality of the LSTM  $n_{LSTM}$  results to 256 and the dense layer to a size of  $n_{dense} = 32$  units. Moreover, 3 s of feature history at 25 Hz, resulting in 75 time steps, is sufficient for the best performing model. As optimization algorithm we chose Adam [20], with  $\alpha = 0.0003$  as optimal learning rate.

When labeling the samples, the time to lane change values are clipped to a maximum of seven seconds, which is also applied to trajectory samples with no lane change ahead. The loss function of the regression problem is defined as mean squared error ( $MSE$ ). As the  $TTL C$  values are contained in the interval  $[0, 7]$  s, there are virtually no outliers that  $MSE$  could suffer from.

In order not to over-represent lane following samples during the training process, the data set used to train the model is randomly undersampled. Accordingly, only one third of the lane following samples are used. A similar strategy is described in [10]. Moreover, the features are scaled to zero mean and unit-variance.

Keras [21], a Python-based deep learning API built on top of Google's TensorFlow [22], is used to assemble, train, and

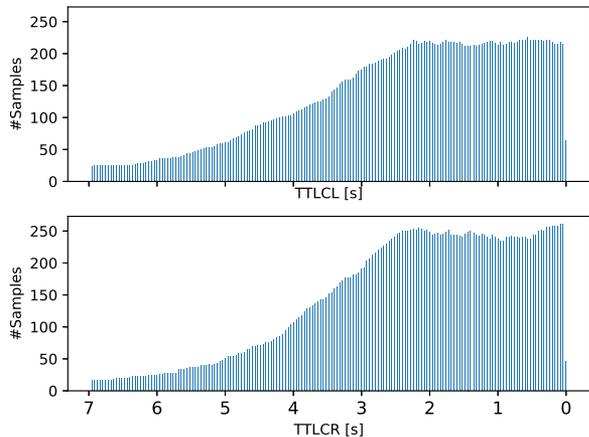


Fig. 4. Distribution of the unclipped time to lane change values. The upper part of the figure only contains samples with an upcoming lane change to the left. Hence, it solely depicts the time to the next lane change to the left  $TTLCL$ . The lower part, in turn, shows an equivalent representation for lane changes to the right. The used data set is not balanced over maneuver classes.

validate the RNN models. The grid search is performed on a high-performance computer equipped with a graphics processing unit, which is exploited by TensorFlow to reach peak efficiency.

#### IV. EVALUATION

To evaluate the resulting time to lane change prediction model, one fold of the highD data set is used. This fold was left out during model training and hyperparameter optimization. It is noteworthy that the used data sets are not balanced over  $TTLCL$ . This means, for example, that there are more samples with a  $TTLCL$  of 3 s than samples with a  $TTLCL$  of 5 s. This fact is illustrated by the histogram depicted in Fig. 4. The reason is that in the highD data set observations for individual vehicles rarely span over the full time of 7 s or more. However, this does not affect the following evaluations significantly. For all experiments we relied on the model, which showed the best performance during the grid search.

In the following, we evaluate two different characteristics of the proposed approach. First, we investigate how well the system solves the actual task, that is to estimate the time to the next lane change (cf. Section IV-A). Subsequently (Section IV-B), we deduce a maneuver classification estimate from the  $TTLCL$  estimates and perform a performance evaluation in comparison to existing works.

##### A. Time to Lane Change Prediction Performance

To investigate the system's ability to estimate the time to the next lane change, we consider the root mean squared error ( $RMSE$ ). This stands in contrast to the loss function that uses the pure mean squared error ( $MSE$ ) (see Section III). However, as evaluation metric the  $RMSE$  is beneficial due to its better interpretability. The latter is caused by the fact that the  $RMSE$  has the same physical unit as the desired output quantity, i. e. seconds in our case. Further note that the overall  $RMSE$  is not always the

TABLE II  
TIME TO LANE CHANGE PREDICTION PERFORMANCE ( $RMSE$  [s]) ON A BALANCED DATA SET

Maneuver	$LCL$	$FLW$	$LCR$	All
#Samples	21 603	21 182	21 656	64 332
Overall	0.497	0.097	0.526	0.416
$\widetilde{TTLCL}$	<b>0.674</b>	0.100	0.052	0.396
$\widetilde{TTLCR}$	0.202	0.094	<b>0.743</b>	0.435

most suitable measure. This fact shall be illustrated by a simple example: For a sample where the driver follows the current lane ( $FLW$ ) or performs a lane change to the right ( $LCR$ ), it is relatively straight forward to predict the  $TTLCL$ . By contrast, it is considerably more challenging to estimate the same quantity for a sample where a lane change to the left ( $LCL$ ) is executed. However, the latter constitutes the more relevant information. Therefore, we decided to calculate the  $RMSE$  values for the two individual outputs  $\widetilde{TTLCL}$  and  $\widetilde{TTLCR}$ . A look at the results presented in Table II makes this thought clearer.

To produce the results shown in Table II, we use a data set that is balanced according to the maneuver labels. The latter are defined according to [4].<sup>3</sup> The evaluation considers all samples with an actual  $TTLCL$  value below 7 s as  $LCL$  samples. Regarding  $LCR$  samples, an equivalent logic is applied. All remaining samples belong to the  $FLW$  class. In some very rare cases, two lane changes are performed in quick succession. Thus, a few samples appear in both  $LCL$  and  $LCR$ . This explains the slightly different number of samples, shown in Table II.

The first row of Table II depicts the overall  $RMSE$ . The  $RMSE$  can be monotonically mapped from the  $MSE$ , which is used as loss function during the training of the network. The two rows below depict the  $RMSE$  values separated by the outputs. The values we consider as the most relevant ones ( $\widetilde{TTLCL}$  estimation error for  $LCL$  samples and vice versa) are highlighted (bold font). Thus, the most interesting error values are close to 0.7 s. The other error values are significantly smaller but this is in fact not very surprising. This can be explained, as the system only has to detect that no lane change is about to happen in the near future in these cases. If this is successfully detected, the respective  $\widetilde{TTLCL}$  can simply be set to a value close to 7 s. Note that these values can be hardly compared with existing works (e.g. [10]) as the overall results strongly depend on the distribution of the underlying data set as well as the  $RMSE$  values considered. In addition, our investigations are based on real-world measurements rather than on simulated data.

In addition to the overall prediction performance, we are interested in the system's prediction performance over time. Obviously, the prediction task is, for example, significantly more difficult 4 s prior to the actual lane change than it is 1 s before it. To investigate this, we evaluate the  $RMSE$  and the distribution of the errors using boxplots as functions of the  $TTLCL$ , as shown in Fig. 5. Attention should be paid to the fact that the illustrated

<sup>3</sup>The definition of the labels essentially complies with the one presented in (2). As opposed to the shown equation, the actual times to the next lane change are used instead of the estimated ones.

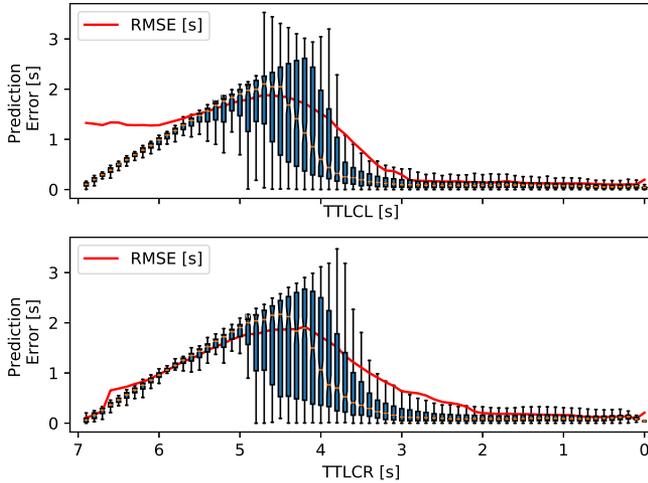


Fig. 5. *RMSE* (red) and error distribution (boxplot) as functions of the remaining time to the next lane change. The underlying data set is not balanced over the actual maneuver classes. The depicted values refer to the error values separated by the output channels.

values correspond to the errors separated by output channels as in Table II. For this investigation we rely on the unbalanced data set, meaning that considerably more *FLW* samples are included. An exact depiction of the label distribution can be found later on in Table IV. By using the unbalanced data set, more samples with *TTLCL* values between 5 s and 7 s remain in the data. Thus, the error values aggregated over *TTLCL* are assumed to be less noisy, especially between 5 and 7 s.

As shown in Fig. 5, the *RMSE* and the median values in the boxplots are mostly very close to each other, but the medians are more optimistic in general. Especially, this is the case in the upper part of Fig. 5 (arising lane change to the left) in the region between 7 s and 6 s. This can be explained with the fact that in this range the data density is relatively low. Thus, a single large error can significantly affect the *RMSE*, whereas this sample is considered as outlier in the boxplot. The illustrations show that our approach reaches very small prediction errors below 0.25 s already 3.5 s before the actual lane change moment. Even though a direct comparison to other approaches is also difficult for this quantity, it is noteworthy that [11] reports *RMSE* values below 0.25 s only 1 s before the lane changes. Conversely, our evaluations show comparable *RMSE* values already 2.5 s in advance of the lane change.

The fact that errors for large *TTLCL* values ( $>4.5$  s) are also very low can be explained as the system may not recognize such examples as lane changes. In that case, the system will solely output a *TTLCL* value of around 7 s. If, for example, the actual value corresponds to 6 s, the error is of course around 1 s. Thus, one can conclude that outputs, which are larger than the break even point of approximately 4.5 s are not very reliable. Note that this is in fact not surprising as predictions with such time horizons are extremely challenging.

Besides, it is known that lane changes to the left are easier to predict than the ones to the right [3], [23]. This is the reason that the *RMSE* values for lane changes to the right decrease slower over time than the values for lane changes to the left.

TABLE III  
MANEUVER CLASSIFICATION PERFORMANCE ON A DATA SET BALANCED OVER ACTUAL MANEUVER CLASSES COMPARED TO THE STUDY IN [4]  
THE PROCEDURE TO CONSTRUCT THE DATA SET CAN BE EXTRACTED FROM THE CONTINUOUS TEXT

	Maneuver #Samples	<i>LCL</i> 21 444	<i>FLW</i> 21 444	<i>LCR</i> 21 444	All/Mean 64 332
Prec.	[4]	0.937	0.899	0.968	0.933
	This study	1.000	0.859	1.000	0.953
	Benefit	+0.063	-0.040	+0.032	<b>+0.020</b>
Rec.	[4]	0.942	0.918	0.938	0.933
	This study	0.918	1.000	0.919	0.945
	Benefit	-0.024	+0.082	-0.019	<b>+0.012</b>
$F_1$	[4]	0.940	0.906	0.953	0.933
	This study	0.957	0.924	0.958	0.946
	Benefit	+0.017	+0.018	+0.005	<b>+0.013</b>

### B. Classification Performance

In addition to the preceding evaluations, we want to know how well our approach performs compared to a pure maneuver classification approach. This can be easily investigated by deriving the classification information from the time to lane change estimates. For this purpose, the logic depicted in (2) is applied:

$$L = \begin{cases} LCL, & \text{if } (\widehat{TTLCL} \leq 5 \text{ s}) \wedge (\widehat{TTLCL} \leq \widehat{TTLCR}) \\ LCR, & \text{if } (\widehat{TTLCR} \leq 5 \text{ s}) \wedge (\widehat{TTLCR} < \widehat{TTLCL}) \\ FLW, & \text{otherwise} \end{cases} \quad (2)$$

$\widehat{TTLCL}$  and  $\widehat{TTLCR}$  denote the estimated time to the next lane change to the left and to the right, respectively. The defined labels *LCL*, *LCR* and *FLW* are used to specify samples belonging to the three already introduced maneuver classes: lane change to the left, lane change to the right, and lane following. This definition matches the one used in [4] for the labeling. Also the prediction horizon of 5 s was adopted from [4] in order to ensure comparability. As lane change maneuvers usually range from 3 s to 5 s (see [24]), this is also a reasonable choice. The following investigations are, therefore, conducted in comparison to the approach outlined in [4], where an MLP for maneuver classification is trained using the highD data set (see Section II). We use the well-known metrics precision, recall and  $F_1$ -score, whose definitions can be found in [25, p. 182 f]. The results on a balanced data set are given in Table III.

This investigation shows that our newly developed LSTM network is able to perform the classification task – for which it was not intended – with a comparable or even slightly better performance than existing approaches. In particular, it is remarkable that not only the overall performance (measured with the  $F_1$ -score) is significantly increased with respect to the *FLW* samples, but also with respect to the *LCL* samples. The improved performance on the *FLW* class can be explained by the adapted training data set. While [4] uses a balanced data set, in this study we use a third of all *FLW* samples and thus significantly more than from the two other classes.

The overall slightly improved performance can presumably be attributed to the recurrent LSTM structure enabling the network to memorize past cell states. As opposed to this approach, [4]

TABLE IV  
MANEUVER CLASSIFICATION PERFORMANCE ON AN UNDERSAMPLED BUT NOT  
BALANCED DATA SET COMPARED TO THE STUDY IN [4]  
THE PROCEDURE TO CONSTRUCT THE DATA SET CAN BE EXTRACTED FROM  
THE CONTINUOUS TEXT

	Maneuver #Samples	<i>LCL</i> 21 444	<i>FLW</i> 190 370	<i>LCR</i> 23 601	All/Mean 235 332
Prec.	[4]	0.667	0.987	0.807	0.820
	This study	0.984	0.981	0.991	0.985
	Benefit	+0.317	-0.006	+0.184	<b>+0.165</b>
Rec.	[4]	0.942	0.920	0.937	0.933
	This study	0.918	0.997	0.917	0.944
	Benefit	-0.024	+0.077	-0.020	<b>+0.011</b>
<i>F</i> <sub>1</sub>	[4]	0.781	0.952	0.867	0.867
	This study	0.950	0.989	0.952	0.964
	Benefit	+0.169	+0.037	+0.085	<b>+0.097</b>

relies on the Markov assumption and, thus, does not model past system states. Although recurrent approaches can improve the prediction performance, Markov approaches have to be also taken into account when it comes to embedded implementations, as the latter ones are more resource-friendly.

Another interesting characteristic of our approach can be observed in Table IV, where its performance is measured on a data set which is undersampled in the same way as during the training.

As shown by Table IV, the new LSTM approach copes significantly better with the changed conditions (using an unbalanced instead of a balanced data set) compared to the MLP approach presented in [4]. On one hand, this is not surprising, as our network is exactly trained on a data set that is distributed in the same way. On the other, together with the results displayed in Table III, where the LSTM also performs quite well, it demonstrates that the LSTM approach is significantly more robust than the MLP. Nevertheless, note that in practice the MLP is applied together with a prior multiplication step. The probabilities estimated this way are then used as weights in a mixture of experts.

## V. SUMMARY AND OUTLOOK

This work presented a novel approach for predicting the time to the next lane change of surrounding vehicles on highways with high accuracy. The approach was developed and evaluated with regard to its prediction performance using a large real-world data set. Subsequently, we demonstrated that the presented approach is able to perform the predictions even during the most challenging situations with an *RMSE* around 0.7 s. Additional investigations showed that the predictions become highly accurate already 3.5 s before a lane change takes place. Besides, the performance was compared to a selected maneuver classification approach. Similar approaches are often used in recent works. Thus, it was shown that our approach is also able to deliver this information with a comparably high and in some situations even better quality. On top of this, our approach delivers the time to the next lane change as additional information.

The described work builds the basis for improving position prediction approaches by integrating the highly accurate time to

lane change estimates into a downstreamed position prediction. Our future research will especially focus on how to use these estimates in an integrated mixture of experts approach instead of maneuver probabilities as sketched in [3].

## APPENDIX

TABLE V  
FEATURE DESCRIPTION

Identifier	Description
$t^{ml}$	type of the left marking
$t^{mr}$	type of the right marking
$actv^{fr}$	activity status of the front right vehicle
$actv^r$	activity status of the right vehicle
$actv^{rr}$	activity status of the rear right vehicle
$w^{lane}$	width of the lane
$d_x^{rel,f}$	longitudinal distance to the front vehicle
$d_x^{rel,fr}$	longitudinal distance to the front right vehicle
$d_x^{rel,r}$	longitudinal distance to the rear vehicle
$d_y^{ml}$	lateral distance to the left marking
$d_y^{rel,r}$	lateral distance to the right vehicle
$d_y^{rel,rr}$	lateral distance to the rear right vehicle
$v_x^{rel,f}$	relative longitudinal velocity of the front vehicle
$v_x^{rel,r}$	relative longitudinal velocity of the front vehicle
$v_y^{rel,f}$	relative lateral velocity of the front vehicle
$v_y^{rel,fr}$	relative lateral velocity of the front right vehicle
$v_y^{rel,l}$	relative lateral velocity of the left vehicle
$v_y^{rel,r}$	relative lateral velocity of the right vehicle
$a_x$	longitudinal acceleration of the prediction target
$a_x^{rel,fr}$	relative longitudinal acceleration of the front right vehicle
$a_y$	lateral acceleration of the prediction target

TABLE VI  
ACRONYMS

Acronym	Description
<i>LCL</i>	lane change left - maneuver class
<i>LCR</i>	lane change right - maneuver class
<i>FLW</i>	lane following - maneuver class
<i>TTLC</i>	actual time to the next lane change
$\widetilde{TTLC}$	estimated time to the next lane change
$\widetilde{TTLC}^L$	actual time to the next lane change to the left
$\widetilde{TTLC}^R$	estimated time to the next lane change to the left
$\widetilde{TTLC}^R$	actual time to the next lane change to the right
$\widetilde{TTLC}^R$	estimated time to the next lane change to the right
<i>MSE</i>	mean squared error -
	$MSE = \frac{1}{\#Samples} \sum_{n=1}^{\#Samples} (\widetilde{TTLC} - TTLC)^2$
<i>RMSE</i>	root mean squared error - $RMSE = \sqrt{MSE}$
LSTM	long short-term memory - artificial neuron
RNN	recurrent neural network - neural network type
MLP	multilayer perceptron - neural network type
ReLU	rectified linear unit - artificial neuron

## REFERENCES

- [1] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH J.*, 2014, pp. 1–14.
- [2] C. Wissing, T. Nattermann, K.-H. Glander, C. Hass, and T. Bertram, "Lane change prediction by combining movement and situation based probabilities," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 3554–3559, 2017.
- [3] F. Wirthmüller, J. Schlechtriemen, J. Hipp, and M. Reichert, "Teaching vehicles to anticipate: A systematic study on probabilistic behavior prediction using large data sets," *IEEE Trans. Intell. Transp. Syst.*, to be published, doi:10.1109/TITS.2020.3002070.

- [4] F. Wirthmüller, J. Schlehtriemen, J. Hipp, and M. Reichert, "Towards incorporating contextual knowledge into the prediction of driving behavior," in *Proc. IEEE 23th Int. Conf. Intell. Transp. Syst.*, 2020, pp. 1–7.
- [5] R. Krajewski, J. Bock, L. Kloecker, and L. Eckstein, "The highD dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems," in *Proc. IEEE 21st Int. Conf. Intell. Transp. Syst.*, 2018, pp. 2118–2125.
- [6] J. Colyar and J. Halkias, "US highway 101 dataset," Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-07-030, 2007.
- [7] F. Wirthmüller, M. Klimke, J. Schlehtriemen, J. Hipp, and M. Reichert, "A fleet learning architecture for enhanced behavior predictions during challenging external conditions," in *Proc. 2020 IEEE Symp. Ser. Comput. Intell. (SSCI)*, 2020, pp. 2739–2745.
- [8] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E*, vol. 62, no. 2, 2000, Art. no. 1805.
- [9] A. Kesting, M. Treiber, and D. Helbing, "General lane-changing model mobil for car-following models," *Transp. Res. Rec.*, vol. 1999, no. 1, pp. 86–94, 2007.
- [10] H. Q. Dang, J. Fürnkranz, A. Biedermann, and M. Hoepfl, "Time-to-lane-change prediction with deep learning," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst.*, 2017, pp. 1–7.
- [11] C. Wissing, T. Nattermann, K.-H. Glander, and T. Bertram, "Probabilistic time-to-lane-change prediction on highways," in *Proc. IEEE 28th Intell. Veh. Symp.*, 2017, pp. 1452–1457.
- [12] C. Wissing, T. Nattermann, K.-H. Glander, and T. Bertram, "Trajectory prediction for safety critical maneuvers in automated highway driving," in *Proc. IEEE 21st Int. Conf. Intell. Transp. Syst.*, 2018, pp. 131–136.
- [13] J. Schlehtriemen, F. Wirthmueller, A. Wedel, G. Breuel, and K.-D. Kuhnert, "When will it change the lane? A probabilistic regression approach for rarely occurring events," in *Proc. IEEE 26th Intell. Veh. Symp.*, 2015, pp. 1373–1379.
- [14] A. Benterki, M. Boukhnifer, V. Judalet, and C. Maaoui, "Artificial intelligence for vehicle behavior anticipation: Hybrid approach based on maneuver classification and trajectory prediction," *IEEE Access*, vol. 8, pp. 56992–57002, 2020.
- [15] F. Altché and A. de La Fortelle, "An LSTM network for highway trajectory prediction," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst.*, 2017, pp. 353–359.
- [16] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi, "Non-local social pooling for vehicle trajectory prediction," in *Proc. IEEE 30th Intell. Veh. Symp.*, 2019, pp. 975–980.
- [17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: <http://www.deeplearningbook.org>
- [19] F. Hutter, L. Kotthoff, and J. Vanschoren, *Automated Machine Learning: Methods, Systems, Challenges*. Berlin, Germany: Springer Nature, 2019.
- [20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: <https://arxiv.org/abs/1412.6980>
- [21] F. Chollet *et al.*, "Keras," 2015. [Online]. Available: <https://keras.io>
- [22] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," *12th {USENIX} Symposium Operating Systems Design Implementation ({OSDI} 16)*, pp. 265–283, 2015. [Online]. Available: [www.tensorflow.org](http://www.tensorflow.org)
- [23] M. Bahram, C. Hubmann, A. Lawitzky, M. Aeberhard, and D. Wollherr, "A combined model-and learning-based framework for interaction-aware maneuver prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 6, pp. 1538–1550, Jun. 2016.
- [24] H. Woo *et al.*, "Lane-change detection based on vehicle-trajectory prediction," *IEEE Robot. Automat. Lett.*, vol. 2, no. 2, pp. 1109–1116, Apr. 2017.
- [25] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, Massachusetts & London, England: The MIT Press, 2012.