Lane Change Classification with Neural Networks for Automated Conversion of Logical Scenarios

Vjekoslav Diklić[1], Ivan Marković[2]

[1] dSPACE, Ltd., Zagreb, Croatia
[2]University of Zagreb Faculty of Electrical Engineering and Computing, Zagreb, Croatia
vdiklic@dspace.hr, ivan.markovic@fer.hr

Abstract. Validation and testing based on simulated scenarios is at the heart of the automotive industry for all vehicle development phases, since it enables perfect repeatability of the experiments and control of certain parameters. However, simulations alone can hardly capture all the complexities of the real world, thus true driving scenarios also represent an indispensable part of the process. Although invaluable, they offer very little freedom in changing the parameters, which motivates approaches for automated conversion of real-world driving scenarios to so-called logical scenarios, which can offer higher abstraction level. To be able to perform the complex process of converting real-world driving data, primarily it is necessary to be able to perform vehicle motion classification. For that purpose, this paper proposes and analyzes five different neural network models. The networks were trained and evaluated on a custom generated dataset to classify lateral vehicle behaviours in three main classes with respect to road lanes: lane keep, lane change right and lane change left. The dataset represents highway driving scenarios on a road with 7 lanes in the curvilinear coordinate system. Model training and evaluation was performed on four different subsets, each of them having a different signal-to-noise ratio. In the end, the best overall result was achieved with the network model composed of a bidirectional long-short term memory and multi-scale convolutional neural network layers.

Keywords: Lane Change Classification, Logical Scenarios, Deep Learning, Highway driving scenarios

1 Introduction

Scenario based simulation testing [1] and validation [2] represent standard practice in automotive industry for almost all phases of vehicle development [3]. The main motivation behind this lies in the ability to make a large number of tests with perfect repeatability and, depending on the abstraction level, with parameter changes to emphasize certain critical situations. There are two possible paths for generating driving scenarios: (i) manually from prior knowledge and experience, or (ii) from recorded real world driving as described in Menzel et al. [4]. The generated scenarios from real world data are much more valuable for testing and validation, but their abstraction level allows only the replay of the exact same scenario, hence the name *replay* scenarios or *concrete* scenarios [4]. In other words, it is not possible to change any parameters, like the vehicle velocity or the time instant of the started lane change. On the other hand, high level abstraction scenarios, know as *logical* scenarios [4, 5] are usually the result of manually generated scenarios and as such allow most, if not all, parameters to be changed. In order to increase the abstraction level of replay scenarios, it is necessary to have the capability to classify all vehicle motions, and based on those classification results recreate scenarios from the ground up. Manual conversion from a replay scenario to a logical scenario is possible, but the process is slow and with an increased number of scenarios it becomes impractical, thus automated scenario generation becomes a necessary step in the process.

The terms *scene*, *situation* and *scenario*, were defined within the automotive field by Ulbrich et al. [6], while the aforementioned concrete and logical scenarios were introduced by Menzel et al.[4]. Additionally, authors in [7] define a *data-driven* approach for generating driving scenarios for testing and validation with ISO 26262 [3] as the reference. Abstract scenario description with sequential acts and event triggering was proposed by Bash et al. [8] and has been used widely in both academic and industry applications, while a more mathematical description was provided by Andreotti et al. [9]. As a solution to the problem of how to generate a large number of testing scenarios, Zhao et al. [10] suggested a conversion of real world driving data through automated classification algorithms based on big data and ontology [11]. Practical solutions to automated scenario generations are mostly very specialized, and depend on the used methods, type and number of vehicle motions recognizable by the method, and format of the input and output data. For example, Lages et al. [12] and Made el al. [13] base their automated conversion on laser sensors. In [14–17] vehicle motion classification has been mostly based on some form of a machine learning algorithm. Other than learning methods, some authors like Hülnhagen et al. [18] used fuzzy logic, while a hybrid method with neural networks and fuzzy logic has been presented by Lin et al. [19]. Sonka et al.[20] used probabilistic and fuzzy classification achieving combined accuracy of up to 95%.

This paper compares the performance of five different neural networks for the problem of classifying lateral vehicle motion.With the assumption that the vehicle is driving in a highway scenario and that the vehicle position is given in the curvilinear coordinate system. Such interpretation of input data allows to neglect most, if not all, dependencies on the scenario recording process, as long as the recorded data can afterwards be successfully converted to the assumed format. Using only lane change based labels for lateral behaviour allows the output data to be used in most driving simulators. The performance of the five different neural networks was analyzed on a custom generated dataset and defined three main classes with respect to road lanes: lane keep, lane change right and lane change left. Final results showed that the best overall score was achieved with the network model composed of bidirectional long-short term memory and multi-scale convolutional neural network layers.

2 Problem Description

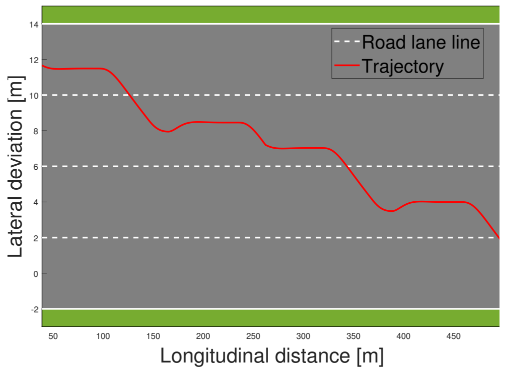
To achieve a more generalized level of input data and an increase the ease of use, the input data format is defined in the shape of an array that consists of vehicle trajectory and road lane coordinates in the curvelinear coordinate system [21]. The curvilinear coordinate system, widely used both in related work and industry, is a road reference curve based coordinate system SD where S is position on the road curve and D is lateral deviation from the road centre curve. This coordinate system allows an easier description of vehicle movement, the ability of easy target road change and the manipulation of vehicle trajectory in the position and time domains, respectively. Conversion from and to curvilinear coordinates is done by referencing the vehicle relative position to the road centerline, which is usually stored in a separate road description file. Road lane lines can also be represented in curvilinear coordinates as an array of lateral distances D, calculated with regard to the vehicle's specific position on road S. The size of the input array $(m + 2) \times n$ depends on the number of samples n and the number of lane lines m. This approach enables importing many different types of recorded data as long as they can be converted to this specific, but simple format. An example of the data can be seen in Fig. 1a. Additionally, for faster learning and convergence, the input data is normalized so that it is in range $[0, 1]$ by the expression (1), where a_i represents a single value from the array and a represents the whole array.

$$a_i = \frac{a_i - \min(a)}{\max(a) - \min(a)}. \tag{1}$$

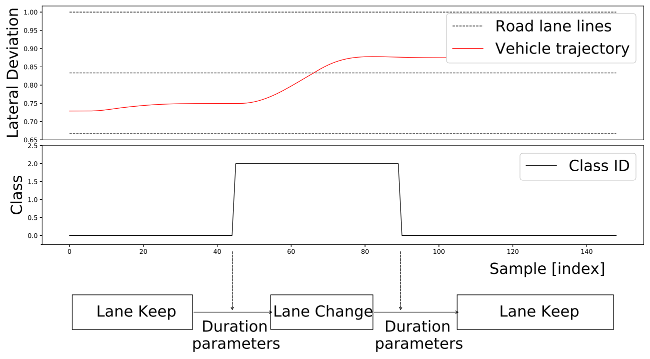
A prerequisite for generating logical segments from trajectory data is classified data, usually both in the lateral and the longitudinal direction and the process is illustrated in Fig. 1b. Simply put, it is necessary to have the capability to recognize the exact moment when the vehicle initiates and finishes the lane change, and this paper covers lateral classification with three possible classes: lane keep (0), lane change right (1), and lane change left (2). For simple highway driving scenarios the longitudinal classes can be described in between lateral classes with time, start velocity, and end velocity. The chosen classification is a simplification of the one described in [11, 14, 20], and they closely match the scenario description language of the dSPACE ASM ModelDesk [22] that was used for scenario generation in this paper.

3 Proposed Neural Network Models for Lane Change Classification

Driven by the work done in vehicle motion classification using neural networks [23], in this section five different neural network models with different



(a) Example of input data



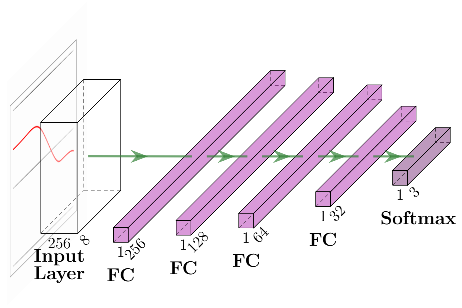
(b) Logical scenario conversion illustration

Fig. 1: Vehicle motion classification problem for automated logical scenarios.

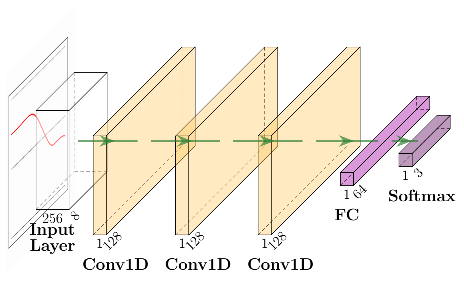
levels of complexity are proposed. This was done in order to determine which model is suitable for this specific classification task. The tested models range from simple fully connected (FC) and convolutional neural networks (CNNs), to combinations of convolutional and long-short term memory (LSTM) network layers, and in the following each of them is presented.

Model 1: Simple FC. A Fully Connected (FC) network [24] is an introductory network for simple classification problems. The proposed network model is composed of 4 fully connected layers and a softmax layer, where FC layers are composed of the following neuron numbers per layer (256,128,64,32) as shown in Fig. 2a. The main advantage of the FC neural networks is that no special assumptions about the structure of the input data is needed.

Model 2: Simple CNN. This network model is proposed in order to consider the spatial nature of input data, as they represent vehicle position in reference to road lane lines. The model is composed of three convolutional layers each containing 32 filters and kernel dimensions of 7, 5 and 3, respectively.



(a) Model 1: Simple FC



(b) Model 2: Simple CNN

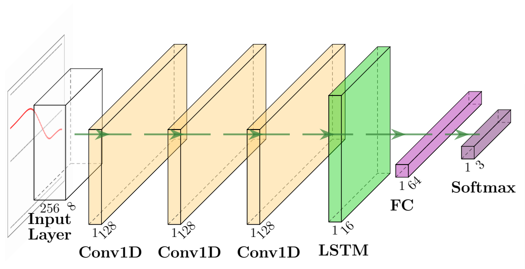
Fig. 2: Two straightforward neural network models for vehicle motion classification.

Convolutional layers are followed by a FC layer with 64 neurons, and a softmax layer for output as shown in Fig. 2b. The advantage of convolutional layers, which are inspired by visual perception, is that they take tensors as input data and can understand spatial relations (correlation between nearby pixels in the image) [25].

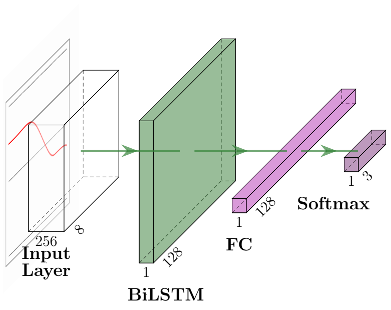
Model 3: CNN + LSTM. Models that combine CNN and LSTM layers were initially developed for visual time series prediction and the purpose of generating textual descriptions from sequences of images [26]. A network model based on a combination of convolutional layers with LSTM layers is proposed because of its ability to tackle spatio-temporal information in input data. Configuration of a simple CNN model is extended with an LSTM layer with 32 neurons inserted after 3 convolutional layers and before the FC layer, and the output softmax layer, as shown in Fig. 3a.

Model 4: Simple Bi-LSTM. This model was chosen to see how a temporal model deals with structured input data and how structured input data will influence the model's classification capability [16]. Composition of the network is based on two bidirectional LSTM (Bi-LSTM) layers with 64 cells, accompanied by a FC layer with 64 neurons and a softmax layer as output, as shown in Fig. 3b. The model based on Bi-LSTM is capable of running input data in both directions, which allows to preserve information from both past and future.

Model 5: Bi-LSTM + MSCNN. This network model was taken from [16] in which it was applied for driver behaviour recognition and it achieved accuracy of 85.27%. According to [16], the advantage of the Bi-LSTM over LSTM in this application is: "Compared to vanilla LSTM, bidirectional LSTM (Bi-LSTM) architecture can use the bidirectional information of the trajectory which prompts us to introduce a Bi-LSTM module to model the dynamic evolution of trajectories.". The structure of the model is composed of two main branches which are a Bi-LSTM and multi-scale CNN (MSCNN) followed by a FC layer



(a) Model 3: CNN + LSTM



(b) Model 5: Simple Bi-LSTM

Fig. 3: Neural network models using convolutional and LSTM layers and a bidirectional LSTM model.

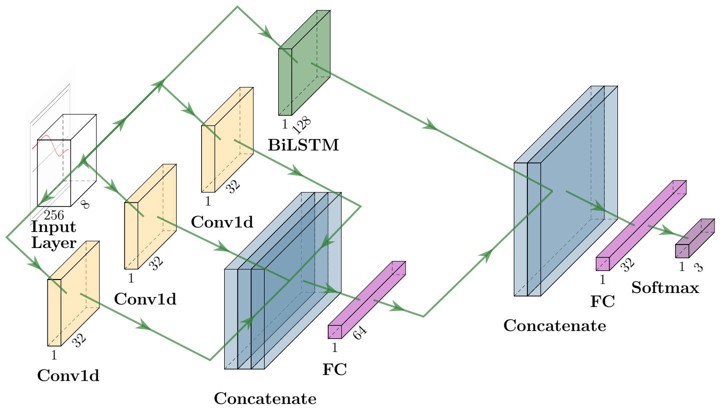


Fig. 4: Model 5: Bi-LSTM + MSCNN

with 32 neurons, and a softmax output layer. The Bi-LSTM module is composed of two LSTM layers each containing 64 cells, while the MSCNN branch has three parallel convolutional layers, accompanied by a FC layer with 64 neurons. Convolutional layers each consist of 32 filters and kernel sizes of 7,5 and 3, respectively.

4 Training and Evaluation

4.1 Dataset Generation

In order to train and evaluate the aforementioned neural network models, a custom dataset was created using dSPACE ASM ModelDesk simulation Toolbox [22]. Although there are trajectory based datasets, such as HighD[27], they lack

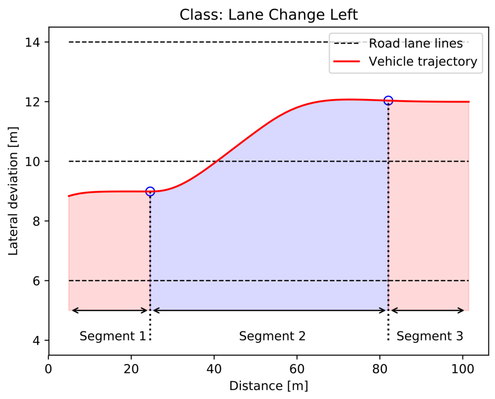


Fig. 5: Base scenario for dataset generation

labels for lane changes and this work can be seen as a step in development of automated annotation of such real world data. Since the goal was to generate a large amount of diverse scenarios, simulated logic scenarios were run in the loop while changing parameters for each loop pass. Base scenarios consisted of multiple lane keep and lane change segments stacked one after the other, where the base scenario is shown in Fig. 5. The road was set as a straight highway with 7 lanes all going in the same direction and curvatures were unnecessary due to conversion to the curvilinear coordinate system (which would neglect all curvatures). Vehicle positions were recorded as lateral deviations on the road. The parameters were changed in two ways: either randomly by using a uniform distribution, or linearly by growing with a preset step value. Parameters that were changed during the simulation were: duration of each block, lateral and longitudinal velocity, lateral offset, and lane change type. Recording of the data was done with a sample rate of 0.02 seconds, after which subsampling of 1:40 was applied to reduce the total size of the dataset and training time. This resulted in 2912 driving scenarios where vehicles drove 1002 km, in duration of more than 20 hours with 975 left lane changes and 1145 right lane changes.

The generated dataset was divided in a 80:20 ratio, where 80% was used for learning and 20% for evaluation. To enrich the dataset, it was further augmented with white noise based on the following expression 2 and shown in Fig. 6:

$$SNR = 10\log_{10}\left(\frac{P_{signal}}{P_{noise}}\right). \qquad (2)$$

White noise was applied on the training dataset in steps of signal to noise ratios of 70, 50 and 30 dB, to simulate a more realistic input data. These values ware chosen as representative examples of noise distributions. During the training of models, noise was progressively increased starting with training on data without any noise and ending with fully noisy data at 30 dB signal-to-noise ratio as shown in Fig. 7.

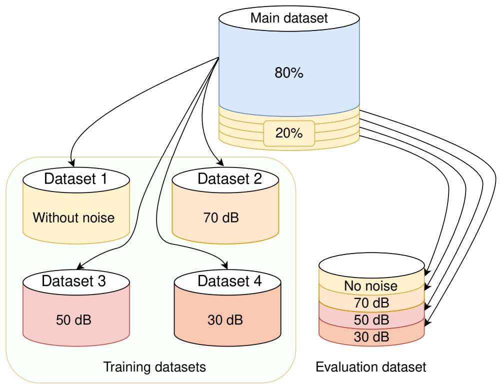


Fig. 6: Division of the dataset for training

Evaluation was compiled from four non-overlaping chunks of data with applied different levels of noise, starting without any noise and ending with 30 dB signal-to-noise ratio, same as the last part of training data. Adam [28] was used as the training optimizer based on stochastic gradient descent since it is well suited for problems that are both large in data and the number of parameters. The following metrics for evaluation were used: accuracy calculated as frequency of prediction matching labels, categorical cross-entropy for the loss function, and confusion matrix macro parameters: True positive (TP), True negative (TN), False positive (FP) and False negative (FN).

Table 1 shows the evalution results, from which it can be see that the best overall score was achived by the Bi-LSTM + MSCNN model. This is most likely due to its ability to combine spatio-temporal features of backward and forward temporal looking capability of Bi-LSTM and spatial characteristics of MSCNN module. The worst performing model was the simple FC because it makes no assumptions about the spatio-temporal correlation of the input data, followed by the simple CNN whose performance was significantly better but does not have the capability to learn temporal aspects of data. CNN + LSTM model showed initial poor performance, but gradually got better as it learned on noisy

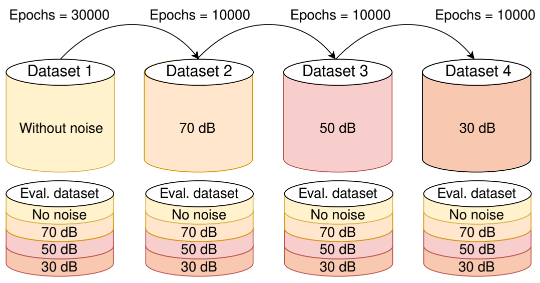
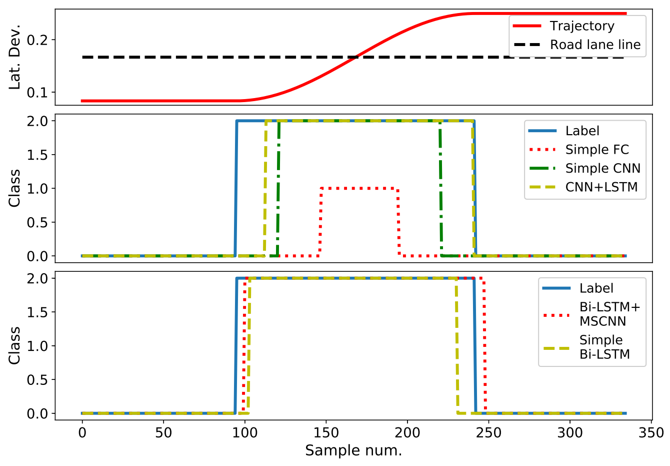


Fig. 7: The dataset in the training process

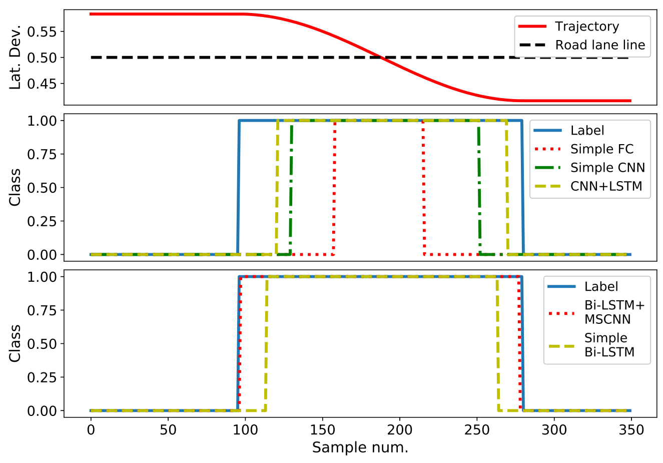
Table 1: Evaluation results of the analyzed network models on the custom lane change detection dataset

Model Name	Value	Without Noise Epoch=30000	Noise 70 dB Epoch=40000	Noise 50 dB Epoch=50000	Noise 30 dB Epoch=60000
Simple FC	Accuracy	74.4889%	74.0898%	74.6113%	74.3476%
	Loss	0.6881	0.6923	0.6911	0.6952
	TP	0.44	0.44	0.44	0.4333
	TN	1.4366	1.44	1.4399	1.4333
	FP	0.5566	0.56	0,56	0.5667
	FN	0.5566	0.56	0,56	0.5667
Simple CNN	Accuracy	72.0325%	75.4199%	88.1666%	88.9023%
	Loss	0.6589	0.6963	0.3227	0.3106
	TP	0.38	1.1933	0.8233	0.8
	TN	1.3799	1.06	1.82	1.8
	FP	0.62	0.42	0.1733	0.2
	FN	0.62	0.3184	0.1733	0.2
CNN+ LSTM	Accuracy	79.1458%	90.1328%	90.513%	90.9863%
	Loss	0.5148	0.3038	0.2712	0.2586
	TP	0.7333	0.89	0.86	0.8533
	TN	1.7333	1.8867	1.86	1.8566
	FP	0.2667	0.1066	0.1399	0.1499
	FN	0.2667	0.1066	0.14	0.1499
Bi-LSTM+ MSCNN	Accuracy	**96.6529%**	94.8645%	**97.2591%**	**97.2929%**
	Loss	0.1049	0.1375	0.0852	0.0976
	TP	0.9667	0.94	0.9667	0.9667
	TN	1.9667	1.9399	1.9667	1.9667
	FP	0.0333	0.06	0.0333	0.0333
	FN	0.0333	0.0599	0.0333	0.0333
Simple Bi-LSTM	Accuracy	94.7434%	**96.9804%**	93.1367%	93.3509%
	Loss	0.1401	0.1212	0.2233	0.1458
	TP	0.9367	0.96	0.92	0.9067
	TN	1,94	1.9533	1.9167	1.9067
	FP	0.0667	0.04	0.08	0.0933
	FN	0.0666	0.04	0.2899	0.0933

data. That is in correspondence with results from [29], which states that LSTM is less susceptible to perturbation in data if noise is present. This model could provide better results if LSTM layer is replaced by Bi-LSTM which would make it backward and forward looking capable. The simple Bi-LSTM model, that does not include the MSCNN part, has initially showed good accuracy until data with larger noise was introduced. Figure 8 shows the results of classification for all models, after having completed the whole learning process (60000 epochs), on the typical examples of lane change left and lane change right without added noise. According to Table 1, which contains five different architectures of neural networks, the final architecture recommendation would be: series of convolutional



(a) Lane Change Left



(b) Lane Change Right

Fig. 8: Evaluation results

layers whose outputs are combined with input and feed into LSTM and then into series of fully connected layers followed by softmax.

5 Conclusion

This paper investigated lane change classification with neural networks for automated logical scenario conversion. The lane change classification problem was categorized into three classes of lateral vehicle motion: lane keep, lane change right, and lane change left. This classification problem was tackled by five different deep learning models with varying complexity. Neural network models were trained and evaluated on a custom dataset that was generated using the dSPACE ASM ModelDesk. The resulting dataset was composed of 7 road lane lines and lateral deviation of the vehicle on the road. The road was set as a straight highway going in one direction, as curvelinear coordinates were used, which neglects all curvature features of the road. Best overall results were achieved with the network model composed of Bi-LSTM and MSCNN layers, achieving accuracy of up to 97.2%, due to its ability to learn spatio-temporal features of input data.

References

1. Berger, C., Block, D., Hons, C., Kühnel, S., Leschke, A., Plotnikov, D., Rumpe, B.: Large-scale evaluation of an active safety algorithm with euroncap and us ncap scenarios in a virtual test environment – an industrial case study. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems. (2015) 2280–2286
2. Pütz, A., Zlocki, A., Bock, J., Eckstein, L.: System validation of highly automated vehicles with a database of relevant traffic scenarios. (2017)
3. ISO: 26262: Road Vehicles : Functional Safety, Geneva, Switzerland (2016)
4. Menzel, T., Bagschik, G., Maurer, M.: Scenarios for development, test and validation of automated vehicles (2018)
5. Wagatsuma, H.: Logical scenarios and coverage analyses enhanced by a representative trajectory model to reduce test cases to limited combinations. In: 2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS). (2018) 734–738
6. Ulbrich, S., Menzel, T., Reschka, A., Schuldt, F., Maurer, M.: Defining and substantiating the terms scene, situation, and scenario for automated driving. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems. (2015) 982–988
7. Menzel, T., Bagschik, G., Isensee, L., Schomburg, A., Maurer, M.: From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment. (06 2019) 2383–2390
8. Bach, J., Otten, S., Sax, E.: Model based scenario specification for development and test of automated driving functions. In: 2016 IEEE Intelligent Vehicles Symposium (IV). (2016) 1149–1155
9. Andreotti, E., Boyraz, P., Selpi, S.: Mathematical definitions of scene and scenario for analysis of automated driving systems in mixed-traffic simulations. IEEE Transactions on Intelligent Vehicles **6**(2) (2021) 366–375
10. Zhao, S., Wu, Y., Zhu, X., Zhou, B., Bao, H., Zhang, L.: Research on automatic classification for driving scenarios based on big data and ontology. In: 2018 IEEE 4th International Conference on Computer and Communications (ICCC). (2018) 1857–1862
11. Bagschik, G., Menzel, T., Maurer, M.: Ontology based scene creation for the development of automated vehicles. (06 2018)

12. Lages, U., Spencer, M., Katz, R.: Automatic scenario generation based on laser-scanner reference data and advanced offline processing. In: 2013 IEEE Intelligent Vehicles Symposium (IV). (2013) 153–155
13. Made, R.V.D., Tideman, M., Lages, U., Katz, R., Spencer, M.: Automated generation of virtual driving scenarios from test drive data. (2015)
14. Beglerovic, H., Schloemicher, T., Metzner, S., Horn, M.: Deep learning applied to scenario classification for lane-keep-assist systems. Applied Sciences **8** (12 2018) 1
15. Hartjen, L., Philipp, R., Schuldt, F., Howar, F., Friedrich, B.: Classification of driving maneuvers in urban traffic for parametrization of test scenarios. (11 2019)
16. Zhang, H., Nan, Z., Yang, T., Liu, Y., Zheng, N.: A driving behavior recognition model with bi-lstm and multi-scale cnn. In: 2020 IEEE Intelligent Vehicles Symposium (IV). (2020) 284–289
17. Woo, H., Ji, Y., Kono, H., Tamura, Y., Yamashita, A., Asama, H.: Detection method of lane change intentions in other drivers using hidden markov models. The Abstracts of the international conference on advanced mechatronics : toward evolutionary fusion of IT and mechatronics : ICAM **2015.6** (12 2015) 253–254
18. Hülnhagen, T., Dengler, I., Tamke, A., Dang, T., Breuel, G.: Maneuver recognition using probabilistic finite-state machines and fuzzy logic. In: 2010 IEEE Intelligent Vehicles Symposium. (2010) 65–70
19. Yu-Chen Lin, Ha-Ly Thi Nguyen, Cheng-Hsien Wang: Adaptive neuro-fuzzy predictive control for design of adaptive cruise control system. In: 2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC). (2017) 767–772
20. Sonka, A., Krauns, F., Henze, R., Küçükay, F., Katz, R., Lages, U.: Dual approach for maneuver classification in vehicle environment data. In: 2017 IEEE Intelligent Vehicles Symposium (IV). (2017) 97–102
21. Kim, J., Jo, K., Lim, W., Lee, M., Sunwoo, M.: Curvilinear-coordinate-based object and situation assessment for highly automated vehicles. IEEE Transactions on Intelligent Transportation Systems **16**(3) (2015) 1559–1575
22. dSPACE GMbH: dspace asm. https://www.dspace.com/en/inc/home/products (Accessed on 01/26/2021).
23. Khosroshahi, A., Ohn-Bar, E., Trivedi, M.M.: Surround vehicles trajectory analysis with recurrent neural networks. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). (2016) 2267–2272
24. Tamura, S., Tateishi, M.: Capabilities of a four-layered feedforward neural network: four layers versus three. IEEE Transactions on Neural Networks **8**(2) (1997) 251–255
25. Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, L., Wang, G., Cai, J., Chen, T.: Recent advances in convolutional neural networks (2017)
26. Donahue, J., Hendricks, L.A., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description (2016)
27. Krajewski, R., Bock, J., Kloeker, L., Eckstein, L.: The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). (2018) 2118–2125
28. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017)
29. Yeo, K.: Short note on the behavior of recurrent neural network for noisy dynamical system. CoRR **abs/1904.05158** (2019)