Proceedings of the 33rd European Safety and Reliability Conference (ESREL 2023)
Edited by Mário P. Brito, Terje Aven, Piero Baraldi, Marko Čepin and Enrico Zio
 Published by Research Publishing, Singapore.
doi: 10.3850/978-981-18-8071-1_P484-cd

Deep Behavioral Replication of Markov Models for Autonomous Cars using Neural Networks

Aishvarya Kumar Jain, Kushal Srivastava, Teo Puig Walz, Ivo Häring, Georg Vogelbacher, Fabian Höflinger, Jörg Finger

Fraunhofer Institute for High-Speed Dynamics, Ernst-Mach-Institut, EMI, Freiburg im Breisgau, Germany.
E-mail: aishvarya.kumar.jain@emi.fraunhofer.de

With autonomous vehicles on the brink of a revolution, there is an increased need for reliable individual driving functions and the overarching vehicle. Classical assessment techniques for such systems, which assume that future states depend only on the current state, include Failure Modes, Effects and Diagnostic Analysis (FMEDA), or classic Markov models, also recommended by IEC 61508, ISO 26262, and SOTIF ISO 21448. More realistic are memoryless approaches like the Monte Carlo simulation of Markov chains. However, these are computationally expensive. As an in-the-loop component to assess the safety of autonomous vehicles, Markov models essentially become the bottlenecks of the toolchain. In this context, trained neural networks are excellent tools to replicate the behavior of the Markov models rendering them an excellent in-the-loop component. To this end, the current work demonstrates that the deep learning models are capable of learning and generalizing the behavior of the Markov models.

Keywords: Autonomous Driving, Deep Learning, Markov chain, Safety of the Intended Functionality (SOTIF), Failure probability, Operational Failure

1. Introduction

The automotive industry is slowly diffusing from manual to autonomous vehicles. This paradigm shift is not only for the reduction of human error, but also for the efficient use of energy, reduced emissions, increase safety, and other economic benefits [1]. There is a significant investment in developing the technology. However, recent accidents involving driverless cars have influenced public opinion on the use of these vehicles. This motivates the need to assess the safety of autonomous vehicles both rigorously and in time.

Markov models have played a significant role in progressing the technology of autonomous driving (AD). For example, hidden markov models (HMM) are used to make accurate estimations of the driver state during pre-crash scenarios [2]. Further applications include assessing risk and uncertainty, fault detection and diagnostics [3], and human-machine interactions.

One such methodology involves combining Markov models with Monte Carlo methods to identify vulnerabilities, and failure scenarios using stochastic simulations. As explained in Section 3, the method starts with an initial state definition of the system, and the Markov model is executed to predict the failure probabilities. For such a method, the computation of huge Markov models with states in the order of thousands could essentially become the bottleneck of the toolchain. This motivates us to look for computationally cheaper alternatives to the Markov process.

To address this problem, deep neural networks are introduced [4, 5]. Neural networks (NN) represent an alternative computational paradigm in which the solution to a problem is learned from a set of examples. NN provides a set of techniques for solving problems spanning pattern recognition, control, and data analysis [6]. The advantages include high processing speed and the ability to learn even from very complex sets of patterns which can be missed in manual pattern matching. For NN, training is the most computationally expensive step. However, once trained, the network can process new data rapidly. The scope of the current work is to establish a deep learning method as an alternative to the Markov process for component failure analysis in modern AD vehicles.

The approach is to use an existing Markov model of a given number of states to generate the training data for the NN. Once trained, the NN emulates the Markov chain and is able to predict the correct state probability distribution for a given transition matrix, which is the output of a converged Markov model.

The paper is further divided into five sections. Section 2 summarizes the use of Markov models in AD and the application of NN in similar problems. Section 3 briefly explains the approach of training a NN with Markov model state transition matrices and state probabilities. Section 4 presents the Markov process set-up along with the specifics about the input and output data needed for the training of NN as well as NN design details. Section 5 discusses the results of training and convergence of NN predictions and compares them with the results of the Markov model. Finally, Section 6 summarizes the paper and discusses the future scope of the work.

2. Related Work

As the Industry progresses, there is an increase in the extent of dependency on electronic components in automotive vehicles. ISO 26262 is a standard that deals with the functional safety of the E/E (Electric and Electronic) components of a road vehicle [7]. Markov models have been widely studied for this application. For example, in [8], a generic Markov model is proposed for electric and electronic systems. In [3], the authors use semi-Markov processes to analyze the safety of AD vehicles. The paper [2] uses Hidden Markov Models for driver behavior near road intersections. Markov-model-like methods have been used to model the safety of AD functions [7].

Machine learning on the other hand has also been widely applied in the domain of autonomous vehicles. In [9], the authors summarize the technologies and different aspects of deep learning used in autonomous vehicles. This includes perceptrons, convolutional neural networks (CNN), long-short term memory (LSTM) based models, and reinforcement learning-based models. CNNs and LSTMs have been used to solve complex problems like steering angle and velocity control, recurrent NN for lane keeping, and obstacle avoidance [10].

Having mentioned the above methods, there is limited work being done in the direction of emulating the behavior of Markov models. In the current work, we are presenting an approach to learning the behavior of Markov models using a multi-layer perceptron-based NN for functional safety analysis of AD vehicles. With this motivation, the current paper tries to emulate a converged Markov process using NN.

3. Methodology

A continuous-time Markov chain (CTMC) is essentially a coalition for several discrete-time Markov chains with varying transition rate parameters. Fig. 1 shows a simplified model of a two-state non-homogeneous CTMC. For a given state-space arrangement, the input is a time-dependent state transition matrix $Q(t)$ and the output is the state space distribution, i.e. $P(t)$, essentially registering the probability to be in each state for each time step. If the time axis is decoupled the model output represents the state distribution for a given transition matrix.

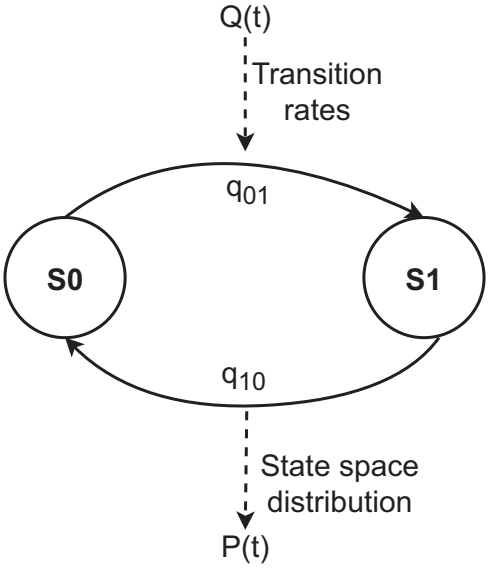


Fig. 1.: Model of a two-state CTMC with two possible transitions.

Keeping this idea in mind the paper proposes a deep learning architecture based on multilayer perceptron trained on the sparse data generated using the Monte Carlo Markov chains (MCMC). As shown in Fig. 2, the input to this network is the transition matrix and the output is the state space distribution.

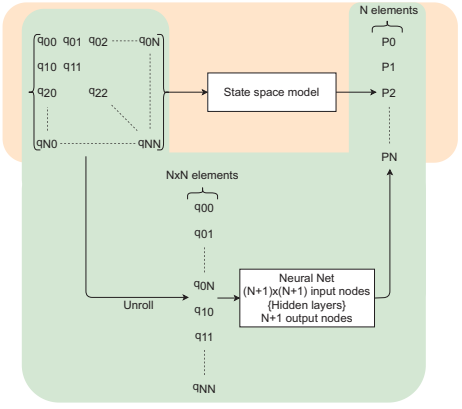
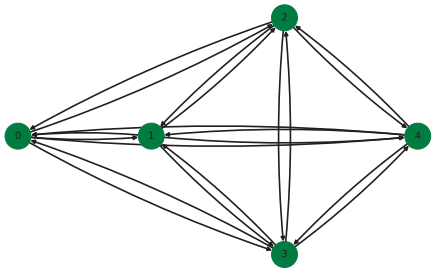


Fig. 2.: Methodology diagram showing the resemblance of the two approaches. The orange highlight shows the Markov approach and the green highlight shows the Deep learning approach.

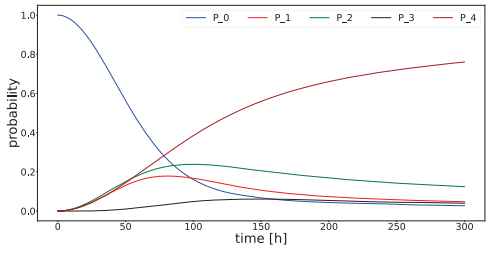
In order to generate the training data, random discrete MCMC simulations are executed. Each simulation corresponds to one data point. For huge networks with states in the order of 1000, data generation is resource intensive. In the NN approach, it is a pre-deployment step and can be completed independently of other in-the-loop components. Once the NN is trained, which is another resource-intensive phase but completed in the pre-deployment phase, the state distribution can be estimated with a single feed-forward step.

4. Setup

For the current study, we propose a state space model with five states as shown in Fig. 3a. The plot in Fig. 3b shows the evolution of state distribution over time for a randomly generated initial transition matrix. Since we are working with time-dependent transition rates, the subsequent transition matrices for each time step are derived using



(a) State space model with 5-states.



(b) Time evolution of the states.

Fig. 3.: State space model with 5 states and a sample time evolution of probability distributions for a randomly generated transition matrix. The transition rates are updated using Weibull distribution.

a Weibull hazard function (see Eq. (1)) with constant shape parameter $\alpha = 2$ and scale parameters λ_{ij} with q_{ij} being the Q-matrix entries:

$$q_{ij}(t) = \alpha\lambda_{ij}(\lambda_{ij}t)^{\alpha-1}, i \neq j, i, j = 0, 1, ..., N \tag{1}$$

The multilayer perceptron used in this article has 25 feature nodes corresponding to each element of the transition matrix and five output nodes that correspond to the steady state probability for a given transition matrix. The overall configuration of the NN is shown in Fig. 4. It has four hidden layers all using the sigmoid activation function [11]. All features, i.e. the transition rates, are normalized before feeding to the network. The final layer consists of the softmax activation function [11] which normalizes the output to a probability distribution.

The configuration and structure of multilayer perceptron greatly decides the accuracy of the predictions. This article focuses on the approach

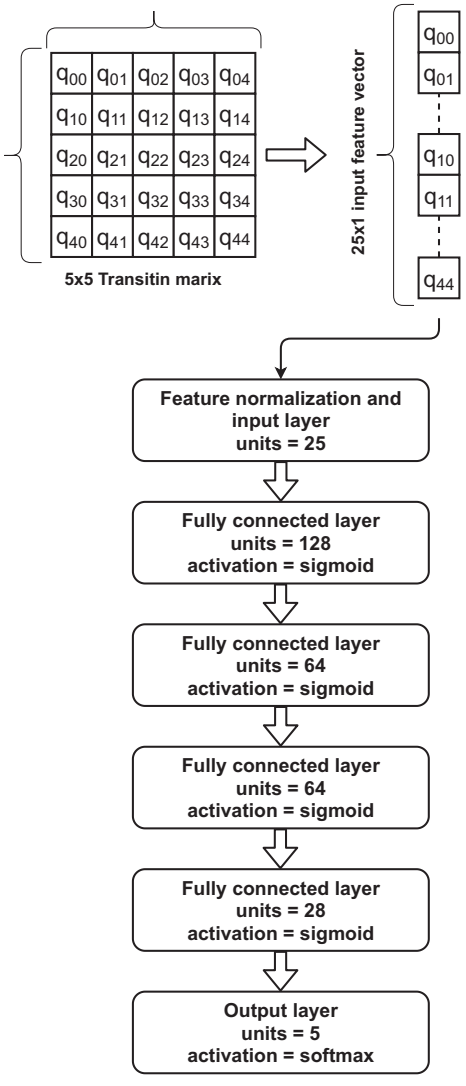


Fig. 4.: Architecture of deep NN with layer configurations.

of behavior replication of the Markov chain within an acceptable error margin. Further studies are needed on hyperparameter optimization to achieve precise state estimations.

5. Results and Discussion

Using the Markov model, a total of 240800 data points were generated which are subsequently used for training, validation, and testing of the NN. For testing post-training and validation, a total of 10% data was kept separate. The remaining data is used for training and cross-validation with a 30% cross-validation split.

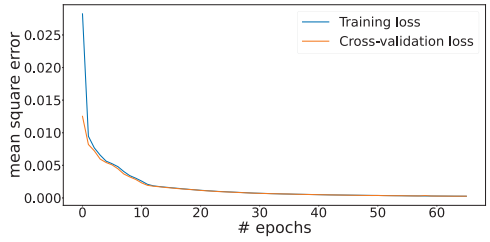


Fig. 5.: Learning curve showing the evolution and corresponding reduction of training and cross-validation mean squared error over the number of epochs.

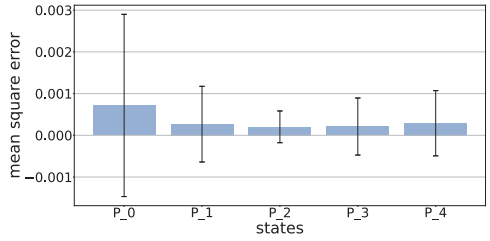
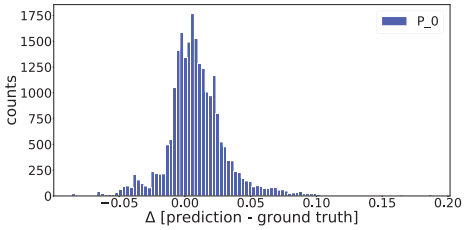


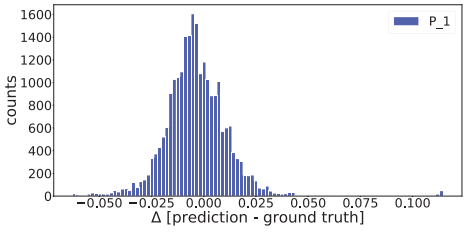
Fig. 6.: Mean squared errors (MSEs) estimations of the predictions with respect to the ground truths for each state of the Markov model shown in Fig. 3a. Blue bars show the mean of MSEs and the black limiters show the distribution (standard deviation) around the means.

The NN was trained for 65 epochs with a batch size of 1000 data points. To prevent overfitting, we used early stopping with an early stopping patience of 2 epochs. As seen in Fig. 5, training error and correspondingly cross-validation error reduce over the number of epochs depicting that the network is able to find and learn the pattern in the data.

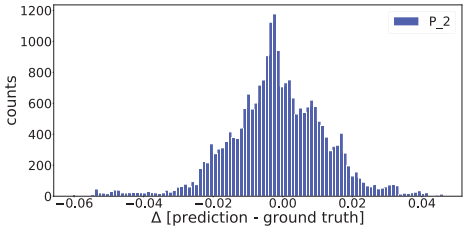
The trained model is then used to predict estimations on the test dataset. Fig. 6 plots the mean squared error (MSE) of the probability distribution of all the five states of the introduced Markov model. It shows that the average MSE of all states is under 1E-03.



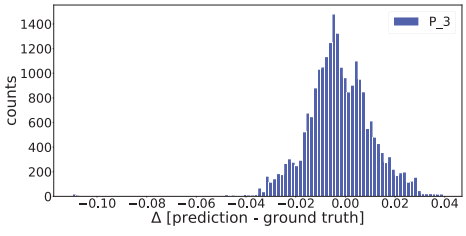
(a)



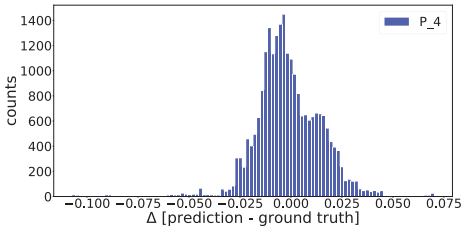
(b)



(c)



(d)

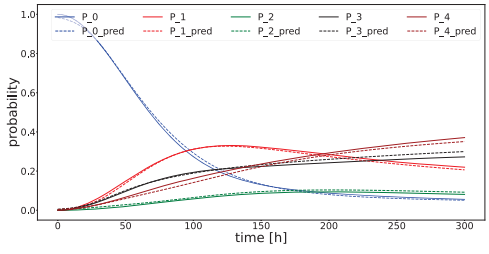


(e)

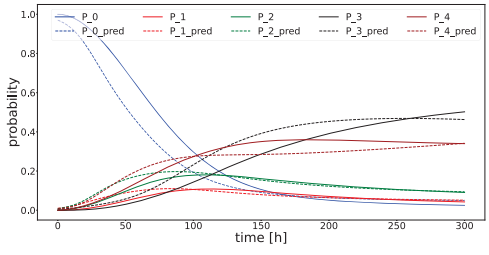
Fig. 7.: Histogram of error values (Δ) between predicted state probabilities and the corresponding ground truth.

Fig. 7 plots the distribution of error values (Δ) which is the difference between predicted state probabilities and the corresponding ground truth. The distribution of all the error points for each state almost follows a Gaussian distribution with the mean centered close to zero. These distributions show the accuracy of the presented approach to replicate the behavior of Markov chains using NN. Although in this article such distributions are not used, a potential application is to use them in a post-processing stage to increase the prediction accuracy.

Fig. 8 shows an end-to-end application of the documented approach to replicate the behavior of the continuous-time Markov chains for the state probability estimation of the state space model shown in Fig. 3a.



(a)



(b)

Fig. 8.: An end-to-end estimation of state evolution of the state space model presented in Fig. 3a using deep learning model shown in Fig. 4 replicating the process of continuous-time Markov chains.

Fig. 8a presents the results with a very slight deviation from the ground truth. On the other hand Fig. 8b shows that the result could also drift far away from the ground truth indicating a need for

hyperparameter optimization of the deep learning model.

6. Conclusion and Outlook

In this work, we presented an approach to replicate the behavior of CTMCs using deep learning, specifically a multilayer perceptron. To generate the data for the training of the NN, we used a state space model with five states where transition matrices, i.e. interconnections, were generated randomly.

We show that deep learning methods can efficiently learn the pattern corresponding to the Markov chains and are able to generate probability distributions of states within acceptable margin error. Using this approach, we are capable to replace the in-the-loop components where Markov chain simulations using traditional approaches can be a bottleneck. The application of the approach is not only limited to the autonomous vehicles but can be extended to other domains where Markov chains are applied intensively, for example, in component failure analysis of power and manufacturing industries.

Although the approach is successful, there is still more work to be done. Hyperparameter optimization of the deep learning model along with imposing some initial conditions can further increase prediction accuracy. Another attempt would be to use error distributions to further refine the predictions in a post-processing step.

Acknowledgement

The presented work is funded by the German joint BMWE project on Real Driving Validation (RDV).
[Grant number 19A21051D, 2022-2025]

References

1. A. Z. Benleulmi and B. Ramdani, "Behavioural intention to use fully autonomous vehicles: Instrumental, symbolic, and affective motives," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 86, pp. 226–237, 2022.
2. S. B. Amsalu and A. Homaifar, "Driver behavior modeling near intersections using hidden markov model based on genetic algorithm," in *2016 IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, pp. 193–200, 2016.
3. M. Nyberg, *Safety analysis of autonomous driving using semi-Markov processes*, pp. 781–788. 06 2018.
4. M. Gardner and S. Dorling, "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences," *Atmospheric Environment*, vol. 32, no. 14, pp. 2627–2636, 1998.
5. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.
6. C. M. Bishop, "Neural networks and their applications," *Review of Scientific Instruments*, vol. 65, pp. 1803–1832, 06 1994.
7. I. Häring, Y. Satsrisakul, J. Finger, G. Vogelbacher, C. Köpke, F. Höflinger, and P. Gelhausen, "Advanced markov modeling and simulation for safety analysis of autonomous driving functions up to sae 5 for development, approval and main inspection," in *32nd European Safety and Reliability Conference (ESREL 2022)*, pp. 104–111, 2022.
8. "Modeling automotive safety mechanisms: A markovian approach," *Reliability Engineering System Safety*, vol. 130, pp. 42–49, 2014.
9. S. Wang, D. Jia, and X. Weng, "Deep reinforcement learning for autonomous driving," 2019.
10. S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.
11. S. Sharma, S. Sharma, and A. Athaiya, "Activation functions in neural networks," *Towards Data Sci*, vol. 6, no. 12, pp. 310–316, 2017.