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

Estimating the Uniqueness of Test Scenarios derived from Recorded Real-World-Driving-Data using Autoencoders

Jacob Langner[1], Johannes Bach[1], Lennart Ries[1], Stefan Otten[1], Marc Holzäpfel[2] and Eric Sax[1]

Abstract— **Advanced Driver Assistant Systems (ADAS) use a multitude of input signals for tasks like trajectory planning and control of vehicle dynamics provided by a large variety of information sources such as sensors and digital maps. To assure the feature's valid behavior all realistically possible environmental situations have to be tested. The test scenarios used for simulation can be derived from real-world-driving-data. However, the significance of derived scenarios is weakened by repetitive similar situations within the driving data, which increase the test efforts without providing new insights regarding the test of the ADAS. In this contribution, an automated selection algorithm for test scenarios based on relevant environmental parameters is presented. Starting with a randomly selected initial testset, the machine-learning concept of autoencoders is utilized to recognize novel scenarios within the data pool, which are iteratively added to the initial testset. Furthermore, the key parameters for the autoencoder's performance are shown in depths. The approach is fully automated, so that the identified novel scenarios within an entire testset are automatically combined to a reduced testset of unique relevant scenarios. The achieved testset reduction and thereby the saving potential in simulation time is demonstrated on a dataset including several thousand test kilometers.**

I. INTRODUCTION

Key innovations in the automotive domain include ADAS and increasingly autonomous features, like valet parking, highway assist or traffic jam pilot. These new features are based upon a deeper understanding of the vehicle's environment, which is enabled by a rising number of cameras, sensors and the integration of additional information sources, like digital maps or car-to-x communication [1]. This deeper understanding enables a better interaction of the feature with its environment but also leads to more complex algorithms and features. With the increasing number of features and as a consequence thereof the increasing interconnections and inter-dependencies between these features current and upcoming vehicles have become vastly complex systems.

More complex systems are more difficult to verify and validate, which leads to a staggering increase in time and costs during the Verification and Validation (V&V) phases. In order to reduce these expenses, V&V activities are broken down according to the system decomposition into separate stages [2]. While the complexity of tests on unit and component level is manageable, V&V on system level remains a challenge. The complete event chain has to be considered with all its relevant signals and internal states. Time-dependencies play a huge role in V&V complexity as well. Proving the correct integration and interoperability of a complex automotive feature is extremely difficult and is usually covered by excessive testing. To define and know how much is sufficient test coverage is still an open research question.

A common approach for system level V&V is to test the system in its future environment - the real world. These tests enable a good preview of the feature and have high realism and validity. However, testing in the real world is expensive, resources are limitedly available and the test environment and test conditions are not fully controllable, leading to low reproducibility.

Real-world tests thereby need to be accompanied by exhaustive simulation-based tests. Simulations are substantially cheaper and real-time constraints do not apply. However, in order to achieve sufficient realism in simulations, they have to be supplied with detailed models for the simulation environment, such as the vehicle, the driver and other components and features, that are not under test [3]. These modeling efforts are not negligible and achieving sufficient realism in a simulation environment is difficult.

Furthermore, in order to achieve significant test coverage, a huge amount of test scenarios is required. Since manual scenario specification is oftentimes not feasible, scenario derivation has to be considered. One solution is to generate test scenarios with the help of statistical models. Knowledge about the frequency distributions can be utilized to generate realistic testsets [4] [5]. Another approach is to use real-world-driving-data. Data recorded during real world tests can be utilized in simulations [6].

For both approaches, redundancy in the test data is a huge problem. Redundant test scenarios do not offer new insights but increase simulation time. However, identifying redundant test data in the complete data pool is difficult, since it depends on the test item and which data is relevant and offers new test content. In this contribution we refine our testset selection approach [7] to minimize redundancies during testset creation with the help of autoencoders.

In section II we give a brief overview about simulation in automotive V&V, followed by the concept for our approach in section III and its prototypical implementation in section IV. The approach is evaluated in section V. Conclusions are drawn in section VI where an outlook on future work is given as well.

[1]Jacob Langner, Johannes Bach, Lennart Ries, Stefan Otten and Eric Sax are with FZI Research Center for Information Technology, 76131 Karlsruhe, Germany langner@fzi.de, bach@fzi.de, ries@fzi.de, otten@fzi.de, sax@fzi.de

[2]Marc Holzäpfel is with Dr. Ing. h.c. F. Porsche AG, 71287 Weissach, Germany marc.holzaepfel@porsche.de

II. STATE-OF-THE-ART

A common approach in Systems Engineering to reduce the system complexity is its decomposition into smaller parts. In Automotive Systems Engineering (ASE) a system is usually decomposed into subsystems, components and units [8]. The complexity is reduced with each decomposition. This is also utilized during V&V with different testing methods for each decomposition layer. Units and components usually have a small number of input and output variables with fixed value ranges, which can be generically tested in so called unit and component tests. Equivalence classes may be used to reduce the number of test cases.

However, the correct integration and interaction of all parts of a system still have to be tested on system level, where the system itself is typically considered a black-box and the focus lies on the externally observable behavior. Systems can be tested according to a definition of Boehm. He considers "'a software program as a mapping from a space of inputs into a space of outputs"' [9]. Systems usually have a whole number of different input signals. Event chains and temporal progressions have to be considered as well, since automotive features are complex and non-deterministic systems and therefore may react differently to the same input depending on the internal system state [10]. Consequently, testing on system level is an intricate and comprehensive task. Ensuring a correct interoperability and integration of units and components is difficult.

One possibility for system level V&V is to test the feature in the real world. With the help of prototype vehicles the feature can be tested in its future environment with the highest possible realism. These tests have a high validity and are vital for the V&V process. However, real-world test drives do not scale well. ADAS and upcoming autonomous features are entailed by an enormous growth in the necessary test coverage, which increases both time and cost for V&V. Test coverage is the proportion of different tested scenarios to all conceivable scenarios. It is usually impossible to guarantee complete test coverage, since the real world falls under the open world assumption and therefore the conclusion, whether all possible scenarios have been tested and whether untested scenarios would be tested successfully, is impossible. Additionally, each new release of a feature needs to be verified, which poses the problem of reproducibility of real-world tests.

Therefore, simulation-based approaches are an essential component of today's V&V strategies in automotive feature development. The cost-effective and reproducible nature of the simulation complements real-world-test-drives and prototypes perfectly. However, simulations must be carefully parametrized. In addition to extensive models for the system environment, the vehicle, the driver and other components, all driving scenarios must also be provided. This results in large manual efforts to create the simulation environment. One approach is X-in-the-loop (XiL) testing, in which the simulation environment is reused throughout all test phases. This not only reduces modeling efforts, but also guarantees consistent simulation models throughout the entire development process. Function models are tested in early phases using Model-in-the-loop (MIL) tests. During and after implementation, the software code can be validated in Software-in-the-loop (SIL) tests. In later V&V phases, the hardware integration is tested using Hardware-in-the-loop (HIL) tests [11].

A common method is to define test scenarios [12], that contain all relevant test cases for the test item. If all scenarios are passed, the feature is considered as validated. However, manual scenario specification results in large efforts in order to create a varying set of scenarios and to ensure their sufficient realism. A newer approach, especially for autonomous vehicles, is to continuously test the feature without specific scenarios and simply observe its behavior [13]. This allows for better test coverage, since not every scenario has to be specified, but rather occurs in the test stream. Though, even more comprehensive models have to be supplied for these tests and they become even more time-consuming.

Another solution is the generation of test scenarios. Generating all possible test scenarios quickly leads to unfeasible testing efforts. Different approaches to reduce the number of generated test scenarios have been proposed. Schuld et al. propose a method using statistical considerations to generate safety relevant test cases for a black-box system [14]. Glauner [15] and Eckstein [16] focus on critical situations, which are detected during real world tests. It is argued, that less critical situations are covered by more critical situations and thereby can be skipped.

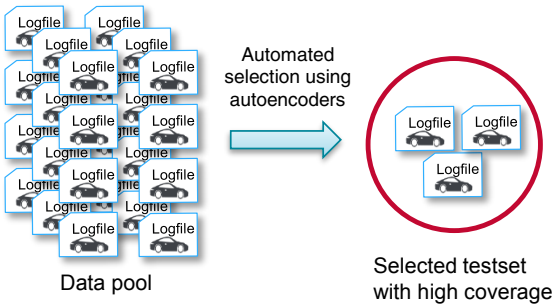


Fig. 1. Illustration of the presented approach to automatically perform the testset selection.

A third option is to derive test scenarios from real-world-driving-data [17] [18]. By logging real-world-driving-data tens of thousands of realistic test kilometers can be generated. With the help of data loggers the messages and signals on the vehicle's bus systems can be recorded and then be reused for simulations, alleviating the need for manual model specification for the simulation. With the gained realism even the virtual execution of geolocation-dependent features becomes reasonable. Additionally, a lot of data can be acquired passively and cost-neutral throughout test campaigns, release approvals and even permanently used vehicles.

The approach however suffers under enormous redundancies in the test data. For crash testing a recent analysis by

Wachenfeld and Winner states, that only every 210 million km a fatal accident occurs [3]. While ADAS tests do not rely on crash data, the problem remains the same. During real-world-tests there are a lot of 'boring', 'normal' driving scenes, where nothing 'out of the order' is happening. Only every now and then an interesting and distinctive event occurs. In order for simulation efforts to remain within reason with an ever growing data pool, methods to reduce the test efforts while not loosing any test coverage are needed.

In previous work we designed a two-step-approach to derive a reduced testset out of the complete data pool [7]. In the first step we utilized a classification tree to separate test-drives by mostly binary or low-dimensional meta data, e.g. the country-of-origin, the model of the test vehicle, the street type, etc. In the second step the relevant input space was analyzed. Applying Boehm's definition, we selected test-drives by looking at their value ranges. By comparing the 2-dimensional cross-parameter-coverage we even considered correlations and dependencies between different input parameters. After reducing the data pool to relevant scenarios in step 1, we iteratively selected the scenarios, that added new cross-parameter-coverage for the final testset. While this approach delivers promising results, we looked at other approaches as well. In this contribution we present an alternative method for step 2 using an autoencoder or replicator neural network in order to automatically detect new scenes within the real-world-driving-data. The method is illustrated in Fig. 1.

An autoencoder is a neural network, that is trained to compress a set of input values and then reconstruct the original data from the compressed set. This is done by reducing the number of neurons per layer in the inner layers of the autoencoder. The network topology is mirrored at the inner most layer, so that each compression layer has an equivalent decompression layer. Autoencoders are commonly used for information compression [19], reduction of data dimensions [20] and more recently for anomaly detection [21] [22]. The idea behind the anomaly detection approach is, that the autoencoder can reconstruct normal, expected values better than anomalies, if trained on normal data. An error for the quality of the reproduction can be calculated. Therefore, values with a high reproduction error can be interpreted as anomalies.

In this contribution we present the utilization of an autoencoder for detecting new scenes within a pool of real-world-test-drives.

III. CONCEPT

In the context of the real-world-driving-data-based simulation [23] during the implementation phase, each iteration of the developed software is tested and debugged with the help of the real-world-driving-data. The complete data pool of recorded test drives offers a huge variety of different driving scenarios and environmental scenes. For the purpose of testing, a scenario is a sequence of interactions of the test item with its environment [24], in our case a different set of relevant inputs. Using all available test drives during simulation results in the best possible test coverage. In order to keep iteration times short during the implementation phase, the complete data pool has to be reduced. A small but decisive testset has to be created which offers the maximum possible test coverage and at the same time the minimum possible test scope. For this purpose, an autoencoder shall be used, which derives a testset tailored to the test item.

For an analysis of the test coverage, all possible cross-parameter-value-combinations of the measured data have to be considered. Due to internal states of the feature considering all possible combinations of input signals alone is not sufficient. The signals' temporal behavior must also be taken into account.

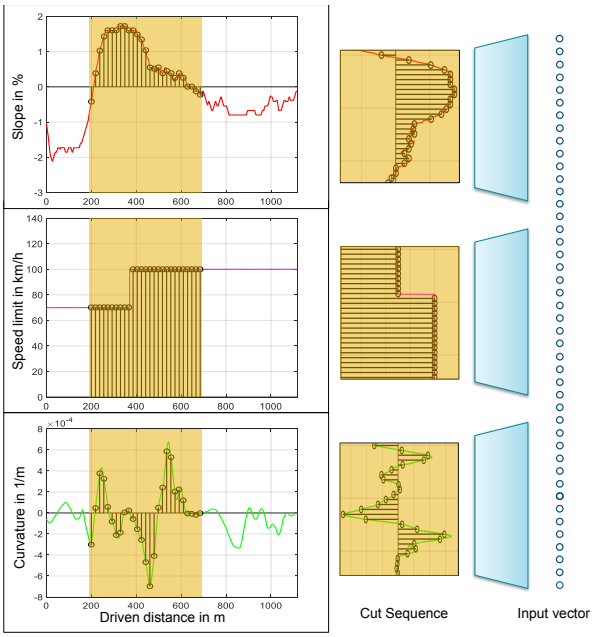


Fig. 2. Flattening of temporal progressions, so that they can be inserted into the neural network at once

Autoencoder networks, however, do not support temporal behavior. They do not have an internal memory and therefore each input vector is evaluated independently. A possible solution is to create an input vector that in itself represents the temporal behavior by inserting not only one data point per signal, but a time-series vector for each signal. As Fig. 2 shows, each signal is inserted into the neural network with several sampling points so that it can also learn and understand temporal or position-based sequences. Position-based, as the measured data is dependent on the time and therefore differs with different velocities. If the signals relevant for the test item are not supposed to be time-dependent, this distortion can be removed by converting the time reference to the track position. E.g. the same road characteristics look different to the autoencoder, if measured at different velocities and sampled by their time reference.

Since feeding an entire test drive at once into the autoencoder requires either a lot of input neurons or very few sampling points, the data is cut into sequences. The length of each sequence does not affect the autoencoder's

ability to reconstruct a given input and therefore is a design parameter, which can be freely chosen depending on the requirements of the test item. The number of sampling points and the length of the sequence must be suitable to represent the relevant road characteristics, like curves and slopes, or driving maneuvers. In order not to miss any road characteristic the consecutive sequences are overlapping. Each sequence contains the same number of sampling points for each selected signal. These represent the relevant information for the test item. The sampling points are put into a vector and then fed into the neural network. The autoencoder then learns to compress and reconstruct this vector at the output neurons. The output vector can then be compared to the input vector and the difference for each sampling point can be calculated. Several methods can be used to calculate an aggregated error. For this autoencoder the Root-Mean-Square Error (RMSE) [25] is used. The higher the RMSE the poorer the reproduction - and as we conclude in section V the more novel the sequence. An average RMSE for the test drive is calculated after all sequences have been fed through the autoencoder.

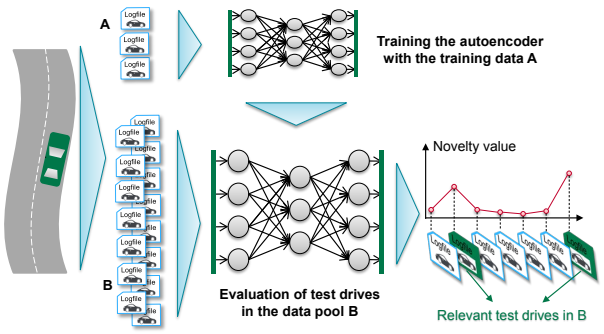


Fig. 3. Training and evaluation of test drives during the testset selection

Fig. 3 depicts the testset selection approach. In the first step, the autoencoder is trained with the training data A, the initial testset. The training ends when the autoencoder is able to replicate the training data sufficiently well. The second step follows, where the trained autoencoder can now be applied to the remaining test drives in the data pool B. For each test drive, the reproduction error is calculated using RMSE. High errors indicate poor reproduction of the test drive, which in turn means that the test drive is less well known to the autoencoder - in other words, it is a novel test drive. Iteratively, the worst reproduced test drives are added to the initial testset and the autoencoder is retrained with the new testset. This procedure can be repeated up to a certain size of the testset or a certain reproduction quality for the left-over test drives in the data pool.

Since the evaluation of the novelty is strongly dependent on the initial testset, the entire process is executed several times with randomized inital testsets. A score is calculated for each test drive in each cycle, which is derived from the iteration in which the drive was added to the testset. After all cycles have been processed the mean score for each test drive is calculated. The smaller the score, the more distinctive or novel the drive.

Since the test drives are of different lengths and the novelty value for each test drive is averaged over all sequences of the test drive, it is less expressive over an entire test drive. High novelty values for a few sequences can be blurred by many sequences with an average or low novelty value. In a first step the mean reproduction error was replaced by the maximum reproduction error for each test drive. For further optimization, the concept of complete test drives was abandoned completely and the concept of tracks was introduced. A track consists of at least one consecutive sequence. Tracks for the testset are chosen by their novelty value from all reproduced sequences. Each sequence consists of a 300 meters long section of the test drive, which the neural network already evaluates independently of the preceding and following sequences of the test drive. Therefore, the evaluation algorithm was switched from evaluating entire test drives to single sequences. Consecutive sequences were combined to longer tracks. Thus, the novelty value becomes independent of the length of the test drive and as the evaluation will show later on, the redundancies in the testset are further reduced.

IV. PROTOTYPICAL IMPLEMENTATION

The method was implemented as a prototype for the Predictive Cruise Control (PCC) feature, which predicts an optimal trajectory and respectively controls the velocity for the current situation. Thus, the feature not only reacts to upfront vehicles but the road topology as well, considering for example speed limits posed by curves or traffic signs. Therefore, the PCC feature has to be tested with regard to the signals describing the different road characteristics and the upfront vehicle. In order to create a reduced testset for the PCC tests during implementation, these signals were also considered for the autoencoder, since the testset should be created with regard to the signals relevant for the feature under test. For the prototypical implementation we reduced the number of signals by introducing an abstraction layer. The signals, that describe the upfront vehicle, were reduced to one signal, stating the existence of an upfront vehicle. For the road characteristics we focused on an abstract description of the environment with the help of the curvature, slope, speed limit as well as the urbanity of the surrounding area.

In a first step, the autoencoder was trained with the selected signals over the entire data pool in order to validate that the approach works at all and to optimize the parametrization of the neural network. Part of the implementation of an autoencoder is the determination of its parameters, which play a key role in the ability to reduce and then reconstruct a given input vector. The parameters were systematically optimized so that an optimal configuration of the network for the reproduction of PCC input variables was given.

We distinguished between network parameters, which control for example the network's size or behavior, and data parameters, which influence the preparation of the

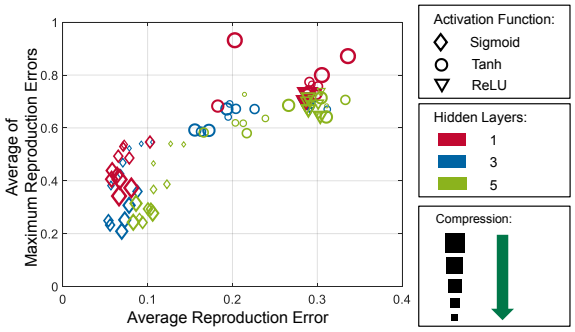
data before it is inserted into the autoencoder. Both classes of parameters influence the reproduction capability of the network:

- network parameters
  - the neuron's activation function
  - the number of hidden layers
  - the number of inner neurons
  - the learning rate of the neural network
  - the batch-size
  - the number of epochs
- data parameters
  - the number of sampling points per signal
  - the distance between the sampling points
  - the size and step-size of the sliding window over the complete test drive
  - the choice of signal smoothing
  - normalization of the signals to ensure equal influence

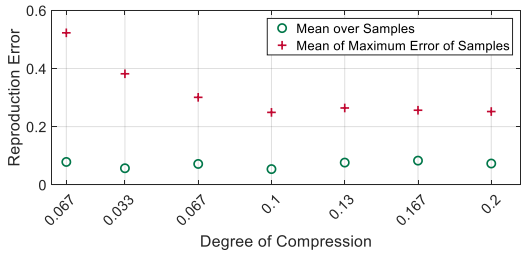
Fig. 4 shows the evaluation of different network parameters. The influence of these parameters was empirically determined. The sigmoid activation function and a network topology with 3 hidden layers and 75 neurons in the innermost layer were chosen. The learning rate was set to $\eta = 0,001$ as it rapidly approximates the global optimum with less computational power required than the learning rate $\eta = 0,0001$, which delivers a slightly more stable approximation of the global optimum. The choice for batch-size and number of epochs mainly depends on the available computing power and time constraints. They were chosen, so that the iterative creation of a novel testset out of the complete data pool took about 8 to 10 hours.

For the data parameters, normalizing the data was the most important step to assure equal representation of each input value in the calculation of the novelty value. And, as already mentioned, the length of the cut sequences had to be chosen carefully in order to calculate a suitable novelty value. If these are cut too small, road characteristics can no longer be seen in the sequence. If, on the other hand, the selected sequences are too long, characteristic areas in the sample may become blurred by the normal areas in front of and behind the relevant area. The number of sampling points must also be taken into account when cutting the sequences, since there is the risk of under-sampling the road characteristics, like sharp turns. 300 meter long sequences, sampled every 2 meters, proved to be a good parameter choice to represent the relevant road characteristics.

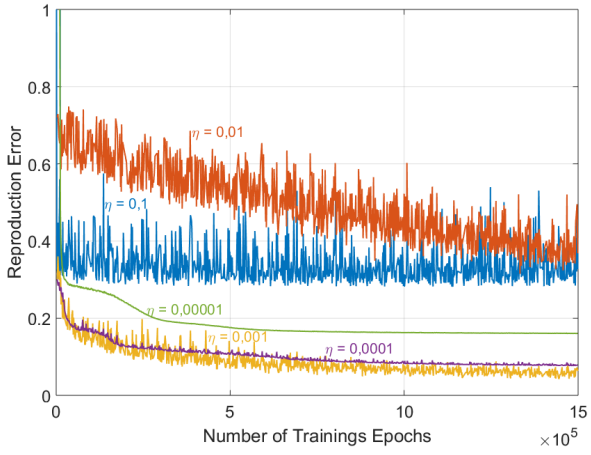
With all network and data parameters determined, in the second step only a small subset of the complete data pool was used for the training phase. The training was stopped when the reproduction of the training data was sufficiently good, in this case when the reproduction error for the training data was below 0,08 (see Fig. 4). Now the trained neural network was applied to the rest of the test drives and the average as well as the maximum reproduction error per test drive were calculated, where the average reproduction error is the average of the reproduction errors of all sequences of



(a) Activation function, number of hidden layers and compression



(b) Compression



(c) Learning rate η

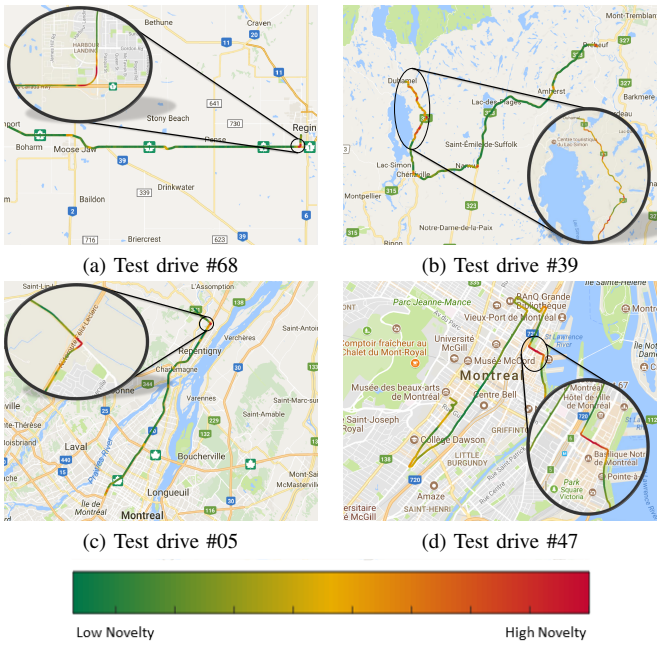
Fig. 4. Empirical evaluation of different neural network parameters

a test drive. However, since the maximum reproduction error is more meaningful in regards to novelty inside the test drive and is also independent of the length of the test drive, a fixed number of test drives with the largest maximum reproduction error were added to the training set in each iteration step. The network is then retrained and the next iteration begins or, if one of the exit conditions is satisfied, the algorithm stops and the final testset is returned.

V. RESULTS AND EVALUATION

Fig. 5 depicts the top 4 test drives, which were selected with the test drive-based approach using the maximum reproduction error. The yellow and red coloring indicates sequences with high novelty scores. Each test drive has at least one sequence with a high novelty value colored in red. For some sequences the reason for the high novelty value can be understood by looking at the map, e.g. at

sharp turns or distinctive curves. Other sequences however are straightforward roads with no clear indication for the high novelty value. Looking at the other signals gives more indication as to why the sequence was selected, e.g. the road to Duhamel in test drive #39, shown in Fig. 5, is out of town but still has a speed limit of 50 km/h - an uncommon parameter combination poorly reproduced by the autoencoder.



(a) Test drive #68 (b) Test drive #39

(c) Test drive #05 (d) Test drive #47

Fig. 5. Top 4 test drives for the test drive-based selection approach

As already described, in order to rate the test drives by novelty the maximum reproduction error is more suitable than the average error over the entire test drive. Novel sequences in longer test drives become blurred when using the average error. However, the maximum error only indicates the existence of at least one novel sequence. It does not indicate how many novel sequences there are in each test drive. We analyzed the selected drives with a high maximum error and were able to find new sequences in every drive. However, every drive also had a lot of similar sequences which with our drive-based approach also ended up in the final testset.

A consequent step is therefore not to include the complete drives in the testset, but to cut the entire data pool into test sequences of equal length and to evaluate them for novelty independently. This is a valid option, since the simulation does not depend on complete test drives but can also be started with smaller test sequences from the data. It only requires a small head start to reconstruct the digital map. To adapt the approach for sequence-based testset selection, some small adjustments had to be made. First and foremost all test drives in the data had to be cut into sequences, which then can be randomly chosen for the initial training independent of their corresponding test drive. Since the test drives already had to be cut into sequences for the drive-based approach, this was only a small change. The autoencoder then had to be trained with independent sequences instead of complete test drives. Some parameters had to be changed, e. g. the batch-size and the size of the initial training set. The score system had to be adapted slightly, since it is not suitable for the huge number of files. Now, only the first x sequences in each iteration get points according to their reproduction error. Therefore, a higher score stands for a higher novelty value.

Fig. 6 shows the final novelty score per sequence for one test drive. The longest part of the test drive is redundant, well trained data, which was never selected during all iterations and thus has zero score points. But, there are also a few quite novel sequences. With the sequence-based approach only these few sequences are selected and unlike the drive-based approach all redundant data is not automatically added to the testset. Continuously selected sequences are automatically concatenated to what we call tracks. Therefore, from the test drive 7 tracks would be derived for the final testset, which are marked in the map. The high score in the special curve seems plausible, which leads to the problem with machine learning approaches.

There is no way known to the authors to prove, that the calculated novelty value is justified. At this point we can not prove, that there are no other sequences alike in the test data. We also can not prove, that every novel sequence is detected by the autoencoder. However, we evaluated exemplary sequences and the novelty value, either high or low, always was justifiable. Therefore, we argue, that this fully automatic

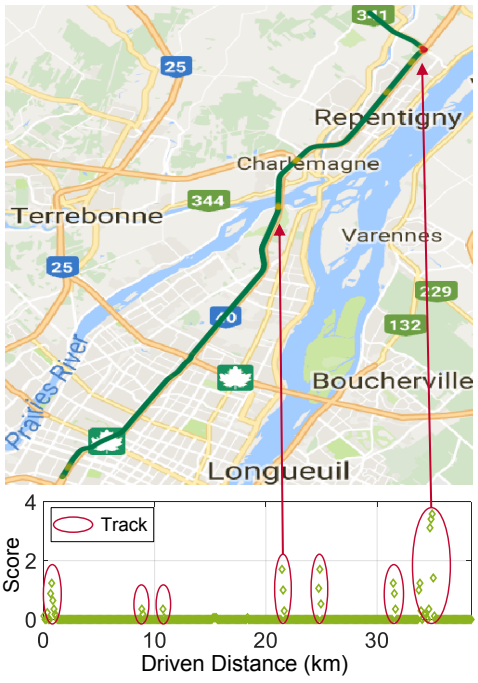


Fig. 6. Novelty score for each sequence in a test drive

approach might not result in the perfect testset, if that even exists, but massively reduces redundancies and still selects interesting and novel sequences from the test data.

The results show that both the evaluation of the novelty value for entire drives as well as for individual sequences can greatly reduce redundancies in the testset. When selecting complete test drives, the distance to be simulated in the final testset was reduced to 69%. With the sequence-based approach, only 19% of the total distance remained, including 300 additional meters of buffer data before each track to reinitialize the simulation. This reduction in length equals a reduction of 29% in time and simulation effort. The variation is caused by varying velocities, resulting in more or less feature execution cycles per meter. This is an indication, that percentage-wise more repetitive highway sequences than diverse city traffic sequences have been filtered out in the final testset and therefore the average velocity of the remaining sequences is lowered.

VI. CONCLUSION AND FUTURE WORK

We introduced an automated approach for testset reduction utilizing an autoencoder. It's compression and reproduction capabilities were used to recreate an input real-world-driving-data sequence. We enabled the autoencoder to consider the temporal behavior of signals by flattening the time-series data into vectors. It has been shown, that the reproduction error can be interpreted as a novelty indicator for the sequence, showing how well the sequence is known to the autoencoder. We can not globally verify this approach, however, we have done so exemplary indicating the validity of this approach.

We covered different network and data parameters, which influence the networks performance. Further, we compared the reduction capabilities of a drive-based and a sequence-based evaluation approach.

In future work we will try to interpret the novelty value of sequences by for example automatic labeling of scenarios within the data or classification of signal values. Additional signals will be included in the autoencoder to improve the novelty rating further. Furthermore, with the ability to freely select used signals, we plan the creation of themed testsets, which focus on specific aspects, e.g. a testset which explicitly includes different types of curves. We will also make efforts to evaluate and validate this approach.

REFERENCES

[1] K. Bengler, K. Dietmayer, B. Farber, M. Maurer, C. Stiller, and H. Winner, "Three decades of driver assistance systems: Review and future perspectives," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 4, pp. 6–22, 2014.

[2] B. Lightsey, "Systems engineering fundamentals. dtic document," 2001.

[3] W. Wachenfeld and H. Winner, "The release of autonomous vehicles," in *Autonomous Driving*. Springer, 2016, pp. 425–449.

[4] T. Helmer, L. Wang, K. Kompass, and R. Kates, "Safety performance assessment of assisted and automated driving by virtual experiments: Stochastic microscopic traffic simulation as knowledge synthesis," in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*. IEEE, 2015, pp. 2019–2023.

[5] S. Khastgir, G. Dhadyalla, S. Birrell, S. Redmond, R. Addinall, and P. Jennings, "Test scenario generation for driving simulators using constrained randomization technique," SAE Technical Paper, Tech. Rep., 2017.

[6] J. Bach, S. Otten, M. Holzäpfel, and E. Sax, "Reactive-replay approach for verification and validation of closed-loop control systems in early development," in *SAE Technical Paper 2017-01-1671*, 2017.

[7] J. Bach, J. Langner, S. Otten, M. Holzäpfel, and E. Sax, "Test scenario selection for system-level verification and validation of geolocation-dependent automotive control systems," in *23rd International Conference on Engineering, Technology and Innovation (ICE/IEEE)*, 2017.

[8] V. Q. W. G. 13 and A. SIG, "Automotive spice process assessment / reference model," 7 2015. [Online]. Available: http://www.automotivespice.com/

[9] B. W. Boehm, "Software engineering," *IEEE Transactions on Computers*, vol. 25, no. 12, pp. 1226–1241, 1976. [Online]. Available: http://dx.doi.org/10.1109/TC.1976.1674590

[10] P. Bourque, R. E. Fairley, *et al.*, *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0*. IEEE Computer Society Press, 2014.

[11] E. Sax, Ed., *Automatisiertes Testen eingebetteter Systeme in der Automobilindustrie*. München: Hanser, 2008. [Online]. Available: http://www.hanser-elibrary.com/action/showBook?doi=10.3139/9783446419018

[12] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*, Sept 2015, pp. 982–988.

[13] S. Otten, J. Bach, C. Wohlfahrt, C. King, J. Lier, H. Schmid, S. Schmerler, and E. Sax, "Automated assessment and evaluation of digital test drives," in *Advanced Microsystems for Automotive Applications (AMAA) 2017*. Springer International Publishing, 2017, pp. 189–199.

[14] F. Schuldt, F. Saust, B. Lichte, M. Maurer, and S. Scholz, "Effiziente systematische testgenerierung für fahrerassistenzsysteme in virtuellen umgebungen," *Automatisierungssysteme, Assistenzsysteme und Eingebettete Systeme Für Transportmittel*, 2013.

[15] P. Glauner, A. Blumenstock, and M. Haueis, "Effiziente felderprobung von fahrerassistenzsystemen," *UNI DAS eV (ed.)*, vol. 8, pp. 5–14, 2012.

[16] L. Eckstein and A. Zlocki, "Safety potential of adas–combined methods for an effective evaluation," in *23rd International Technical Conference on the Enhanced Safety of Vehicles (ESV) Seoul, South Korea*, 2013.

[17] J. Mazzega, F. Köster, K. Lemmer, and T. Form, "Absicherung hochautomatisierter fahrfunktionen," *ATZ-Automobiltechnische Zeitschrift*, vol. 118, no. 10, pp. 48–53, 2016.

[18] J. Bach, K.-L. Bauer, M. Holzäpfel, M. Hillenbrand, and E. Sax, "Control based driving assistant functions test using recorded in field data," *Proc. 7. Tagung Fahrerassistenzsysteme*, 2015.

[19] L. Theis, W. Shi, A. Cunningham, and F. Huszár, "Lossy image compression with compressive autoencoders," *arXiv preprint arXiv:1703.00395*, 2017.

[20] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[21] S. Hawkins, H. He, G. Williams, and R. Baxter, "Outlier detection using replicator neural networks," in *DaWaK*, vol. 2454, 2002, pp. 170–180.

[22] M. Weber, F. Pistorius, E. Sax, J. Maas, and B. Zimmer, "A hybrid anomaly detection system for electronic control units featuring replicator neural networks," in *Future of Information and Communication Conference (FICC), 2018 IEEE*, 2018, to appear in.

[23] J. Bach, J. Langner, S. Otten, M. Holzäpfel, and E. Sax, "Data-driven development, a complementing approach for automotive systems engineering," in *2017 IEEE International Symposium on Systems Engineering (ISSE)*, 2017.

[24] "Iso/iec/ieee international standard - software and systems engineering–software testing–part 4: Test techniques," *ISO/IEC/IEEE 29119-4:2015*, pp. 1–149, Dec 2015.

[25] A. G. Barnston, "Correspondence among the correlation, rmse, and heidke forecast verification measures; refinement of the heidke score," *Weather and Forecasting*, vol. 7, no. 4, pp. 699–709, 1992.