RL-based Generation of a Synthetic Automotive Driving Scenario Dataset

Riccardo Berta[1], Francesco Bellotti[1], Marianna Cossu[1], Luca Forneris[1], Luca Lazzaroni[1], and Alessandro Pighetti[1]

[1] Department of Electrical, Electronic and Telecommunication Engineering (DITEN), University of Genoa, Via Opera Pia 11a, 16145 Genova, Italy
alessandro.pighetti@edu.unige.it

Abstract. The increasing complexity of automated driving functions (ADFs) necessitates efficient data collection strategies due to the significant resources required for real-world data acquisition. Traditional methods rely on real-world vehicles, which implies costly and time-consuming collection. On the other hand, modern synthetic data generation methods do not explicitly account for the variation in complexity between the scenario classes and the data balance in relation to the training performance of the model. This paper presents a novel approach for generating synthetic automotive driving scenario datasets using a Reinforcement Learning (RL) -based framework. The system is designed to optimize data distribution dynamically at runtime, thus enhancing the stability of the training procedure. The proposed architecture iteratively guides the scenario generation process by identifying the most difficult scenarios in terms of model training F1 score performance. Preliminary experimental results demonstrate that this approach effectively stabilizes the training procedure for different scenario classes by introducing a controlled data imbalance. The study provides promising insights into the potential of RL-driven data generation for learning modules, also highlighting areas for future research to further enhance the capabilities of the system.

Keywords: Driving Scenarios, Driving Scenarios Generator, Synthetic Datasets, Automated Driving, Reinforcement Learning, Dataset Engineering.

1 Introduction

The development of reliable ADFs requires thorough study and understanding of the surrounding dynamic driving scene. Before the real-life deployment phase, an in-depth analysis of the Operational Design Domain (ODD) of the target ADF is necessary. This can be achieved through the creation of automotive scenario datasets, to both gather large amounts of data to instill relevant information into the target ADF, as well as to evaluate the system in its ODD. Driving scenarios are temporal sequences of data that describe a dynamic scene in the ADF operational environment which usually includes interactions with other ODD-specific and common objects (e.g. vehicles, pedestrians etc.) as well as complex decision-making aspects [1].

A common approach for gathering automotive scenario data is to deploy vehicles equipped with multiple sensors and record driving session in real time and labeling this data automatically through Machine Learning (ML) techniques [2, 3]. A different and more recent approach involving real-world data consists in extracting relevant scenarios from large driving dataset [4]. Even through extraction, such methods rely on huge quantities of high-quality sensor data which is very expensive due to the investment in components and the time-consuming labeling procedures, especially when dealing with complex temporal sequences. This problem can be partially addressed by exploiting digital and high-fidelity simulation environments such as the well-known open-source Car Learning to Act (CARLA) [5] framework. Recent approaches generate synthetic scenario datasets from such simulators to complement real-life data, enabling both ADF training and evaluation [6, 7]. Generating data through simulators comes with the benefit of not relying on expensive equipment while also providing deep customization options for the scenarios, allowing for a greater variety of samples overall. The scene to be synthetically generated is usually defined through a set of parameters and rules guiding the simulation agents. Recent work adopts a Deep Reinforcement Learning (DRL) approach to generate diverse synthetic driving scenes [8]. This method introduces even more variety to the final dataset by guiding vehicle agents through interactions with the surrounding environment. However, these methods do not directly consider the difference in complexity between the scenario classes and the data balance in relation to the training performance of the model.

In this context, this paper presents the development of an RL-based architecture for the generation of synthetic automotive scenarios, providing the training model with specific data at runtime to stabilize its learning process. This is achieved through a Learner-Teacher mechanism that efficiently identifies the most difficult scenario categories for the training task. Finally, a first practical evaluation of the pipeline is performed by defining a scenario detection problem and analyzing the result in terms of the final data distribution rather than the performance of the learning model itself. The full training session is reported, demonstrating the effectiveness of the proposed architecture in identifying the most complex scenario categories by introducing a controlled data imbalance.

2 System architecture

Fig. 1 depicts the proposed system architecture, containing its three main components: the Simulator, the Teacher and the Learner. The Simulator module, based on previous work [6], oversees the current scenario description and performs the actual simulation task through CARLA. Here, the Director sub-module is tasked to produce an Extensible Markup Language (XML) document describing the current scenario by random initialization of the scene parameters. The Scenario Generator then runs the scene as per the XML definition using CARLA. The output simulation scene is then sub-divided into single frames and passed to the Learner module.

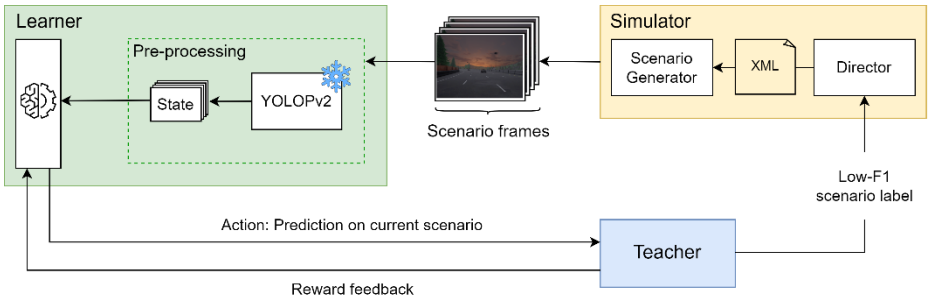

Fig. 1. Proposed DRL-based system architecture.

The Learner is the central DRL component that controls the generation process by acting according to the input state while training for a particular task specified by the user. To improve the learning capability of the underlying DRL algorithm, a pre-processing unit is responsible for decomposing each simulation frame into relevant features that are collectively used as the current state for the algorithm. This is achieved by using a pre-trained and frozen YOLOPv2 [9] model that extracts both the drivable area and highlights the presence of surrounding vehicles through bounding box detection. The resulting pre-processed frames are downscaled to a lower resolution (from 1280x720 to 64x64) and converted to greyscale images before being passed as a state to allow for less training time while retaining important information about the current driving context. The final block of the Learner module is the DRL agent itself, which receives the current state as input and returns an action that corresponds to its prediction. The structure of the action depends on the user-defined learning task and serves as an input for the Teacher component.

The Teacher block monitors both the learning procedure and the distribution of data generation by evaluating the prediction obtained from the learning model and aligning the next simulation iteration with the Learner's deficits. This is done by analyzing the prediction of the model and by task-based modelling of the reward function, which provides feedback to the Learner in a classical RL manner. The scenarios to be generated are at initially selected randomly and, after a user-defined threshold of N episodes (set by default to 200), guided by the Teacher module. After the threshold episode, the Teacher calculates the F1 scores for each scenario class based on the current performance of the training model and selects the scene with the lowest score. This is then the next scenario class defined and generated by the Simulation module. After the Learner responds with a prediction on the new scenario instance the next scenario is chosen based on the current lowest F1 score of the agents' action. This process is iteratively repeated until Learner model convergence.

The full execution loop of the proposed architecture aims at ensuring both a stable learning procedure for the DRL agent and runtime data generation to favor overcoming model deficits in the current task.

3 Synthetic driving scenarios

The synthetic dataset is composed of five driving scenarios, as described in Table 1. During model training one of the shown classes are simulated, divided in single frames and passed to the learning algorithm, which is tasked to give out a prediction based on the user-defined problem.

For each individual class, different settings and weather conditions may be defined, such as in [6]. During training time, both common and scenario-specific parameters are chosen under uniform distribution to guarantee variability and improve upon the generalization capabilities of the learning model.

Fig. 2 shows some of the possible weather combinations generated through the proposed procedure. This definition of driver scenario settings is performed by the Director module.

Table 1. Dataset driving scenario classes

Scenario Class	Description
Cut Out (Front)	The lead vehicle performs a lane change in either lateral direction
Cut Out (Behind)	The rear vehicle performs a lane change in either lateral direction
Cut In (Front)	A vehicle performs a lane change from either lateral direction, becoming the lead vehicle
Cut In (Behind)	A vehicle performs a lane change from either lateral direction, becoming the rear vehicle
Ego Lane Change	The ego vehicle performs a lane change in either lateral direction

The simulation is performed through the Scenario Runner framework which runs the driving scene, defined through the instance description document, within CARLA. Originally, the Scenario Generator block would output data as MP4 format video sequences. Since this kind of data is particularly memory-hungry and hard to use as input data for a learning model at run-time, we opted for subdividing the simulation instances in a user-defined number of stacks of subsequent frames per batch. This is done before the pre-processing module.

Fig. 2. Examples of different weather conditions. (a) Fog and rain, (b) clear dawn and (c) foggy night.

4 Experiment

To take a first step towards thorough validation of the proposed method, we define an example problem for the agent to solve, as a user of the system would. In this context, we define the task as a 5-class scenario detection problem which will have to be solved by the DRL-based agent. To guide our agent towards a solution to this problem, we deploy the Proximal Policy Optimization (PPO) [10] algorithm that aims to increase the projected reward intake. It achieves stability and consistent performance by iteratively modifying the policy parameters in response to a clipped surrogate objective, striking a balance between exploration and exploitation. We use the implementation of PPO provided by the Stable-Baselines3 [11] open-source library. To enable correct communication between the DRL-driven pipeline and CARLA we deploy the CARLA-Gym bridge that we proposed in prior work [12].

For this initial evaluation of the system, the learning agent has access to the fully observable state space, defined as $S = \{FS_0, FS_1, \dots, FS_i\}$, where $FS = [f_0, \dots, f_j]$ is a stack of the pre-processed frames, composed of j subsequent frames for the current scenario. For our experiment we choose $i = 10$ to ensure the model properly observes the scenario at hand before predicting its class. The action space of the agent is defined as $A = \{l_0, l_1, l_2, l_3, l_4\}$, where l is a string literal representing one of the 5 scenario classes presented in Table 1 (e.g. "CutOutFront"). The reward function defining the objective of the agent is defined as a simple binary-feedback term of the form:

$$R = \begin{cases} -1, & if\ \hat{l} \neq l' \\ 1, & if\ \hat{l} = l' \end{cases},$$

where $\hat{l}$ refers to the agent predicted detection label for the current scenario and l' is the ground truth label. The Teacher module activates after 450 episodes.

As a benchmark for the system, we train a PPO agent for 4300 episodes, which equal to roughly 14k simulation time steps. The experiments were conducted on a Linux machine with one NVIDIA Quadro RTX 4000 GPU, with 8GB V-RAM. Since for each training episode, one scenario is generated, the final dataset is composed of 4300 scenarios. The deep learning architecture at the core of the PPO learning algorithm is a Convolutional Neural Network (CNN), structured as per standard library definition [13]. This basic CNN structure is chosen for training speed purposes and serves, for this initial assessment, as a proof of concept to focus the study on the assessment of the overall pipeline generation capabilities. To evaluate the performance of the proposed system, we focus primarily on how the data distribution, supervised by the Teacher module, effects the training curve of the model, specifically its F1 score, for each of the scenario classes.

Table 2 presents the distribution of data across the generated scenario classes. The final distribution underscores that the "Cut Out (Behind)" scenario posed the most significant challenge to our DRL-based detector in terms of accurate labeling. This class exhibited the highest number of samples among all classes throughout the entire experiment. Collectively, the two scenarios depicting the rear view of the surroundings of the ego vehicle constitute most of the dataset, accounting for over 66% of the total data.

This result shows that the proposed method is able to accurately identify the most challenging scenario categories (e.g. “Behind”), and not only the individual classes themselves. This enables comprehensive studies on the corresponding generation methods and the components of these scenario categories.

Table 2. Synthetic dataset distribution after training for 4300 episodes

Scenario Class	Number of samples
Cut Out (Front)	417 (9.7%)
Cut Out (Behind)	**1534 (35.7%)**
Cut In (Front)	589 (13.7%)
Cut In (Behind)	**1329 (30.9%)**
Ego Lane Change	431 (10%)
Total generated scenarios	**4300**

Fig. 3 (a) depicts the F1 score of the model, calculated at training time, while Fig. 3 (b) shows the data distribution of the generated scenarios as the dataset size increases over training time. By comparing the figures, we can see how, even though the agent is not able to solve the problem effectively yet (i.e. correctly detecting the scenarios with high confidence), the F1 scores for all the classes are rising simultaneously. This shows how, by prioritizing the most difficult scenario instances throughout training, the proposed method is able to stabilize the learning process of the model over all the classes.

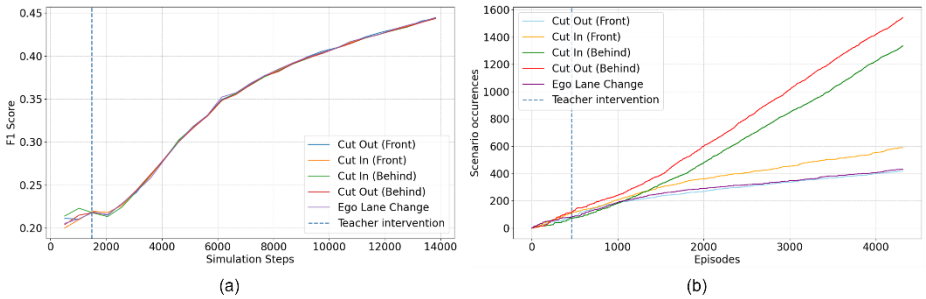

Fig. 3. Training plots for (a) model F1 score over simulation time steps and (b) scenario class distribution over dataset size. The dotted blue line refers to the Teacher intervention threshold.

This detection task served as a tool to approximate a sample data distribution generated by the proposed architecture. The final performance of the agent is to be improved upon through the deployment of more complex network structures and hyperparameter tuning, which were left out of this experimental phase to speed up data generation.

5 Conclusion

In this work, we deployed a DRL-driven pipeline to generate synthetic datasets for automotive scenarios. The final dataset comprises 4,300 samples, with the highest occurrences of scenarios taking place behind the vehicle. Although it is only a preliminary study, the evaluation of the data distribution of the generated scenarios has highlighted the most difficult scenario categories for the model to process. The proposed method has shown to provide a stable training procedure for the agent by prioritizing the generation of the lowest scoring data classes and populating the dataset accordingly. This is emphasized by the upward trend of the F1 score throughout the training process, which remains similar for all scenario classes after Teacher intervention.

Future work will include an improved data generation pipeline tailored to runtime execution and a higher-level feature extraction method to reduce training time by retaining only important data information, thus reducing the amount of state exploration required. The complexity of the underlying architecture of the neural network will also be part of the next research direction to train a model capable of solving the task at hand. Moreover, the Teacher structure will be replaced by a more advanced adversarial policy to enhance the training model capabilities. This paradigm has proven to be effective in training robust and reliable DRL policies in different fields [14, 15] and will be a direct upgrade over the current F1 scoring mechanism. Finally, to further validate the proposed pipeline, other learning problems will be investigated in addition to the analyzed detection task.

References

1. Berta, R., Lazzaroni, L., Capello, A., Cossu, M., Forneris, L., Pighetti, A., Bellotti, F.: Development of deep-learning-based autonomous agents for low-speed maneuvering in Unity. Journal of Intelligent and Connected Vehicles. (in press). https://doi.org/10.26599/JICV.2023.9210039.
2. Izquierdo, R., Quintanar, A., Parra, I., Fernandez-Llorca, D., Sotelo, M.A.: The PREVENTION dataset: a novel benchmark for PREdiction of VEhicles iNTentIONs. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). pp. 3114–3121. IEEE, Auckland, New Zealand (2019). https://doi.org/10.1109/ITSC.2019.8917433.
3. Ettinger, S., Cheng, S., Caine, B., Liu, C., Zhao, H., Pradhan, S., Chai, Y., Sapp, B., Qi, C., Zhou, Y., Yang, Z., Chouard, A., Sun, P., Ngiam, J., Vasudevan, V., McCauley, A., Shlens, J., Anguelov, D.: Large Scale Interactive Motion Forecasting for Autonomous Driving : The Waymo Open Motion Dataset. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV). pp. 9690–9699. IEEE, Montreal, QC, Canada (2021). https://doi.org/10.1109/ICCV48922.2021.00957.
4. Guo, D., Sánchez, M.M., De Gelder, E., Van Der Sande, T.P.J.: Scenario Extraction from a Large Real-World Dataset for the Assessment of Automated Vehicles. In: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC). pp. 1570–1575. IEEE, Bilbao, Spain (2023). https://doi.org/10.1109/ITSC57777.2023.10421930.

5. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An Open Urban Driving Simulator, http://arxiv.org/abs/1711.03938, (2017).
6. Cossu, M., Berta, R., Capello, A., De Gloria, A., Lazzaroni, L., Bellotti, F.: Developing a Toolchain for Synthetic Driving Scenario Datasets. In: Berta, R. and De Gloria, A. (eds.) Applications in Electronics Pervading Industry, Environment and Society. pp. 222–228. Springer Nature Switzerland, Cham (2023). https://doi.org/10.1007/978-3-031-30333-3_29.
7. Abeysirigoonawardena, Y., Shkurti, F., Dudek, G.: Generating Adversarial Driving Scenarios in High-Fidelity Simulators. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 8271–8277. IEEE, Montreal, QC, Canada (2019). https://doi.org/10.1109/ICRA.2019.8793740.
8. Lu, C., Yue, T., Ali, S.: DeepScenario: An Open Driving Scenario Dataset for Autonomous Driving System Testing. In: 2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR). pp. 52–56. IEEE, Melbourne, Australia (2023). https://doi.org/10.1109/MSR59073.2023.00020.
9. Han, C., Zhao, Q., Zhang, S., Chen, Y., Zhang, Z., Yuan, J.: YOLOPv2: Better, Faster, Stronger for Panoptic Driving Perception, http://arxiv.org/abs/2208.11434, (2022). https://doi.org/10.48550/arXiv.2208.11434.
10. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal Policy Optimization Algorithms, http://arxiv.org/abs/1707.06347, (2017). https://doi.org/10.48550/arXiv.1707.06347.
11. Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., Dormann, N.: Stable-Baselines3: Reliable Reinforcement Learning Implementations. Journal of Machine Learning Research. 22, 1–8 (2021).
12. Lazzaroni, L., Pighetti, A., Bellotti, F., Capello, A., Cossu, M., Berta, R.: Automated Parking in CARLA: A Deep Reinforcement Learning-Based Approach. In: Bellotti, F., Grammatikakis, M.D., Mansour, A., Ruo Roch, M., Seepold, R., Solanas, A., and Berta, R. (eds.) Applications in Electronics Pervading Industry, Environment and Society. pp. 352–357. Springer Nature Switzerland, Cham (2024). https://doi.org/10.1007/978-3-031-48121-5_50.
13. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature. 518, 529–533 (2015). https://doi.org/10.1038/nature14236.
14. Gleave, A., Dennis, M., Wild, C., Kant, N., Levine, S., Russell, S.: Adversarial Policies: Attacking Deep Reinforcement Learning, http://arxiv.org/abs/1905.10615, (2021). https://doi.org/10.48550/arXiv.1905.10615.
15. Pighetti, A., Bellotti, F., Oh, C., Lazzaroni, L., Forneris, L., Fresta, M., Berta, R.: Investigating Adversarial Policy Learning for Robust Agents in Automated Driving Highway Simulations. In: Bellotti, F., Grammatikakis, M.D., Mansour, A., Ruo Roch, M., Seepold, R., Solanas, A., and Berta, R. (eds.) Applications in Electronics Pervading Industry, Environment and Society. pp. 124–129. Springer Nature Switzerland, Cham (2024). https://doi.org/10.1007/978-3-031-48121-5_18.