Deeper at the SBST 2021 Tool Competition: ADAS Testing Using Multi-Objective Search

Mahshid Helali Moghadam*†, Markus Borg*, Seyed Jalaleddin Mousavirad‡
* *RISE Research Institutes of Sweden* , Sweden
{mahshid.helali.moghadam, markus.borg}@ri.se
† *Mälardalen University* ,Västerås, Sweden
‡ *Department of Computer Engineering, Hakim Sabzevari University*, Sabzevar, Iran
{sj.mousavirad}@hsu.ac.ir

***Abstract*—Deeper is a simulation-based test generator that uses an evolutionary process, i.e., an archive-based NSGA-II augmented with a quality population seed, for generating test cases to test a deep neural network-based lane-keeping system. This paper presents Deeper briefly and summarizes the results of Deeper's participation in the Cyber-physical systems (CPS) testing competition at SBST 2021.**

***Index Terms*—search-based software testing, cyber-physical systems, advanced driver assistance systems, deep learning, automotive simulators**

I. Introduction

There is an increasing trend of leveraging machine learning techniques, in particular deep learning, in a wide range of application areas. Self-driving cars are one of those areas that have benefited considerably from deep learning and its associated techniques such as deep neural networks (DNN). To perform Verification and Validation (V&V) for self-driving cars, there is a need for *System-level* testing to ensure safety and reliability of these systems before making them available to be used publicly [1].

Simulation-based testing is one of the potential approaches for system-level testing, as a proper complementary solution to field testing. Meanwhile, the approach could capture the entire operational environment efficiently and effectively using thorough and high-fidelity physics-based simulators [2]. Lately, various commercial and open-source simulators have been developed to support the need for realistic simulation of self-driving cars [3]–[5]. From the literature, several system-level testing techniques, relying on such simulators, have been proposed in recent years. One category of test generation techniques is based on deriving *critical test inputs* from the input domain using different search-based testing approaches [6]–[8]. The critical test inputs are those that break, or almost break, the safety requirements of the system under test.

This paper presents the results obtained by our proposed approach, the Deeper test generator tool [9], in the Cyber-physical systems (CPS) testing competition at SBST 2021. Deeper uses an evolutionary process mainly based on Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [10], a multi-objective optimization algorithm, to search the input space and generate critical test inputs for a lane-keeping assistance system (LKAS) in the competition setup. The tool has been integrated into the competition's testing infrastructure according to the competition's guideline and executed against the test subject with regard to certain test budgets and pre-defined driving conditions, such as single, flat roads surrounded by green areas and fixed weather conditions.

II. Deeper

Deeper test generator uses a multi-objective search process (i.e., based on NSGA-II) to generate critical test roads, leading the car to get out of the lane. The search process focuses on maximizing the distance of the car from the center of the lane during the simulation, while minimizing the road's length. Deeper augments the NSGA-II with a simple greedy archive to store the non-dominated candidate solutions that force the car to go out of the lane with regard to a certain tolerance threshold. This threshold defines at least "*how much*" (e.g., 50%) of the car must get out of the lane for the test case to be considered as a *failure*. The use of an archive is beneficial to keep a memory and improve the performance of the search process in finding optimal solutions within the search space [11].

Deeper leverages an initial quality population seed to boost the search process with respect to the fixed test budget. It uses Catmull-Rom cubic splines [12] to represent the roads and relies on the evolution operators implemented by DeepJanus [13]. Throughout the development, the impact of the different initial population seeds has been investigated. For instance, starting from a random population seed like an initial seed generated based on a recursive evaluation algorithm for Catmull-Rom cubic splines [14] could not lead the search to find the desired solutions for high failing thresholds within a reasonable test budget.

III. Benchmark results

Deeper is integrated into the testing infrastructure of the competition, which uses the research-specific BeamNG driving simulator. The test subject is BeamNG.AI, the built-in machine learning-enabled driving agent in the simulator. Two experiments, DEFAULT and SBST21, with different setups, presented in Table I, were arranged. The

TABLE I
EXPERIMENTS' SETUPS

Ex. Name	Test Budget	Map Size	Speed Limit	Failing Tolerance
DEFAULT	5h	200×200	None	0.95
SBST21	2h	200×200	70 Km/h	0.85

experiments were executed on a PC, with a quad-core Intel i7-7700K CPU, 16 GB RAM, and an NVidia GeForce GTX 1080 GPU, that runs Microsoft Windows 10 [15].

The target of the competition was to generate the highest number of diverse test cases leading to failures, i.e., the valid test inputs causing the car to get out of the lane. The quality criteria used in the competition for evaluating the tools are as follows [15]:

Detected Failures, which refer to the number of test cases that led the car to drive out of the lane with respect to the pre-defined *tolerance* threshold. Regarding this metric, Deeper could expose failures with an average rate of 2.6 and 0.5 in all runs on DEFAULT and SBST21 setups, respectively.

Failure Diversity, which represents the calculated sparseness between the test inputs leading to the failures. The version of Deeper submitted to the competition did not propose any specific mechanisms for promoting diversity between the generated test roads, thus it showed low sparseness for the generated test cases.

Test Generation Effectiveness and Efficiency, which relate to how the test generator tool uses the allocated test budget to generate the test cases, i.e., how many test cases are generated in total, and what fraction of test cases are valid (or invalid). Regarding efficiency and effectiveness, Deeper showed satisfactory performance. The tool generated many test cases within the allocated test budget in both experimental setups, i.e., Deeper ranked second regarding the number of generated test cases. Meanwhile, it generated the highest number of *valid* test cases among the competing tools, and achieved around 90% generation effectiveness in both setups. Notably, it did neither generate any self-intersecting test roads nor any roads outside the map boundaries [15].

ACKNOWLEDGMENT

This work has been funded by Vinnova through the ITEA3 European IVVES project (https://itea3.org/project/ivves.html). Furthermore, the project received financial support from the SMILE III project financed by Vinnova, FFI, Fordonsstrategisk forskning och innovation under the grant number: 2019-05871 and Kompetensfonden at Campus Helsingborg, Lund University, Sweden.

REFERENCES

[1] M. Borg, C. Englund, K. Wnuk, B. Durann, C. Lewandowski, S. Gao, Y. Tan, H. Kaijser, H. Lönn, and J. Törnqvist, "Safely entering the deep: A review of verification and validation for machine learning and a challenge elicitation in the automotive industry," *Journal of Automotive Software Engineering*, vol. 1, no. 1, pp. 1–19, 2019.

[2] M. Borg, R. B. Abdessalem, S. Nejati, F.-X. Jegeden, and D. Shin, "Digital Twins Are Not Monozygotic–Cross-Replicating ADAS Testing in Two Industry-Grade Automotive Simulators," *arXiv preprint arXiv:2012.06822, accepted in IEEE International Conference on Software Testing, Verification and Validation*, 2021.

[3] BeamNG GmbH., "BeamNG.research," https://beamng.gmbh/research/, Retrieved March, 2021.

[4] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[5] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, S. Mehta *et al.*, "Lgsvl simulator: A high fidelity simulator for autonomous driving," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–6.

[6] R. Ben Abdessalem, S. Nejati, L. C. Briand, and T. Stifter, "Testing advanced driver assistance systems using multi-objective search and neural networks," in *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*, 2016, pp. 63–74.

[7] A. Gambi, M. Mueller, and G. Fraser, "Automatically testing self-driving cars with search-based procedural content generation," in *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2019, pp. 318–328.

[8] R. B. Abdessalem, A. Panichella, S. Nejati, L. C. Briand, and T. Stifter, "Testing autonomous cars for feature interaction failures using many-objective search," in *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2018, pp. 143–154.

[9] "Deeper," https://github.com/mahshidhelali/Deeper_ADAS_Test_Generator, Retrieved March, 2021.

[10] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[11] E. D. De Jong, "The incremental pareto-coevolution archive," in *Genetic and Evolutionary Computation Conference*. Springer, 2004, pp. 525–536.

[12] E. Catmull and R. Rom, "A class of local interpolating splines," in *Computer aided geometric design*. Elsevier, 1974, pp. 317–326.

[13] V. Riccio and P. Tonella, "Model-based exploration of the frontier of behaviours for deep learning system testing," in *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020, pp. 876–888.

[14] P. J. Barry and R. N. Goldman, "A recursive evaluation algorithm for a class of catmull-rom splines," *ACM SIGGRAPH Computer Graphics*, vol. 22, no. 4, pp. 199–204, 1988.

[15] S. Panichella, A. Gambi, F. Zampetti, and V. Riccio, "Sbst tool competition 2021," in *International Conference on Software Engineering, Workshops, Madrid, Spain, 2021*. ACM, 2021.