Testing Learning-Enabled Cyber-Physical Systems with Large-Language Models: A Formal Approach

Xi Zheng
Macquarie University
Sydney, Australia
james.zheng@mq.edu.au

Aloysius K. Mok
University of Texas at Austin
Austin, USA
mok@cs.utexas.edu

Ruzica Piskac
Yale University
New Heaven, USA
ruzica.piskac@yale.edu

Yong Jae Lee
University of Wisconsin-Madison
Madison, USA
yongjaelee@cs.wisc.edu

Bhaskar Krishnamachari
University of Southern California
Los Angeles, USA
bkrishna@usc.edu

Dakai Zhu
University of Texas at San Antonio
San Antonio, USA
Dakai.Zhu@utsa.edu

Oleg Sokolsky
University of Pennsylvania
Philadelphia, USA
sokolsky@cis.upenn.edu

Insup Lee
University of Pennsylvania
Philadelphia, USA
lee@seas.upenn.edu

ABSTRACT

The integration of machine learning into cyber-physical systems (CPS) promises enhanced efficiency and autonomous capabilities, revolutionizing fields like autonomous vehicles and telemedicine. This evolution necessitates a shift in the software development life cycle, where data and learning are pivotal. Traditional verification and validation methods are inadequate for these AI-driven systems. This study focuses on the challenges in ensuring safety in learning-enabled CPS. It emphasizes the role of testing as a primary method for verification and validation, critiques current methodologies, and advocates for a more rigorous approach to assure formal safety.

CCS CONCEPTS

• **Software and its engineering** → **Software verification and validation**; • **Computer systems organization** → **Embedded and cyber-physical systems**; • **Computing methodologies** → *Machine learning*; • **Theory of computation** → *Formal languages and automata theory*.

KEYWORDS

AI-based Systems, LLM-based Testing, automata-learning, model-based testing

ACM Reference Format:

Xi Zheng, Aloysius K. Mok, Ruzica Piskac, Yong Jae Lee, Bhaskar Krishnamachari, Dakai Zhu, Oleg Sokolsky, and Insup Lee. 2024. Testing Learning-Enabled Cyber-Physical Systems with Large-Language Models: A Formal Approach. In *Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering (FSE Companion '24), July 15–19, 2024, Porto de Galinhas, Brazil.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3663529.3663779

FSE Companion '24, July 15–19, 2024, Porto de Galinhas, Brazil

ACM ISBN 979-8-4007-0658-5/24/07
https://doi.org/10.1145/3663529.3663779

1 CONTEXT, MOTIVATIONS AND AIMS

The integration of machine learning (ML) with cyber-physical systems (CPS) has revolutionized various sectors, including transportation, logistics, service industries, and healthcare, with innovations like autonomous vehicles (Waymo [63], Tesla Autopilot [56], Uber ATG [60]), delivery drones (Amazon Prime Air [2], Google Wing [28], Zipline [70]), and robotic surgeries (Da Vinci [15], Mazor [43], Mako [42]). However, these advancements have raised significant safety concerns, evidenced by reported incidents causing fatalities and economic loss [25, 47, 53, 61].

Addressing these issues requires rigorous verification and validation, presenting unique challenges due to the complexities of learning-enabled CPS. These systems incorporate critical machine learning components like perception and planning, as seen in autonomous driving, making them significantly different from traditional software systems. This complexity, involving a paradigm shift in the software development life cycle to incorporate data and learning, demands novel approaches in verification and validation techniques [3, 54].

We present our initial efforts to explore practical testing strategies for the verification and validation of learning-enabled CPS. This focus is particularly relevant given the extensive use of testing in the CPS industry and the considerable amount of recent literature on this topic. From a summary of the current state-of-the-art testing methodologies for learning-enabled CPS, we propose a roadmap to formalize the testing effort.

More specifically, we use large-language models (LLMs) to extract human knowledge from existing rules and regulations, and analyze vast amounts of data generated or captured by learning-enabled CPS, including sensor data and logs. By extracting human knowledge and analyzing data, LLMs can offer insights into the system's behavior and generate a wealth of realistic and high-quality test data. With this improved data quality, it becomes feasible to employ data-driven learning to extract underlying formal specifications.

The large language models can also be tasked with identifying corresponding formal specifications. Based on these derived formal specifications, we can engage in model-based testing—a considerably more formal approach than existing methodologies, such as search-based testing.

We present a case study utilizing a vision-based LLM to analyze traffic accident from photos, showing promising results. This lays a foundation for employing multi-modal LLMs to distill meaningful latent representations from real-world sensor data in learning-enabled CPS. For instance, leveraging front-facing cameras in autonomous vehicles to capture diverse traffic incidents. Another study demonstrates using GPT-4 and a customized domain-specific language to extract knowledge and generate test scenarios from a real-world traffic handbook. Early results indicate potential in uncovering diverse bugs in autonomous driving systems, aligning well with the roadmap outlined for formal testing and verification.

2 RELATED WORK

Recent research in formal verification of machine learning models and testing of learning-enabled CPS has used various techniques like NNV star sets [59], Sherlock [24], Reluplex [36], and Branch and Bound [10]. However, these approaches often lack in providing comprehensive safety guarantees and are limited in handling the complexity of industry-scale, multi-modal CPS [41, 46]. Similarly, while testing in system properties [17, 39, 57] and system robustness [12, 16, 58] are progressing, they fail to offer concrete formal assurances. This highlights the gap in current methodologies and underscores the need for more robust testing approaches that can ensure safety and reliability in real-world applications of CPS [50, 52, 65].

3 A ROADMAP TOWARDS FORMAL TESTING

Our roadmap, depicted in Fig.1, targets Multi-Modal LLM-based Test Generation, Data-Driven Model Learning, and Model-based Testing. We begin by capturing test scenarios from varied sources such as rules, sensor data, and accident logs, processing them with domain-specific languages like OpenScenario [7] and Scenic [27] via an LLM-based parser (ChatGPT or LLaVA [40]). This initial phase is vital for generating diverse test cases, particularly for safety and liveness properties. Subsequent phases involve Data-Driven Model Learning using the L^* algorithm for model construction and validation through high-fidelity simulations in collaboration with industry leaders [18, 20, 21]. The final phase, Model-based Testing, utilises these models to generate test cases that formally assure compliance with specified requirements. This approach is designed to effectively bridge the gap between model checking and runtime verification in learning-enabled CPS.

3.1 Multi-Modal LLM-Based Test Generation

Generating diverse test cases that violate the formal properties of learning-enabled CPS, setting the conditions for our second stage, is a significant hurdle [41]. This complexity arises from a vast search space [16, 39, 57] and limited domain knowledge [17, 58]. On the other hand, LLMs are being intensively studied for their applications in personalized learning [34], software testing [62], and multi-modal scenarios [31]. In learning-enabled CPS, abundant data including system logs and sensor outputs (such as camera and lidar feeds) offer opportunities for fine-tuning multi-modal LLMs. These refined models can interpret the semantics of various physical environments and their interaction with the system. For example, in some countries, front-facing cameras are widely installed in vehicles to deter "pedestrian scams", where individuals deliberately throw themselves in front of cars to extort drivers or launch false insurance claims [11]. An ancillary benefit of this is that multi-angle video data can be collected for individual traffic incidents. Such data, coupled with accident descriptions, can be used to train video-based LLMs to understand the nuanced factors leading to accidents.

These fine-tuned LLMs can then facilitate test generation in various ways. For example, they can be employed for seed generation, mutation, and selection in fuzzing techniques, a concept already explored in traditional software domains [19, 66]. Similarly, in the case of autonomous drone systems, drones capture a range of images of surveying or landing sites [1, 22]. These images, along with local government guidelines on surveying and landing, can be utilised to generate relevant test cases. To mitigate hallucination, we implement multi-stage validation to ensure the correctness of LLM outputs, as demonstrated in our pilot study briefly discussed in Section 4.1.

3.2 Data-Driven Model Learning

Model learning techniques, from Biermann's offline approach [9] to Angluin's online L^* algorithm [4], have been thoroughly explored. Online methods generally outperform offline methods, being polynomial rather than NP-complete [38]. Implementing membership and equivalence queries, however, introduces practical challenges due to the complexities of real system tests [55]. Recent studies suggest alternatives like falsification for equivalence queries [5] and using positive examples for learning signal temporal logic (STL) properties [32], though these require extensive setup or strong assumptions [8, 33]. Our goal is to refine system behavior modelling using structured learning to bypass these traditional automata-learning limitations, utilising diverse corner cases generated in the prior step and leveraging our experience in building high-fidelity simulation/co-simulation environments as oracles, we can effectively address the queries needed for the adapted L^* algorithm.

The process begins with the L^* learning algorithm, which issues membership queries to a simulator acting as an oracle for the underlying CPS, such as Carla [23] for autonomous driving and AirSim for unmanned aerial vehicles [51]. These queries are instrumental in exploring the system's behavior across a myriad of scenarios generated by the LLM in Section 3.1, with the simulator's feedback aiding the construction of a preliminary hypothesis model.

The membership queries, categorized into positive and negative test cases executed and evaluated by the simulator, are pivotal in shaping the hypothesis model. Positive test cases provide insights into valid system behaviors, while negative test cases highlight invalid or erroneous behaviors. This dichotomy aids in the accurate refinement of the hypothesis model, steering it towards a more accurate representation of the system's dynamics. As the hypothesis model evolves, the role of equivalence queries becomes important. These queries are directed at the simulator to ascertain the consistency

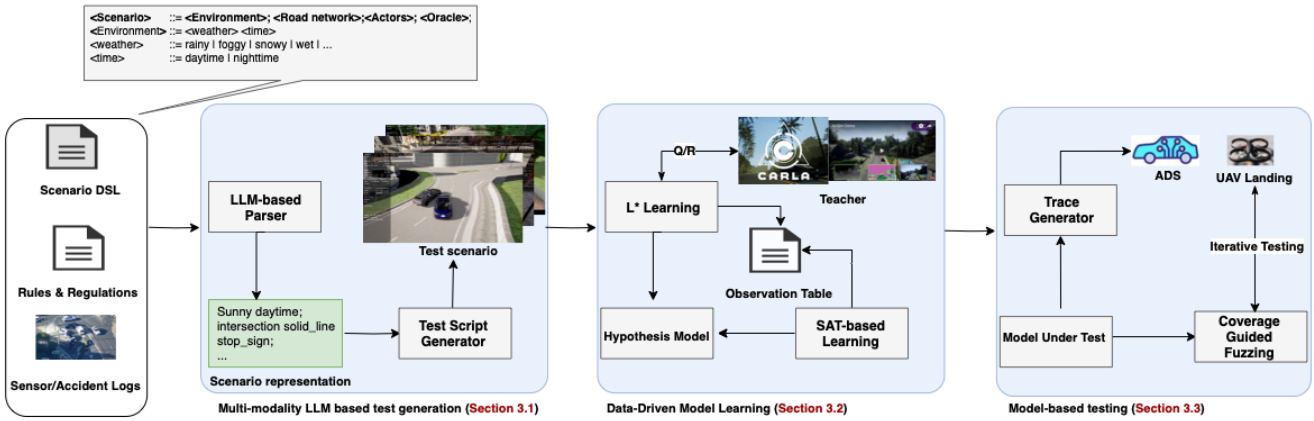

Figure 1: Proposed Roadmap

between the hypothesis model and the simulated system behavior. Discrepancies uncovered through equivalence queries point out areas where the hypothesis model needs further refinement.

Parallelly, the SAT-based learning algorithm similar to [45] enhances this iterative learning process by managing the logical structuring of the constraints derived from both membership and equivalence queries. It reviews the consistency of these constraints against the hypothesis model, identifying and rectifying inconsistencies. This complementary role of SAT-based learning augments the L^* learning, ensuring a more precise and robust hypothesis model. The synergy between L^* learning, SAT-based learning, and the strategic deployment of equivalence queries, all interfaced with the simulator, crafts a robust framework for model extraction. This iterative, multi-faceted approach progressively refines the hypothesis model, aligning it closely with the simulated system behavior. The simulator's role as an oracle is crucial, providing a practical and reliable benchmark for validating and refining the hypothesis model.

In summary, this blended learning approach, rooted in iterative interaction with a simulator oracle, presents a viable pathway for extracting accurate models from complex systems. Through an iterative engagement of L^* learning, SAT-based learning, and strategic querying, this proposed solution promises a substantial advancement in the field of model extraction, especially in scenarios where a concrete system specification is absent.

3.3 Model-Based Testing

In this paper, we aim to rejuvenate the concept of "model-based testing" within the realm of learning-enabled CPS. One of the earliest proponents of this concept hails from Bellcore, where a test data model was articulated using a straightforward specification named AETGSpec. This specification supports hierarchy in both fields and relations [14]. Building on this foundation, Schiefer et al. [49] highlighted that test case generation for model-based testing can adopt various methods. One approach is deductive theorem proving, wherein the model is segmented into equivalence classes based on a set of logical expressions. In its most basic form, each class can function as a test case. In the context of model checking, test case generation revolves around identifying counterexamples where the specification is breached. Symbolic execution can be employed to navigate every potential program execution path. Tools like Modbat [6], tailored for event-driven systems, and MoMuT [37], designed for UML and timed automata, facilitate test case generation from state machine models. However, such black-box type of approach relies on random and mutation tests are not applicable to learning-based CPS as key learning models are not applicable to random seed generation and mutation operators.

In our research, we gravitate towards generating test cases steered by STL fuzzing due to STL's ability to express complex temporal and spatial relationships within CPS. In a contemporary study by Meng et al. [44], given a linear-time temporal logic (LTL) property ϕ, a Büchi automaton A$\neg\phi$ can be crafted that recognizes the contravention of the property ϕ. This approach requires manually extracting LTL properties, dealing with LTL's limited expressiveness compared to STL, and identifying program parts impacted by LTL's atomic propositions. The goal is to generate logs to identify potential proposition violations, guiding future test generation. This ensures focus on areas likely to breach LTL properties, enhancing test effectiveness. However, in learning-based CPS, like in autonomous vehicles where speed control is distributed across multiple neural networks, pinpointing specific locations for such violations is unfeasible, complicating the application of this strategy.

We propose using data-driven specifications from Section 3.1 to identify STL properties and construct corresponding automata with suitable acceptance conditions for the negation of these properties. As learning-based CPS often use robotic operating systems with message queues, we will create a trace generator to monitor and record events for each STL predicate, generating relevant traces. Coverage-guided fuzzing will then be employed to produce test cases that activate trace data across sequential states up to the accepting state, creating counterexamples. Considering the complexity of the resulting timed automata, traversal methods such as depth-first, breadth-first, or random walk will be considered to ensure measurable coverage and formal assurance.

In conclusion, our model-based testing paradigm presents a significant improvement from existing fuzzing techniques. Where

conventional coverage-guided fuzzers like AFL [64] primarily detect crashes and memory overflows, and fitness function-guided fuzzers [29, 30, 39, 68] are designed for specific scenarios and oracles with no formal coverage guarantees, our approach elevates the discourse. Recent efforts parallel to ours, albeit requiring manual, error-prone specification of detailed scenarios and properties using new DSLs, underscore the laborious nature of these tasks [69]. Our model-based testing, on the other hand, automates scenario generation and property extraction, rendering it more accessible for industry practitioners dealing with black-boxed critical components in learning-enabled systems. The trace generator plugin in our framework is designed to cater to the extracted specifications, and our coverage-guided fuzzing not only maximizes failure coverage but also explores diverse counterexamples violating the tested property. This approach, therefore, not only aligns with real-world industrial contexts but also opens new opportunities in ensuring more robust, formally verified learning-enabled CPS.

4 EARLY RESULTS AND EFFORTS

We conducted two case studies to investigate Research Question 1 (RQ1): assessing the ability of LLM to generate diverse yet real test cases through in-context learning. Similarly, initial evaluations were made for Research Question 2 (RQ2): examining a multi-modal LLM's understanding of traffic accident's root cause. We will share some promising initial results, aligning well with our roadmap.

4.1 RQ1: Test Case Generation Capability using LLM

Using ChatGpt4.0, we interpreted the Texas traffic rule handbook to create a DSL (Fig. 2) that transforms these rules into traffic scenarios. This DSL, focusing on semantic descriptions rather than precise coordinates, differentiates from others such as OpenScenario and Scenic [7, 26, 48]. We implemented multi-level validation to ensure the correctness of the DSL specifications, mitigating hallucination issues. We then translated these DSL specifications into test scripts for the CARLA simulation platform [23], uncovering significant bugs in autonomous driving systems. The DSL comprises elements like *Environment*, *Road network*, *Actor*, and *Oracle*, each capturing intricate scenario semantics. This approach enabled us to generate diverse, semantically rich test scenarios that revealed rule violations in real-world autonomous systems, aiding developers in pinpointing specific issues. In our experiments, we observed that the MMFN model [67] failed to stop at the stop sign, Autoware [35] collided with a front vehicle, and LAV [13] did not respond to a pedestrian crossing the road. We reported these issues, along with detailed log data, to the developers of these widely-used autonomous driving systems. They utilised our logs to pinpoint the root causes of these rule violations. Further details about the DSL, including insights, examples, and the methodology for translating DSLs into test scripts, along with replicable artifacts, are provided in [20].

4.2 RQ2: Accident Root Cause Analysis using Multi-Modal LLM

In this study, we utilized a vision encoder-based multi-modal Large Language Model (LLaVA [40]) to closely examine specific traffic accidents. Our objective is to investigate the boundaries of what such

```
<Scenario>     ::= <Environment>; <Road network>;
                   <Actors>; <Oracle>;
<Environment>  ::= <weather> <time>
<Road network> ::= <road type> <road marker> <traffic signs>
<Actors>       ::= <ego vehicle>, <npc actors>
<Oracle>       ::= <longitudinal oracles> <lateral oracles>
...
```

Figure 2: High-level structure of the Scenario DSL

a multi-modal LLM can discern about the causes of these incidents. We hypothesize that if the LLM can accurately comprehend these causes, then its latent vector representation should effectively capture essential features. Consequently, this would enable the LLM to generate a variety of "corner cases" that closely resemble real-world traffic incidents. These test cases serve dual functions: they can evaluate the robustness of autonomous driving systems (Section 3.1) and also pave the way for our future work in data-driven learning (Section 3.2) and model-based testing (Section 3.3).

As demonstrated in Figure 3, without fine-tuning or in-context learning, when we posed the question "Can you describe the scene?", LLaVA responded, "The scene in the image shows a chaotic and damaged street, with two cars involved in a collision. One of the cars has been flipped over, and debris is scattered around the area. The accident has caused significant damage to both vehicles...".

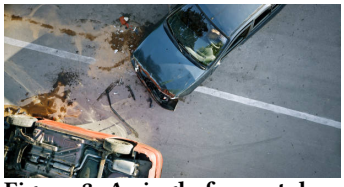

Figure 3: A single frame taken from a traffic collision video

When we modified the query to "Can you imagine what leads to such a collision?", the LLM responded, listing several common causes including speeding, distracted driving, and poor visibility. The first response accurately depicts the accident, while the rest elaborate on potential contributing factors.

We are transitioning from an image-based to a video-based vision encoder in our LLM, expecting improved confidence and temporal information capture. This enhancement should better detect key features of traffic incidents and facilitate the creation of realistic and complex corner cases, as outlined in stage 1 of our plan.

5 CONCLUSION

This paper critically evaluates the existing formal verification processes for learning-enabled CPS and highlights the limitations of traditional software testing methods due to their lack of robust guarantees. Our forward-looking three-stage roadmap addresses key challenges, such as generating diverse corner cases, accessing sensor data ethically, and improving methods for timed automata extraction and state coverage. Our case studies demonstrate the roadmap's potential to fulfill the rigorous requirements of stakeholders like manufacturers, lawmakers, and customers. We anticipate this roadmap will catalyze collaborative efforts across communities to enhance formal guarantees in learning-enabled CPS.

REFERENCES

[1] Md Shah Alam et al. 2021. A survey of safe landing zone detection techniques for autonomous unmanned aerial vehicles (UAVs). *Expert Systems with Applications* 179 (2021), 115091. https://doi.org/10.1016/j.eswa.2021.115091

[2] Amazon Prime Air 2023. *AmazonPrimeAir.* Retrieved Aug 3, 2023 from https://shorturl.at/otyU3

[3] Saleema Amershi et al. 2019. Software engineering for machine learning: A case study. In *ICSE-SEIP*. IEEE, 291–300. https://doi.org/10.1109/ICSE-SEIP.2019.00042

[4] Dana Angluin. 1987. Learning regular sets from queries and counterexamples. *Information and computation* 75, 2 (1987), 87–106. https://doi.org/10.1016/0890-5401(87)90052-6

[5] Yashwanth Annpureddy et al. 2011. S-taliro: A tool for temporal logic falsification for hybrid systems. In *TACAS*. Springer, 254–257.

[6] Cyrille Valentin Artho et al. 2013. Modbat: A model-based API tester for event-driven systems. In *HVC*. Springer, 112–128.

[7] ASAM. 2021. ASAM OpenSCENARIO: User Guide. https://shorturl.at/epuP0.

[8] Ezio Bartocci et al. 2022. Survey on mining signal temporal logic specifications. *Information and Computation* 289 (2022), 104957. https://doi.org/10.1016/j.ic.2022.104957

[9] Alan W Biermann and Jerome A Feldman. 1972. On the synthesis of finite-state machines from samples of their behavior. *IEEE transactions on Computers* 100, 6 (1972), 592–597. https://doi.org/10.1109/TC.1972.5009015

[10] Rudy R Bunel et al. 2018. A unified view of piecewise linear neural network verification. *NeurIPS* 31 (2018).

[11] Mark Button and Graham Brooks. 2016. From 'shallow'to 'deep'policing. *Policing and Society* 26, 2 (2016), 210–229.

[12] Feiyang Cai and Xenofon Koutsoukos. 2020. Real-time out-of-distribution detection in learning-enabled cyber-physical systems. In *ICCPS*. IEEE, 174–183. https://doi.org/10.1109/ICCPS48487.2020.00024

[13] Dian Chen and Philipp Krähenbühl. 2022. Learning from all vehicles. In *CVPR*.

[14] Siddhartha R Dalal et al. 1999. Model-based testing in practice. In *ICSE*. 285–294. https://doi.org/10.1145/302405.302640

[15] Davinci 2023. *Da Vinci Surgical System.* Retrieved Aug 3, 2023 from http://www.intuitivesurgical.com/

[16] Yao Deng et al. 2020. An analysis of adversarial attacks and defenses on autonomous driving models. In *PerCom*. IEEE, 1–10. https://doi.org/10.1109/PerCom45495.2020.9127389

[17] Yao Deng et al. 2022. A declarative metamorphic testing framework for autonomous driving. *TSE* (2022). https://doi.org/10.1109/TSE.2022.3206427

[18] Yao Deng et al. 2022. Scenario-based test reduction and prioritization for multi-module autonomous driving systems. In *FSE*. 82–93. https://doi.org/10.1145/3540250.3549152

[19] Yinlin Deng et al. 2023. Large Language Models are Zero-Shot Fuzzers: Fuzzing Deep-Learning Libraries via Large Language Models. In *ISSTA*. 423–435. https://doi.org/10.1145/3597926.3598067

[20] Yao Deng et al. 2023. TARGET: Traffic Rule-based Test Generation for Autonomous Driving Systems. *arXiv preprint arXiv:2305.06018* (2023).

[21] Yao Deng, Xi Zheng, et al. 2020. RMT: Rule-based Metamorphic Testing for Autonomous Driving Models. *arXiv preprint arXiv:2012.10672* (2020).

[22] Naqqash Dilshad et al. 2020. Applications and challenges in video surveillance via drone: A brief survey. In *ICTC*. IEEE, 728–732.

[23] Alexey Dosovitskiy et al. 2017. CARLA: An open urban driving simulator. In *Conference on robot learning*. PMLR, 1–16.

[24] Souradeep Dutta et al. 2019. Sherlock-a tool for verification of neural network feedback systems: demo abstract. In *HSCC*. 262–263.

[25] DW. 2015. *Volkswagen: Robot kills worker installing it.* Retrieved Aug 5, 2023 from https://t.ly/aK_qb

[26] Daniel J. Fremont et al. 2019. Scenic: A Language for Scenario Specification and Scene Generation. In *PLDI* (Phoenix, AZ, USA). ACM, 63–78.

[27] Daniel J Fremont et al. 2022. Scenic: A language for scenario specification and data generation. *Machine Learning* (2022), 1–45.

[28] Google Wing 2023. *Google Wing.* Retrieved Aug 3, 2023 from http://www.wing.com/

[29] Fitash Ul Haq et al. 2023. Many-objective reinforcement learning for online testing of dnn-enabled systems. In *ICSE*. IEEE, 1814–1826. https://doi.org/10.1109/ICSE48619.2023.00155

[30] Fitash Ul Haq, Donghwan Shin, and Lionel Briand. 2022. Efficient online testing for DNN-enabled systems using surrogate-assisted and many-objective optimization. In *ICSE*. 811–822. https://doi.org/10.1145/3510003.3510188

[31] Hanyao Huang et al. 2023. ChatGPT for shaping the future of dentistry: the potential of multi-modal large language model. *International Journal of Oral Science* 15, 1 (2023), 29.

[32] Susmit Jha et al. 2017. Telex: Passive stl learning using only positive examples. In *RV*. Springer, 208–224.

[33] Austin Jones, Zhaodan Kong, and Calin Belta. 2014. Anomaly detection in cyber-physical systems: A formal methods approach. In *CDC*. IEEE, 848–853.

[34] Enkelejda Kasneci et al. 2023. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and individual differences* 103 (2023), 102274. https://doi.org/10.1016/j.lindif.2023.102274

[35] Shinpei Kato et al. 2018. Autoware on board: Enabling autonomous vehicles with embedded systems. In *ICCPS*. IEEE, 287–296.

[36] Guy Katz et al. 2017. Reluplex: An efficient SMT solver for verifying deep neural networks. In *CAV*. Springer, 97–117.

[37] Willibald Krenn et al. 2015. Momut:: UML model-based mutation testing for UML. In *ICST*. IEEE, 1–8. https://doi.org/10.1109/ICST.2015.7102627

[38] Martin Leucker. 2006. Learning meets verification. In *FMCO*. Springer, 127–151.

[39] Guanpeng Li et al. 2020. Av-fuzzer: Finding safety violations in autonomous driving systems. In *ISSRE*. IEEE, 25–36. https://doi.org/10.1109/ISSRE5003.2020.00012

[40] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *NeurIPS* (2023).

[41] Guannan Lou et al. 2022. Testing of autonomous driving systems: where are we and where should we go?. In *FSE*. 31–43. https://doi.org/10.1145/3540250.3549111

[42] Mako 2023. *Mako Robotic-Arm.* Retrieved Aug 3, 2023 from https://t.ly/VLEwP

[43] Mazor 2023. *Mazor Robotics.* Retrieved Aug 3, 2023 from http://www.medtronic.com/

[44] Ruijie Meng et al. 2022. Linear-time temporal logic guided greybox fuzzing. In *ICSE*. 1343–1355. https://doi.org/10.1145/3510003.3510082

[45] Daniel Neider and Ivan Gavran. 2018. Learning linear temporal properties. In *FMCAD*. IEEE, 1–10. https://doi.org/10.23919/FMCAD.2018.8603016

[46] Aditya Prakash et al. 2021. Multi-modal fusion transformer for end-to-end autonomous driving. In *CVPR*. 7077–7087.

[47] Associated Press. 2022. *Nearly 400 car crashes in 11 months involved automated tech, companies tell regulators.* Retrieved Aug 5, 2023 from https://t.ly/UTb2e

[48] Rodrigo Queiroz et al. 2019. GeoScenario: An Open DSL for Autonomous Driving Scenario Representation. In *IV*. 287–294.

[49] Ina Schieferdecker and Andreas Hoffmann. 2012. Model-based testing. *IEEE software* 29, 1 (2012), 14–18.

[50] Sanjit A Seshia. 2017. Compositional verification without compositional specification for learning-based systems. *UC Berkeley* (2017), 1–8.

[51] Shital Shah et al. 2017. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Field and Service Robotics*. arXiv:arXiv:1705.05065 https://arxiv.org/abs/1705.05065

[52] Shai Shalev-Shwartz et al. 2017. On a formal model of safe and scalable self-driving cars. *arXiv preprint arXiv:1708.06374* (2017).

[53] Shivanshu Singh et al. 2016. Instrument malfunction during robotic surgery: A case report. *IJU* 32, 2 (2016), 159. https://doi.org/10.4103/0970-1591.174781

[54] Ian Sommerville. 2011. Software engineering (ed.). *America: Pearson Education Inc* (2011).

[55] Bernhard Steffen, Falk Howar, and Maik Merten. 2011. Introduction to active automata learning from a practical perspective. *SFM* (2011), 256–296.

[56] Tesla Autopilot 2023. *Tesla.* Retrieved Aug 3, 2023 from http://www.tesla.com/autopilot/

[57] Haoxiang Tian et al. 2022. MOSAT: finding safety violations of autonomous driving systems using multi-objective genetic algorithm. In *FSE*. 94–106. https://doi.org/10.1145/3540250.3549100

[58] Yuchi Tian et al. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *ICSE*. 303–314. https://doi.org/10.1145/3180155.3180220

[59] Hoang-Dung Tran et al. 2020. Verification of deep convolutional neural networks using imagestars. In *CAV*. Springer, 18–42.

[60] Uber ATG 2023. *Uber.* Retrieved Aug 3, 2023 from https://shorturl.at/mSV59

[61] The Verge. 2022. *Food delivery drone.* Retrieved Aug 5, 2023 from https://t.ly/ATsPu

[62] Junjie Wang et al. 2023. Software Testing with Large Language Model: Survey, Landscape, and Vision. *arXiv preprint arXiv:2307.07221* (2023).

[63] Waymo 2023. *Waymo.* Retrieved Aug 3, 2023 from http://www.waymo.com/

[64] Cerdic Wei Kit Wong. 2022. American fuzzy lop (AFL) fuzzer. (2022).

[65] Eleni Zapridou et al. 2020. Runtime verification of autonomous driving systems in CARLA. In *RV*. Springer, 172–183.

[66] Cen Zhang et al. 2023. Understanding Large Language Model Based Fuzz Driver Generation. *arXiv preprint arXiv:2307.12469* (2023).

[67] Qingwen Zhang et al. 2022. MMFN: Multi-Modal-Fusion-Net for End-to-End Driving. In *IROS*. IEEE, 8638–8643. https://doi.org/10.1109/IROS47612.2022.9981775

[68] Ziyuan Zhong et al. 2022. Neural network guided evolutionary fuzzing for finding traffic violations of autonomous vehicles. *TSE* (2022). https://doi.org/10.1109/TSE.2022.3195640

[69] Yuan Zhou et al. 2023. Specification-based Autonomous Driving System Testing. *TSE* (2023). https://doi.org/10.1109/TSE.2023.3254142

[70] Zipline 2023. *Zipline.* Retrieved Aug 3, 2023 from http://www.flyzipline.com/

Received 18-JAN-2024; accepted 2024-04-09