A Review of ”Compositional Falsification of Cyber-Physical Systems with Machine Learning Components”

Theo Olausson
School of Informatics
University of Edinburgh
Edinburgh, United Kingdom
tolausso@inf.ed.ac.uk

***Abstract*—Systems based on Artificial Intelligence (AI) have become commonplace in society, including those which are Cyber-Physical Systems (CPS) employing Machine Learning (ML) for tasks such as perception. As these systems become more important in society, so does making sure that they can be trusted with our well-being. In this paper, we provide an overview of the challenges in verifying AI-based CPS, basing our discussion on Seshia et al.'s influential paper *Towards Verified Artificial Intelligence* [1]. We then review one recent effort for the verification of CPS with ML components, Dreossi et al.'s *Compositional Falsification of Cyber-Physical Systems with Machine Learning Components* [2], initially summarising but then also critically evaluating its contributions. Finally, we outline future work in this space.**

***Index Terms*—artificial intelligence, machine learning, formal verification, cyber-physical systems**

I. Introduction: The problem of verifying AI-based Cyber-Physical Systems

Formal verification has been used to verify computational systems for a long time. Some of its biggest success stories include the verification of cache coherence protocols [3], hardware design (see [4] for a comprehensive survey), and cryptographic protocols [5].

More recently, formal verification has come to play an important role in *cyber-physical systems*: those which operate at the border between the cyber and physical worlds, such as medical devices or robots. The nature of these systems introduces several technical difficulties to the verification process, such as the stochasticity and poor observability of the environments in which they operate [6]. Perhaps even more challenging is the fact that these devices are often safety-critical. Consider for example an implanted device which automatically regulates a person's insulin levels by injecting insulin into their bloodstream, as might for example be necessary if the person is diabetic. When working correctly, such a device can be of unfathomable help to the person, but when it does not it may become a danger to their life; especially if the person has come to expect it to work at all times, and no longer regularly inspects its functionality.

The keen reader may have noticed another problem in this illustrative example: not only is the device safety-critical, but it is also an autonomous agent, independently choosing when to take action based on the inputs provided by its sensors. Increasingly, we are surrounding ourselves with AI-based systems which have to make split-second decisions (e.g. when to inject insulin into the bloodstream) without consulting a human. These systems come in a range of different shapes and appearances, from the invisible, purely digital systems which power the social networks to the large, autonomous vehicles which Uber, FiveAI, Waymo and countless other companies are developing. What they all have in common is that they have a profound impact on the lives of all of us, and as these systems come to play a larger and larger role in our lives, making sure that they are free from errors, secure from adversary attacks, and safe to use becomes increasingly important.

Given its success in other areas, formal verification could undeniably be of aid in the verification of AI-based systems. Unfortunately, as one might imagine, verifying that a self-driving car will not recklessly drive into pedestrians is a far harder problem than verifying that the caches of multi-processor systems will abide to their coherence protocol. In *Towards Verified Artificial Intelligence*, Seshia et al. [1] identify five main challenges which formal verification faces with AI-based systems, along with five principles they argue could (if researched further) be solutions (table I).

Problem	Solution
Environment Modeling	Introspective Environment Modeling
Formal Specification	End-to-End Specifications, Quantitative Specifications, Specification Mining
Modeling Systems that Learn	Formal Abstractions and Explanations for Machine Learning
Computational Engines for Training, Testing, and Verification	Randomized and Quantitative Formal Methods for Training, Testing and Verification
Correct-by-Construction Intelligent Systems	Formal Inductive Synthesis and Integrating Design Time with Run Time

TABLE I
The five challenges faced by Verified AI, along with principles which may serve as solutions to them [1].

Each of these challenges and proposed solutions are an advanced field in their own, and the literature already contains papers focusing on each of these in more detail (e.g. [7] for Formal Specification). In this review, we focus our gaze on the third problem: *Modeling Systems that Learn*.

The problem with modeling systems that learn is that, unlike in the traditional success stories of formal specification, the system behavior itself cannot easily be captured. Compare for example the task of verifying an integer addition component of a CPU with the task of verifying a deep neural network such as AlexNet [8]. The former consists of an electronic circuit described in some hardware description language, has a small, concrete input space, and its outputs can be described with the use of truth tables. The latter is a complex structure described by a series of 61 million parameters, operates on a much larger input space, and its outputs are the result of countless non-linear mathematical operations. Clearly, the latter places greater need on sophisticated abstraction techniques in order to capture its behavior, even if we ignore the fact that its behavior may continue to evolve as it gathers more data.

Sashia et al.'s suggested solution to the problem of modeling systems that learn, *Formal Abstractions and Explanations for Machine Learning*, revolves around two key research areas:

1) Developing new techniques which can automatically abstract ML components into formal representations for which efficient verification techniques can either easily be developed or already exist. This likely requires more work on probabilistic logics (e.g. *Probabilistic Signal Temporal Logic* [9]) and ways to propagate measures of uncertainty through the system (e.g. *Convex Markov Decision Processes* [10]).

2) Continuing to progress the field of *Explainable AI* (see [11] for a comprehensive survey), as generating explanations which are compatible with the modeling language can simplify the verification task; we will discuss the implications of this in more detail when outlining future research directions in section III.

In the rest of this paper, we will review one recent technical contribution in the first of these areas: Dreossi et al.'s paper *Compositional Falsification of Cyber-Physical Systems with Machine Learning Components* [2]. In this paper, the authors propose a compositional framework which joins together a signal temporal logic falsifier (the "CPS Analyzer") with a machine learning analyzing unit (the "ML Analyzer") to find falsifying executions of the CPS. In other words, they propose a method for finding example inputs which violate desired properties such as safety in systems that employ machine learning components. At the core of their framework is the division of the search space into that of the individual ML component and that of the rest of the CPS, ultimately joining the two to close the loop. Critically, their approach is agnostic to the type of machine learning component used, even being able to handle deep neural networks such as the previously mentioned AlexNet.

On a high level, Dreossi et al.'s method begins with having the CPS Analyzer identify a region of interest in its inputs, where it may be affected by a malfunctioning ML module. This is done by the CPS Analyzer considering abstractions of the ML component, for example one where it is always right and one where it is always wrong, so that it can operate on a lower-dimensional input space. Then, the identified region of interest is sent to the ML Analyzer which projects it onto the input space of the ML component, exploring the projected space in detail to find inputs which the ML component would misclassify. Finally, when the ML Analyzer finds example inputs which trigger misclassifications, it sends these back to the CPS Analyzer so that it can check whether this may lead to a system-level safety violation. If if finds that this is the case, the falsification has been successful; if not, the region of interest is adjusted.

This back-and-forth communication between the ML Analyzer and the CPS Analyzer allows Dreossi et al.'s framework to get around the intractability of the ML component's input space: note for example that something as seemingly simple as a 100×100 grey-scaled video input would lead to a feature space with $256^{100 \cdot 100}$ elements (assuming 8-bit pixel encoding). This is the key novel contribution of the paper, as it means the falsification procedure can be employed in CPS where the ML component is used for perception (typically leading to a very large input space). However, Dreossi et al. also present the ML Analyzer itself, including: a novel sampling-based approach for the exploration of the projected region of interest, a paramaterization used to abstract the ML input space and relate it to that of the entire CPS (necessary to project the region of interest), and an approximation-based method to identify misclassifications which may violate the desired property.

Dreossi et al.'s experimental evaluation focuses on two case studies of Advanced Emergency Breaking Systems (AEBS), one of a closed-loop Simulink model of the system and one where the system is deployed within a simulation engine. In all of Dreossi et al.'s experiments, the AEBS is assumed to use an ML component for perception, the output of which is then fed into a controller which decides whether an emergency breaking maneuver is needed. However, they vary the specific ML component used between a custom CNN architecture, AlexNet [8], and Inception-v3 [12] to showcase the flexibility of their framework. These case studies show that the framework is capable of generating counter-examples which break the AEBS regardless of which specific ML model is used for perception. However, they also show that not all misclassifications of the ML component affect the correctness of the CPS, and that this is dependent on the specific model used.

We have now set the scene for why AI-based Cyber-Physical Systems need to be verified. We have also discussed some of the main challenges faced in this field, and we have introduced one concrete attempt at resolving some of these. Without further ado, let us now continue to some more thorough critical evaluation of Dreossi et al.'s framework.

II. Critical Evaluation: Strengths and Weaknesses of Compositional Falsification of Cyber-Physical Systems with Machine Learning Components

We believe Dreossi et al.'s framework to be a big leap forward in the space of verifying Cyber-Physical Systems which rely on Machine Learning. In particular, we believe three properties of the suggested framework make it stand out in the literature:

1) As discussed before, the framework is completely agnostic to the specific details of the ML component, making no assumptions about its structure or mathematical detail. This allows it to handle arbitrarily complex ML models, and means the framework can easily be used with future ML techniques.

2) Although Dreossi et al. focus on formulations in Signal Temporal Logic and use Breach [13] as their CPS Analyzer, they also note that the framework is largely agnostic to these details. This means that as new temporal logics (and corresponding falsification engines) are developed, they can easily be integrated into the existing framework.

3) Critically, the compositional nature of the framework allows it to handle the intractably large input spaces associated with using Machine Learning models for perception; this means it is applicable to use cases such as autonomous driving, in which ML is frequently used for perception.

In summary, the greatest strength of the proposed framework is its flexibility. However, as always, it is not without its issues.

One limitation of the framework presented in this paper is that while the compositional nature of the framework allows the verification task to become tractable, it introduces a new problem into the picture: there is an implicit, and we believe poorly explored by the authors, relationship between the misclassification of the ML components and the correctness of the CPS at large. That is, the search for counter-examples is driven by generating inputs which cause misclassification of the ML component, yet the objective is to produce counter-examples of the entire CPS. This is somewhat alleviated by adjusting the region of interest when a candidate misclassification fails to cause a system-wide falsification, but even this effort is at risk of biasing the search with poorly calculated adjustments. Addressing this concern would require developing methods which search for violations of system safety directly on the closed-loop behaviour of the CPS. [14], [15] are two examples of such methods which have recently been suggested in the literature, although more research is needed in order to judge how the scalability and flexibility (in terms of use cases and applicable ML components) of these methods compare to that of Dreossi et al.'s framework.

Another shortcoming of the proposed framework is that while producing a falsifying counter-example is useful to the designer, it would be even more useful if this could be more tightly integrated into the design process of the ML component itself. Dreossi et al. acknowledge this shortcoming in their conclusions, and in a later paper the same UC Berkeley group present a methodology for using counter-examples to augment the training data [16], effectively patching holes found by the falsifier. Future work along this line of "closing the design loop" could greatly improve the practical usefulness of Dreossi et al.'s falsification framework; we will conclude section III of this paper with an ambitious view of how this might even be used to produce correct-by-construction intelligent systems.

Ultimately, the biggest drawback of the framework is that sampling-based falsification is only a partial step towards verification, since it can typically only be used to prove a system *incorrect*. Using falsification to prove correctness would require covering the entire input space, or at least covering a section of it which with other means could be proven to be sufficient. Compared to approaches presented in the literature which aim to carry out full-on verification by exploiting known properties of the ML component (such as the neural network activation functions, e.g. [17] for ReLU and [18] for sigmoid), the proposed approach thus trades coverage for flexibility. Whether this trade-off proves worth it or not largely depends on whether future progress in machine learning converges to a small set of fixed architectures which can be used across several tasks, in which case the cost of fine-tuning the verification process to the architecture may not be prohibitively expensive.

III. Future Work

While Dreossi et al.'s framework is an important step towards making sure that future autonomous Cyber-Physical Systems can safely be deployed, the preceding section shows that this problem is far from solved. In this section, we briefly discuss three potential directions for future work in this space.

First of all, we believe there is potential to join together efforts in logic-based machine learning with the current verification efforts discussed in this paper. Take for example *Neural Logic Reinforcement Learning* (NLRL) [19], a reinforcement learning algorithm which learns generalizable policies specified in first-order logic. If a CPS utilizing this algorithm was evaluated with Dreossi et al.'s falsification framework then the counter-examples produced would automatically be interpretable by virtue of consisting of first-order logical formulas, thus providing insight into *why* the component failed. Such joining together of sophisticated verification measures and explainable AI components was in fact precisely the principle which Seshia et al. identified as essential for modeling systems that learn [1]. However, it should be noted that methods such as NLRL are still in their infancy, and much work still remains before they could become relevant to real-world use cases such as CPS.

Secondly, we believe that it may be possible to use deep learning to more accurately search for falsifying counter-examples, or at least regions of interest in the input space, for a closed-loop CPS. Recent work has shown that deep learning can be used to assess risk in autonomous driving [20]. While not exactly the same, we conjecture that this problem is somewhat analogous to the problem of identifying regions of uncertainty in the CPS' input, since uncertainty should be correlated with risk. Although attempting to use such a method for verification would naturally introduce a chicken-and-egg problem (i.e., how do you verify the ML-based verification engine), it might still prove useful as an alternative way to perform the search for counter-examples.

Finally, and perhaps most ambitiously, we envision a future in which falsifying counter-examples become an integrated part of the training process of ML components, making reality of Seshia et al.'s dreams of so called correct-by-construction intelligent systems. Consider for example a future reinforcement learning algorithm, which takes as input a temporal logic formula to uphold (along with "liveness" constraints which prevent it from "doing nothing"). The algorithm then constructs a policy, and this policy is run through a falsification engine. If the falsification engine finds a counter-example, the policy is updated based on the counter-example, and the process starts over again; this continues until the falsification engine is unable to construct any more counter-examples, at which point the learned policy may be considered to have been formally verified (subject to the capabilities of the falsification engine). Of course, such an algorithm introduces several technical challenges. For example, modern (deep) learning methods rely on gradient-based methods to learn, and it is not at all clear how one might compute a gradient based on a given counter-example. In some sense, this is adversarial training [21] taken to its logical extreme.

IV. Conclusion

We began this short paper by giving an overview of the problem of verifying Artificial Intelligence in general and Cyber-Physical Systems with Machine Learning components in particular. We then reviewed one paper by Dreossi et al. [2] which targets the latter of these, summarizing its contributions before digging deeper into its strengths and weaknesses. Finally, we identified three novel future research directions which would bring us closer towards ensuring future AI-based systems are free from errors, secure from adversarial attacks, and safe to use.

References

[1] S. A. Seshia, D. Sadigh, and S. S. Sastry, "Towards Verified Artificial Intelligence," *ArXiv e-prints*, July 2016.

[2] T. Dreossi, A. Donzé, and S. A. Seshia, "Compositional falsification of cyber-physical systems with machine learning components," *CoRR*, vol. abs/1703.00978, 2017. [Online]. Available: http://arxiv.org/abs/1703.00978

[3] D. L. Dill, A. J. Drexler, A. J. Hu, and C. H. Yang, "Protocol verification as a hardware design aid," in *Proceedings 1992 IEEE International Conference on Computer Design: VLSI in Computers Processors*, 1992, pp. 522–525.

[4] C. Kern and M. R. Greenstreet, "Formal verification in hardware design: A survey," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 4, no. 2, p. 123–193, Apr. 1999. [Online]. Available: https://doi.org/10.1145/307988.307989

[5] C. Meadows, "Formal verification of cryptographic protocols: A survey," in *Proceedings of the 4th International Conference on the Theory and Applications of Cryptology: Advances in Cryptology*, ser. ASIACRYPT '94. Berlin, Heidelberg: Springer-Verlag, 1994, p. 135–150.

[6] P. Bagade, A. Banerjee, and S. Gupta, *Validation, Verification, and Formal Methods for Cyber-Physical Systems*, 01 2017, pp. 175–191.

[7] S. A. Seshia, A. Desai, T. Dreossi, D. J. Fremont, S. Ghosh, E. Kim, S. Shivakumar, M. Vazquez-Chanlatte, and X. Yue, "Formal specification for deep neural networks," in *Automated Technology for Verification and Analysis*, S. K. Lahiri and C. Wang, Eds. Cham: Springer International Publishing, 2018, pp. 20–34.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, May 2017. [Online]. Available: https://doi.org/10.1145/3065386

[9] D. Sadigh and A. Kapoor, "Safe control under uncertainty with probabilistic signal temporal logic," in *Proceedings of Robotics: Science and Systems XII*, June 2016. [Online]. Available: https://www.microsoft.com/en-us/research/publication/safe-control-uncertainty-probabilistic-signal-temporal-logic/

[10] A. Nilim and L. El Ghaoui, "Robust control of markov decision processes with uncertain transition matrices," *Oper. Res.*, vol. 53, no. 5, p. 780–798, Sep. 2005. [Online]. Available: https://doi.org/10.1287/opre.1050.0216

[11] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (xai)," *IEEE Access*, vol. 6, pp. 52 138–52 160, 2018.

[12] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015. [Online]. Available: http://arxiv.org/abs/1512.00567

[13] A. Donzé, "Breach, a toolbox for verification and parameter synthesis of hybrid systems," in *Proceedings of the 22nd International Conference on Computer Aided Verification*, ser. CAV'10. Berlin, Heidelberg: Springer-Verlag, 2010, p. 167–170. [Online]. Available: https://doi.org/10.1007/978-3-642-14295-6_17

[14] C. E. Tuncali, G. E. Fainekos, H. Ito, and J. Kapinski, "Simulation-based adversarial test generation for autonomous vehicles with machine learning components," *CoRR*, vol. abs/1804.06760, 2018. [Online]. Available: http://arxiv.org/abs/1804.06760

[15] C. E. Tuncali, J. Kapinski, H. Ito, and J. V. Deshmukh, "Reasoning about safety of learning-enabled components in autonomous cyber-physical systems," in *Proceedings of the 55th Annual Design Automation Conference*, ser. DAC '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3195970.3199852

[16] T. Dreossi, S. Ghosh, X. Yue, K. Keutzer, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Counterexample-guided data augmentation," *arXiv preprint arXiv:1805.06962*, 2018.

[17] X. Sun, H. Khedr, and Y. Shoukry, "Formal verification of neural network controlled autonomous systems," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 2019, pp. 147–156.

[18] R. Ivanov, J. Weimer, R. Alur, G. J. Pappas, and I. Lee, "Verisig: Verifying safety properties of hybrid systems with neural network controllers," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, ser. HSCC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 169–178. [Online]. Available: https://doi.org/10.1145/3302504.3311806

[19] Z. Jiang and S. Luo, "Neural logic reinforcement learning," *CoRR*, vol. abs/1904.10729, 2019. [Online]. Available: http://arxiv.org/abs/1904.10729

[20] P. Feth, M. N. Akram, R. Schuster, and O. Wasenmüller, "Dynamic risk assessment for vehicles of higher automation levels by deep learning," *CoRR*, vol. abs/1806.07635, 2018. [Online]. Available: http://arxiv.org/abs/1806.07635

[21] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "Ensemble adversarial training: Attacks and defenses," 2017.