2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks - Supplemental Volume (DSN-S) | 979-8-3503-2545-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/DSN-S58398.2023.00054

Rich and Expressive Specification of Continuous-Learning Cyber-Physical Systems

Thomas Flinkow*, Barak A. Pearlmutter*†, and Rosemary Monahan*†
*Department of Computer Science, Maynooth University, Maynooth, Ireland
†Hamilton Institute, Maynooth University, Maynooth, Ireland
Email: thomas.flinkow.2023@mumail.ie

***Abstract*—Neural networks are being applied to a growing variety of applications, including safety-critical domains like autonomous vehicles and aircraft, due to their ability to approximate complex functions from limited datasets and adapt by continuing to learn from real-world data after deployment.**

Ensuring dependability in cyber-physical systems with neural network controllers requires verification beyond linear input-output constraints and local robustness of the network in isolation and must address uncertainties introduced by learning and network behaviour on unseen data, especially when continuous learning after deployment is allowed. Verification of these continuous-learning cyber-physical systems requires assessing probabilistic, temporal, and behavioural specifications that express concepts like resilience and other key properties.

Our research seeks to develop strategies and associated tooling for rigorously specifying continuous-learning cyber-physical systems and verifying them against rich and expressive specifications.

***Index Terms*—Machine learning, neural networks, cyber-physical systems, formal specifications, formal verification**

I. Introduction

Recent advances in machine learning (ML) performance and capabilities have brought machine-learnt models to a level that is comparable to or better than humans at tasks such as pattern recognition and image classification [1], indicating great potential for incorporating machine-learnt models into cyber-physical systems (CPSs) [2]. CPSs combine continuous and discrete dynamics and usually consist of multiple modules, each exhibiting complex functionality. Integrating ML components into CPSs can lead to improved performance and adaptability, especially if the learning components are allowed to continue learning after deployment (which we will refer to as "continuous-learning" in the following). However, this learning process introduces uncertainties that must be addressed during verification.

As opposed to traditional software development, which often begins with formal requirements expressed in logic, ML is frequently used precisely when defining correct behaviour is infeasible, such as in object detection and classification tasks. In these situations, the only specification of correct behaviour is a finite ground-truth dataset, expecting the ML component to approximate the desired function and correctly generalise from that data to previously unseen data.

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant number 20/FFP-P/8853.

In order to ensure the dependability of continuous-learning CPSs and to guarantee that they are operating as intended, we first need to capture the complex behaviour of these systems through rigorous formal specification. Subsequently, we must establish techniques and associated tooling to verify these systems against specifications that go beyond simple input-output constraints and may include probabilistic, temporal, and behavioural specifications. In the following, we will refer to these specifications as "rich and expressive" to indicate their complex nature.

Our proposed research will focus on the following research questions:

- RQ1: What properties of continuous-learning CPSs need to be specified and verified, and how can they be categorised?
- RQ2: What is the best way to express and formalise rich and expressive properties?
- RQ3: What tooling can be provided to verify continuous-learning CPSs against these specifications?

II. Background

Significant research has in the past few years focused on verifying neural networks in isolation [2]. Given a neural network $\mathcal{N} : \mathcal{X} \to \mathcal{Y}$, arguing about correct behaviour often means reasoning about linear input-output constraints, that is, checking whether

$$\forall x \in \mathcal{X}.\ x \vDash \varphi \ \to\ \mathcal{N}(x) \vDash \psi, \tag{1}$$

given input and output specifications $\varphi \subseteq \mathcal{X}$ and $\psi \subseteq \mathcal{Y}$.

Often, one is also interested in assessing the robustness of the neural network to ensure that it generalises correctly. Local robustness requires the classification of a particular input x_0 to be stable to slight perturbations:

$$\forall x \in \mathcal{X}.\ \|x_0 - x\|_\infty \leq \varepsilon \ \to\ \mathcal{N}(x_0) = \mathcal{N}(x) \tag{2}$$

In recent years, an extensive ecosystem of tools for verifying neural networks against these properties has emerged, employing a wide variety of techniques such as constraint satisfiability solving through specialised SAT/SMT solvers. Prominent examples include Planet [3], Reluplex [4], and its successor Marabou [5], [6], which are sound and complete verifiers for verifying neural networks against linear input-output constraints and local robustness queries.

979-8-3503-2545-4/23/$31.00 ©2023 IEEE
DOI 10.1109/DSN-S58398.2023.00054

Other tools, such as Fast-Lin and Fast-Lip [7], and α, β-CROWN (based on CROWN [8], α-CROWN [9], β-CROWN [10], and GCP-CROWN [11]) focus on exploiting the network's internal structure to estimate robustness bounds and to efficiently compute tight output bounds. α, β-CROWN combines bounds propagation with efficiency optimisation techniques such as branch-and-bound and won the last two iterations of the International Neural Network Verification Competition (VNN-COMP) [12].

Another strand of tools focuses on output reachability analysis: ReachNN [13] computes the output reachable set of a neural network based control system using a sequence of transformations and overapproximations, and the NNV [14] tool propagates intervals of values through the neural network layers using a variety of abstract domains and set representations such as polyhedra, star sets, and image stars to obtain the exact output reachable set or an overapproximation thereof.

III. Preliminary Findings

Specifying and verifying only the previously mentioned input-output constraints and local robustness queries of the neural network component in isolation ("open-loop verification") fails to sufficiently capture the behaviour and safety properties of a continuous-learning CPS as a whole [15], [16].

Although some verification tools such as NNV, ReachNN, and Verisig [17] are able to perform closed-loop verification of neural networks controllers as part of learning-enabled CPSs, all verification tools we investigated appear to assume trained neural networks and do not target systems that continue to learn after deployment [18].

A promising approach for verifying learning itself are so-called differentiable logics such as DL2 [19] or those based on fuzzy logic [20], which can be used to impose constraints on the learning process by translating desired properties into additional loss terms. While originally proposed for improving accuracy in unsupervised or semi-supervised learning scenarios, their suitability to obtaining correct-by-construction machine-learnt models is an exciting direction that we will investigate.

Specifying and verifying continuous-learning CPSs should also include probabilistic reasoning (in the form of explicit probabilities or confidence levels [21]), and must assess more sophisticated temporal and behavioural properties such as resilience to stressors [22].

Hyperproperties are one type of more expressive safety and liveness properties as they relate multiple system executions to one another [23]. Logics for hyperproperties, such as HyperLTL [23] and HyperPCTL [24], have been developed along with decidability, equivalence and checking algorithms [25], [26] and [24] for fragments of these.

Hyperproperties have been shown to be useful for verification of CPSs [27], [28], and we would like to investigate their usefulness for specifying and verifying continuous-learning CPSs, as hyperproperties allow the specification of complex properties relevant in ML contexts, such as robustness for all inputs [29] or ε-differential privacy.

The concept of ε-differential privacy requires that the outcome S of a randomised algorithm A has a bounded probability of revealing which of two similar datasets D_1 and D_2 an input belongs to, formally expressed in (3).

$$\Pr[A(D_1) \in S] \leq e^{\varepsilon} \Pr[A(D_2) \in S] \tag{3}$$

Equation (4) shows HyperPCTL's natural and intuitive means to specify ε-differential privacy [24], closely aligning with the definition (3), where $\mathrm{adj}(\sigma, \sigma')$ describes two datasets that differ by one input, out represents the algorithm's outcome, and $\mathbb{P}(\varphi_\sigma)$ is the HyperPCTL operator denoting the probability of property φ being satisfied from state σ.

$$\begin{aligned} &\forall \sigma. \forall \sigma'.\ \mathrm{adj}(\sigma, \sigma') \rightarrow \\ &\qquad \left[\, \mathbb{P}(\Diamond(\mathsf{out} \in S)_\sigma) \leq e^{\varepsilon}\, \mathbb{P}(\Diamond(\mathsf{out} \in S)_{\sigma'}) \,\right] \end{aligned} \tag{4}$$

IV. Future Work

We plan to investigate how to express rich and expressive specifications for continuous-learning CPSs in a rigorous manner to sufficiently capture their safety behaviour. These rich and expressive specifications will include probabilistic, temporal and behavioural properties, or hyperproperties. Building on this foundation, we will explore techniques and associated tooling for verifying continuous-learning CPSs against these specifications in a next step.

Our research methodology continues with an extensive literature review to gather information on related work and existing approaches for verifying neural networks, both in isolation and as components of CPSs. We also survey the literature to evaluate and compare techniques for verifying systems that continue to learn after deployment.

Since the properties currently verifiable using popular neural network verification tools do not target the complex behaviour of continuous-learning CPSs, we plan to explore case studies and construct examples to compile a set of properties and specifications necessary for ensuring the safety of these systems.

We ultimately aim to develop a framework for specifying and verifying properties of continuous-learning CPSs to establish the foundation for their application in safety-critical domains.

The major contribution of our framework is the provision of support for both specification and verification of continuous-learning CPSs. To evaluate the effectiveness of our approach, we will apply our framework to case studies that require the expressive specification and verification of a CPS that continues to learn after deployment. An example is an autonomous robot that learns how to adapt to its environment.

Our evaluation will focus on quantifying improvements in the expressiveness of our specifications, the percentage of property violations that we detect through verification, the impact of the learning process on these verification metrics, as well as identifying limitations of our framework. The generalisation of our work to different categories of neural networks, different input modalities, and the regulations that govern these will be driven by use cases which we will explore with industrial collaborators and through integration with existing state-of-the-art tools.

REFERENCES

[1] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, Dec. 2020.

[2] W. Xiang, P. Musau, A. A. Wild, D. M. Lopez, N. Hamilton, X. Yang, J. Rosenfeld, and T. T. Johnson, "Verification for Machine Learning, Autonomy, and Neural Networks Survey," Oct. 2018, arXiv:1810.01989.

[3] R. Ehlers, "Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks," in *Automated Technology for Verification and Analysis*, ser. Lecture Notes in Computer Science, D. D'Souza and K. Narayan Kumar, Eds. Cham: Springer International Publishing, 2017, pp. 269–286.

[4] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, "Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks," in *Computer Aided Verification*, ser. Lecture Notes in Computer Science, R. Majumdar and V. Kunčak, Eds. Cham: Springer International Publishing, 2017, pp. 97–117.

[5] G. Katz, D. A. Huang, D. Ibeling, K. Julian, C. Lazarus, R. Lim, P. Shah, S. Thakoor, H. Wu, A. Zeljić, D. L. Dill, M. J. Kochenderfer, and C. Barrett, "The Marabou Framework for Verification and Analysis of Deep Neural Networks," in *Computer Aided Verification*, I. Dillig and S. Tasiran, Eds. Cham: Springer International Publishing, 2019, vol. 11561, pp. 443–452.

[6] Y. Y. Elboher, E. Cohen, and G. Katz, "Neural Network Verification Using Residual Reasoning," in *Software Engineering and Formal Methods*, ser. Lecture Notes in Computer Science, B.-H. Schlingloff and M. Chai, Eds. Cham: Springer International Publishing, 2022, pp. 173–189.

[7] L. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, L. Daniel, D. Boning, and I. Dhillon, "Towards Fast Computation of Certified Robustness for ReLU Networks," in *Proceedings of the 35th International Conference on Machine Learning*. PMLR, Jul. 2018, pp. 5276–5285.

[8] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, "Efficient neural network robustness certification with general activation functions," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18. Red Hook, NY, USA: Curran Associates Inc., Dec. 2018, pp. 4944–4953.

[9] K. Xu, H. Zhang, S. Wang, Y. Wang, S. Jana, X. Lin, and C.-J. Hsieh, "Fast and Complete: Enabling Complete Neural Network Verification with Rapid and Massively Parallel Incomplete Verifiers," Mar. 2021, arXiv:2011.13824.

[10] S. Wang, H. Zhang, K. Xu, X. Lin, S. Jana, C.-J. Hsieh, and J. Z. Kolter, "Beta-CROWN: Efficient Bound Propagation with Per-neuron Split Constraints for Complete and Incomplete Neural Network Robustness Verification," Oct. 2021, arXiv:2103.06624.

[11] H. Zhang, S. Wang, K. Xu, L. Li, B. Li, S. Jana, C.-J. Hsieh, and J. Z. Kolter, "General Cutting Planes for Bound-Propagation-Based Neural Network Verification," Dec. 2022, arXiv:2208.05740.

[12] C. Brix, M. N. Müller, S. Bak, T. T. Johnson, and C. Liu, "First Three Years of the International Verification of Neural Networks Competition (VNN-COMP)," Jan. 2023, arXiv:2301.05815.

[13] C. Huang, J. Fan, W. Li, X. Chen, and Q. Zhu, "ReachNN: Reachability Analysis of Neural-Network Controlled Systems," *ACM Transactions on Embedded Computing Systems*, vol. 18, no. 5s, pp. 106:1–106:22, Oct. 2019.

[14] H.-D. Tran, X. Yang, D. Manzanas Lopez, P. Musau, L. V. Nguyen, W. Xiang, S. Bak, and T. T. Johnson, "NNV: The Neural Network Verification Tool for Deep Neural Networks and Learning-Enabled Cyber-Physical Systems," in *Computer Aided Verification*, ser. Lecture Notes in Computer Science, S. K. Lahiri and C. Wang, Eds. Cham: Springer International Publishing, 2020, pp. 3–17.

[15] 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*, ser. HSCC '19. New York, NY, USA: Association for Computing Machinery, Apr. 2019, pp. 147–156.

[16] S. Bak and H.-D. Tran, "Neural Network Compression of ACAS Xu Early Prototype Is Unsafe: Closed-Loop Verification Through Quantized State Backreachability," in *NASA Formal Methods*, ser. Lecture Notes in Computer Science, J. V. Deshmukh, K. Havelund, and I. Perez, Eds. Cham: Springer International Publishing, 2022, pp. 280–298.

[17] 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*. Montreal Quebec Canada: ACM, Apr. 2019, pp. 169–178.

[18] H.-D. Tran, W. Xiang, and T. T. Johnson, "Verification Approaches for Learning-Enabled Autonomous Cyber–Physical Systems," *IEEE Design & Test*, vol. 39, no. 1, pp. 24–34, Feb. 2022.

[19] M. Fischer, M. Balunovic, D. Drachsler-Cohen, T. Gehr, C. Zhang, and M. Vechev, "DL2: Training and Querying Neural Networks with Logic," in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 1931–1941.

[20] N. Slusarz, E. Komendantskaya, M. L. Daggitt, and R. Stewart, "Differentiable Logics for Neural Network Training and Verification," Jul. 2022, arXiv:2207.06741.

[21] M. Farrell, A. Mavridou, and J. Schumann, "Exploring Requirements for Software that Learns: A Research Preview," in *Requirements Engineering: Foundation for Software Quality*, 2023, to appear.

[22] J. Laprie, "From Dependability to Resilience," in *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2008.

[23] M. R. Clarkson, B. Finkbeiner, M. Koleini, K. K. Micinski, M. N. Rabe, and C. Sánchez, "Temporal Logics for Hyperproperties," in *Principles of Security and Trust*, ser. Lecture Notes in Computer Science, M. Abadi and S. Kremer, Eds. Berlin, Heidelberg: Springer, 2014, pp. 265–284.

[24] E. Ábrahám and B. Bonakdarpour, "HyperPCTL: A Temporal Logic for Probabilistic Hyperproperties," in *Quantitative Evaluation of Systems*, ser. Lecture Notes in Computer Science, A. McIver and A. Horvath, Eds. Cham: Springer International Publishing, 2018, pp. 20–35.

[25] B. Finkbeiner, C. Hahn, and M. Stenger, "EAHyper: Satisfiability, Implication, and Equivalence Checking of Hyperproperties," in *Computer Aided Verification*, ser. Lecture Notes in Computer Science, R. Majumdar and V. Kunčak, Eds. Cham: Springer International Publishing, 2017, pp. 564–570.

[26] B. Finkbeiner and C. Hahn, "Deciding Hyperproperties," in *27th International Conference on Concurrency Theory (CONCUR 2016)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), J. Desharnais and R. Jagadeesan, Eds., vol. 59. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016, pp. 13:1–13:14.

[27] M. Anand, V. Murali, A. Trivedi, and M. Zamani, "Formal verification of hyperproperties for control systems," in *Proceedings of the Workshop on Computation-Aware Algorithmic Design for Cyber-Physical Systems*. Nashville Tennessee: ACM, May 2021, pp. 29–30.

[28] ——, "Verification of Hyperproperties for Uncertain Dynamical Systems via Barrier Certificates," Nov. 2021, arXiv:2105.05493.

[29] 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, vol. 11138, pp. 20–34.