Verification Approaches for Learning-Enabled Autonomous Cyber-Physical Systems

Hoang-Dung Tran, *Member, IEEE,* Weiming Xiang, *Senior Member, IEEE,* Taylor T. Johnson, *Member, IEEE*

Abstract—This paper presents an overview survey of verification techniques for autonomous systems, with a focus on safety-critical autonomous cyber-physical systems (CPS) and subcomponents thereof. Autonomy in CPS is enabled by recent advances in artificial intelligence (AI) and machine learning (ML) through approaches such as deep neural networks (DNNs), embedded in so-called learning enabled components (LECs) that accomplish tasks from classification to control. Recently, the formal methods and formal verification community has developed methods to characterize behaviors in these LECs with eventual goals of formally verifying specifications for LECs, and this article presents a survey of many of these recent approaches.

Index Terms—verification, machine learning, autonomy, cyber-physical systems

1 Introduction

Artificial intelligence (AI) is in a renaissance, and AI methods, such as machine learning (ML), are now at a level of accuracy and performance to be competitive or better than humans for many tasks. Deep neural networks (DNNs) in particular are increasingly effective at recognition and classification tasks. For instance, much of the sensing, estimation, and fusion of data that enables applications such as autonomous driving [1], aircraft collision avoidance [2] and other autonomous cyber-physical systems (CPS) [3] increasingly relies on DNNs and related ML techniques. However, this progress comes at significant risk when these methods are deployed in operational safety-critical systems, especially those without direct human supervision. Notably, it has been observed that neural networks can react in unexpected and incorrect ways to even slight perturbations of their inputs [4]. Therefore, there is a need for methods beyond testing [5] that can provide formal guarantees about the behavioral properties and specifications of autonomous CPS with learning-enabled components (LECs), especially for the purpose of safety assurance [6]. There are two main streams of verification for intelligent systems with machine learning components. The first one focuses on the correctness of only the learning components while the second one aims at proving the correctness of the whole system in which the learning-based component interacts with the physical dynamics of the system. As one of the most important and prevalent AI/ML techniques, this paper specifically surveys the current state-of-the-art for safety verification of autonomous CPS systems with neural network controllers.

Artificial neural networks are used in systems that introduce machine learning components to resolve complex problems. This can be attributed to the impressive ability of neural networks to approximate complex functions as shown by the Universal Approximation Theorem [7]. Neural networks are trained over a finite amount of input and output data, and are expected to generalize said data and produce desirable outputs for the given inputs and even for previously unseen inputs. The data-driven nature and lack of efficient methods for analysis of neural networks leads to, in most cases, the treatment of neural networks as black boxes with no assurance in safety. However, due to the rapid development artificial intelligence inspired applications, neural networks have recently been deployed in several safety-critical systems, such as autonomous systems [1], [3], and aircraft collision avoidance procedures [2]. Regrettably, it has been observed that neural networks can react in unexpected and incorrect ways to even slight perturbations of their inputs [4]. Thus, methods are needed that can provide formal guarantees about the behavioral properties of neural networks, especially for the purpose of safety assurance [6].[1]

2 Verification of Neural Networks

Verifying neural networks is a difficult problem, and it has been demonstrated that validating even simple properties about their behavior is NP-complete [10]. The difficulties encountered in verification mainly arise from the presence of activation functions and complex structures (i.e., an interconnected group of nodes, inspired by biological neurons) of neural networks. Moreover, neural networks are large-scale, nonlinear, non-convex, and often incomprehensible to humans so that traditional verification tools are not applicable for neural networks. The action of a neuron is described by the activation function in the form of $y_i = f(\sum_{j=1}^{n} \omega_{ij} x_j + \theta_i)$, where x_j is the jth input of

• *Hoang-Dung Tran and Taylor T. Johnson are with the Department of Electrical and Computer Science, Vanderbilt University, Nashville, TN, 37235.*
Weiming Xiang is with the School of Computer and Cyber Sciences, Augusta University, Augusta, GA, 30912.
E-mail: trhoangdung@gmail.com (H.-D. Tran), wxiang@augusta.edu (W. Xiang), taylor.johnson@gmail.com (T.T. Johnson)

1. This paper summarizes an extended survey of work in this area [8]. A related survey in the area is [9]. Source code for many of the methods described are available in our NNV tool (https://github.com/verivital/nnv), and other methods corresponding to [9] are also available (https://github.com/sisl/NeuralVerification.jl/).

the ith neuron, ω_{ij} is the weight from the jth input to the ith neuron, θ_i is called the bias of the ith neuron, y_i is the output of the ith neuron, and $f(\cdot)$ is the activation function. Typically, the activation function is either the rectified linear unit, logistic sigmoid, hyperbolic tangent, the exponential linear unit, or another linear function. In general, existing methods for neural network verification can be categorized into geometric (reachability) methods, mixed-integer linear programming (MILP) methods, satisfiability (SAT)-based and satisfiability modulo theory (SMT)-based methods, optimization-based methods, and others.

2.1 Geometric and Reachability Methods

To circumvent the difficulties brought by the nonlinearities present in the neural networks, many recent results focus on activation functions of the form $f(x) = \max(0, x)$, known as Rectified Linear Units (ReLUs). Taking advantage of the piecewise linear feature of ReLUs and considering the input as polyhedra or special classes of polyhedra, such as zonotopes or hyper-rectangles, the verification process can be turned into a sequence of operations on polyhedra. For instance, in [11], the computation process involves standard polytope operations, such as intersection and projection, and all of these can be computed by employing sophisticated computational geometry tools, such as MPT3 [12]. The essence of the approach is to be able to obtain an exact output set with respect to the input set. However, the number of polytopes involved in the computation process increases exponentially with the number of neurons in its worst case performance which makes the method not scalable to neural networks with a large number of neurons. Within the polyhedra computation framework, further developments have been made in the recent tool NNV [13], [14], [15], [16], [17] through using star sets, an efficient representation for convex sets, to enhance the scalability with respect to polyhedral operations. Due to the parallelizability of the approach, parallel computing techniques can be employed to speed up the computation to some extent.

In the framework of zonotopes, a verification engine for ReLU neural networks called AI^2 was proposed in [18]. In this approach, perturbed inputs and safety specifications are abstracted as zonotopes, and reasoning about their behaviors use operations over zonotopes. The framework AI^2 is capable of handling neural networks of size up to 50,000 neurons and, in particular, their approach has had success dealing with convolutional neural networks (CNNs). The star set approach within NNV has also recently been extended to CNNs using an extension of star sets to the computer vision domain called ImageStars [15], applied to large classification networks such as VGG16 and VGG19 that have tens-to-hundreds of millions of parameters [19]. Another special class of polyhedra that are called interval sets or hyper-rectangles is also considered for verification problems. These interval-based methods perform reachability analysis as the propagation of interval sets across hidden layers and eventually derive the output intervals. Specification-guided methods have been developed to provide an adaptive partitioning method for the input space [20]. By making use of the information of specification, unnecessary partitioning can be avoided so that the computational complexity can be reduced significantly. In [21], an interval symbolic method is developed to compute rigorous bounds for outputs of neural networks. Their approach is easily parallelizable and makes use of symbolic interval analysis in order to minimize overestimation (conservativeness). The authors implement their approach as part of ReluVal, a system for checking the security properties of ReLU-based neural networks.

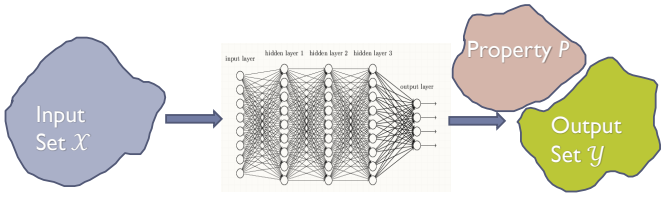


Fig. 1. Illustration of neural network reachability, where the output reachable set of a mathematical function $F : \mathbb{R}^n \mapsto \mathbb{R}^m$ representing the neural network's behavior under a set of inputs $\mathcal{X}$ is defined and computed in an exact or overapproximative manner. If $F(\mathcal{X}) = \mathcal{Y} \subseteq \neg\mathcal{P}$, then safety property $\neg P$ holds, while if $P \cap \mathcal{Y} \neq \emptyset$, then unsafe states may be reached.

2.2 MILP Methods

The use of binary variables to encode piecewise linear functions is standard in optimization [22]. In [23], the constraints of ReLU functions are encoded as an MILP. Combining output specifications that are expressed in terms of linear programming (LP), the verification problem for output set eventually turns to the feasibility problem of MILP. For layer i, the MILP encoding is given as

$$\begin{aligned} C_i = \{ & x_j^{[i]} \geq W_j^{[i]} x^{[i-1]} + \theta_j^i, \\ & x_j^{[i]} \leq W_j^{[i]} x^{[i-1]} + \theta_j^i + M\delta_j^{[i]}, \\ & x_j^{[i]} \geq 0, \\ & x_j^{[i]} \leq M(1 - \delta_j^{[i]}) \mid j = 1 \ldots \left| L^{[i]} \right| \} \end{aligned} \tag{1}$$

where M is sufficiently large so that it is larger than the maximum possible output at any node. A similar MILP problem is formulated in [24], where the authors conduct a robustness analysis and search for adversarial examples in ReLU neural networks. It is well known that MILP is an NP-hard problem and, in [25], [26], the authors elucidate significant efforts for solving MILP problems efficiently to make the approach scalable. Their methods combine MILP solvers with a local search yielding a more efficient solver for range estimation problems of ReLU neural networks than several other approaches. Basically, a local search is conducted using a gradient search and then a global search is formulated as MILP. Instead of finding the global optimum directly, it performs the search seeking values greater/smaller than the upper/lower bound obtained in the preceding local search. This is the primary reason for the computational complexity reduction. This MILP-based approach is integrated in a tool called Sherlock [27]. In [28], an MILP encoding scheme is used for a class of neural networks whose input spaces are encoded as binaries. This MILP encoding has a similar flavor to the other encodings present in the research literature for non-binarized networks. In their framework, since all the inputs are integer values, the real-valued variables can be

rounded so that they can be safely removed, resulting in a reformulated integer linear programming (ILP) problem that is smaller in comparison to the original MILP encoding. With the ILP encoding, an SAT solver is utilized in order to reason about the behavior of a binarized neural network of hundreds of neurons.

2.3 Satisfiability and SMT Methods

In [10], an SMT solver called Reluplex is developed. An algorithm, that stems from the Simplex algorithm for linear functions, for ReLU functions is proposed. Due to the piecewise linear feature of ReLU functions, each node is divided into two nodes. Thus, in their formulation, each node consists of a forward-facing and backward-facing node. If the ReLU semantics are not satisfied, two additional update functions are given to fix the mismatching pairs. Thus, the search process is similar to the Simplex algorithm that pivots and updates the basic and non-basic variables with the addition of a fixing process for ReLU activation pairs. This method is applied on a deep neural network implementation of a next-generation airborne collision avoidance system for unmanned aircraft (ACAS-X), which has been used as a benchmark for a number of successive works. In [29], Scheibler et al. use bounded model checking (BMC) to create formulas that are solved using the SMT-solver iSAT3, which is able to deal with transcendental functions, such as `exp` and `cos` (that exist in various activation functions) that frequently appear in neural network controllers and plant models. Although the verification framework is rigorously developed, the verification problem suffers scalability barriers due to the curse of dimensionality and state-space explosion problems. An approach for finding adversarial inputs using SMT solvers that relies on a layer-by-layer analysis is presented in [30]. The work focuses on the robustness of a neural network where safety is defined in terms of classification invariance within a small neighborhood of one individual input. An exhaustive search of the region is conducted by employing discretization and propagating the analysis layer by layer. In a similar manner, a recent paper, proposed by Ruan et al. [31], generalizes the local robustness criterion into a global notion on a set of test examples.

An early software tool in this area, called Planet, was developed based on the MILP verification approaches [32]. This LP-based framework combine SAT solving and linear over-approximation of piecewise linear functions in order to verify ReLU neural networks against convex specifications. Given the output of a ReLU denoted by d and the input $c \in [l, u]$, the relationship between c and d can be approximated by the linear constraints $d \geq 0$, $d \geq c$, and $d \geq u\frac{c-l}{u-l}$. Based on the LP problem formulation, additional heuristic algorithms were developed to detect infeasibility and imply phase inference faster. Pulina et al present an abstraction-refinement and SMT-based tool for verifying feed-forward neural networks. Their scheme is based on encoding the network into a boolean satisfaction problem over linear arithmetic constraints [33].

2.4 Other Optimization-Based Methods

As some of the earliest papers for neural network verification, in [34], [35], a piecewise-linearization of the nonlinear activation functions is used to reason about their behavior. In this framework, the authors replace the activation functions with piecewise constant approximations and use the bounded model checker hybrid satisfiability (HySAT) [36] to analyze various properties. The authors highlight the difficulty of scaling this technique and, currently, are only able to tackle small networks with at most 20 hidden nodes.

In [37], a simulation-based approach was developed, which used a finite number of simulations/computations to estimate the reachable set of multi-layer neural networks in a general form. Despite this success, the approach lacks the ability to resolve the reachable set computation problem for neural networks that are large-scale, non-convex, and nonlinear. Still, simulation-based approaches, like the one developed in [37], present a plausibly practical and efficient way of reasoning about neural network behavior. The critical step in improving simulation-based approaches is bridging the gap between finitely many simulations and the essentially infinite number of inputs that exist in the continuity set. A critical concept that is introduced in the work is called maximal sensitivity, which measures the maximal deviation of outputs for a set of inputs suffering disturbances in a bounded cell. The output set of the neural network can be over-approximated by the union of a finite number of reachtubes computed using a union of individual cells that cover the input set. Thus, verification of a network can be done by checking the existence of intersections of the estimated reachable set and safety regions. This approach has been extended to allow for the reachable set estimation and verification of nonlinear autoregressive-moving average (NARMA) models in the form of neural networks [38] as well as closed-loop system verification with the help of the state-of-the-art reachability tool for hybrid systems dealing with the plant dynamics [39].

In a recent result [40], an improved simulation-guided method is developed to reduce computational complexity. Unnecessary input partitions are avoided as the corresponding partition behaviors upon input space are guided by simulations instead of uniform partition. In particular, it is applicable to a variety of neural networks regardless of the specific form of the activation functions. Given a neural network, there is a trade-off between the precision of the reachable set estimation and the number of simulations used to execute the procedure. In addition, since the approach executes in a layer-by-layer manner, the approximation error will accumulate as the number of layers present in the network increases. In this case, more simulations are required at the expense of increasing the computational cost. A novel approach for neural network verification based on optimization duality has been developed [41]. The verification problem is posed as an optimization problem that tries to find the largest violation of a property related to the output of the network.

2.5 Other Methods

There exists a rich literature of other methods for neural network verification [8], [9], but we highlight a few. A comparison of the verification approaches mentioned above can be found in [42]. Additionally, the authors present a novel approach for neural network verification called

Branch and Bound Optimization. This approach adds one more layer behind the output layer $cy - b$ to represent the linear property $cy > b$ that we wish to verify. If $cy - b > 0$, it means that the property is satisfied, otherwise it is unsatisfiable. Thus, the verification problem is converted into a computation of the minimum or maximum value of the output of the neural network. By treating the neural network as a nonlinear function, model-free optimization methods are utilized to find optimal solutions. In order to have a global optimum, the input space is also discretized into subregions. This approach is not only applicable to ReLU neural networks, but the model-free method allows the approach to be applied to neural networks with more general activation functions. However, despite its generalization capabilities, in the model-free framework, there is no guarantee that the algorithm will converge to a solution.

Cheng et al. studied the verification of Binarizied neural networks (BNNs) [43]. The forward propagation of input signals is reduced to bit arithmetic. The authors argue that the verification of BNNs can be reduced to hardware verification and represents a more scalable problem than traditional neural network verification. A randomized approach for rigorously verifying neural networks in safety-critical applications has been developed [44]. In an effort to mitigate challenges related to the curse of dimensionality, the authors make use of Monte Carlo methods to estimate the probability of neural network failure. However, although Monte Carlo methods are more efficient than methods that deterministically search through hyper-rectangular input spaces, they are probabilistic in nature. The authors further demonstrate that although the number of samples needed to guarantee this may be large, it is not as prohibitive as other methods.

In addition to neural network verification, there are also results on falsification and testing of neural networks. Several ideas for integrating semantics into adversarial learning have been explored, including a semantic modification space and the use of more detailed information about the outputs produced by machine learning models [45]. In work by Tsui Weng et al. [46], an attack independent robustness metric against adversarial examples for neural networks is described. Their approach converts the robustness analysis into a local Lipschitz constant estimation problem and uses Extreme Value Theory for efficient solving. In [47], an automatic test case generator is presented that leverages real-world changes in driving conditions like rain, fog, lighting conditions, etc. The tool, called DeepTest, systematically explores different parts of the deep neural network logic by generating test inputs that maximize the number of activated neurons. An improved version of the tool, called DeepXplore, is proposed in [48], which is the first efficient whitebox testing framework for large-scale deep learning systems.

3 Verification and Fasification of Neural Network Control Systems

Verification and falsification of feedback neural network control systems (NNCS) have become an emerging research topic recently. Unlike verification of a neural network where the specifications of interest are usually defined as

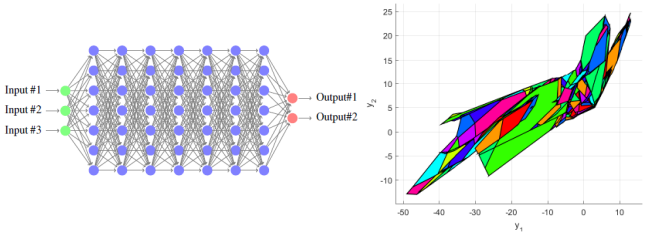


Fig. 2. Example output reachable set computation for a neural network with 3 inputs, 2 outputs, and 7 hidden layers with 7 neurons each, where all activation functions are ReLUs and all parameters of the network (weights, biases) are chosen randomly. The input set $I = \{x \mid \|x\|_\infty \leq 1, x \in \mathbb{R}^3\}$ is a cube and convex, while the output set shown is non-convex, represented as the union of the different colored polygons.

predicates over the outputs of the network, in NNCS, the specification are usually defined based on the states of the plant controlled by a neural network controller. Notably, the behavior of the whole feedback control system depends not only on the behavior of the neural network controller but also the system's physical dynamics which is usually described in terms of ordinary differential equations (ODEs). The interaction between the nonlinear neural network controller and the physical dynamics makes the behavior of the whole system complicated and difficult to analyze. To overcome this challenge, several methods have been proposed recently to verify system-level safety properties of NNCS with *feed-forward neural network controllers*.

The polyhedron-based approach [11] has been extended for safety verification of NNCS with *linear and discrete* dynamics [49]. Recently, a hybridization approach has been proposed in the Verisig tool [50] that transforms a NNCS to an equivalent nonlinear hybrid system that can be verified using Flow* [55], a verification tool for nonlinear hybrid systems. This approach applies for neural network controllers with smooth activation functions, such as Sigmoid and hyperbolic tangent (Tanh). Sound and complete satisfiability modulo convex (SMC)-based approaches for formal verification of NNCS have been developed, in which the closed-loop neural network control system with *linear and discrete* dynamics is encoded as monotone SMC formulas that were formally verified by SMC decision procedures [51]. In [52], a new abstraction method has been proposed for NNCS verification in which an "local" Taylor model over-approximation of neural network controller was obtained and integrated in Flow* [55] to compute a tight over-approximation reachable set of NNCS. The advantage of this method is that it fast and scalable and more importantly, it is proposed to reduces significantly over-approximation errors in reachable set computation process. In [54], a new reachability approach based on Bernstein polynomials has been proposed to verify NNCS with more general of activation functions. This approach can control the over-approximation error in the analysis, however, the cost of being more accurate is increased computation time.

An extension of the star-based reachability analysis method for neural networks has been implemented in NNV [53] to verify safety properties of NNCS. The star set method can deal with different activation functions, such as ReLU, Satlin, Sigmoid, and Tanh, as well as different

Approaches	Plant Dynamics	Discrete/Continuous	Activation Function
Polyhedron-based [49]	Linear	Discrete	ReLU
Verisig [50]	Linear, Nonlinear	Discrete, Continuous	Sigmoid, Tanh
SMC-based [51]	Linear	Discrete	ReLU
Sherlock [52]	Linear, Nonlinear	Discrete, Continuous	ReLU
NNV [14], [16], [53]	Linear, Nonlinear	Discrete, Continuous	ReLU, Satlin, Sigmoid, Tanh
ReachNN [54]	Nonlinear	Continuous	ReLU, Sigmoid, Tanh

TABLE 1
Reachability and bounded-model checking approaches for neural network control system verification.

types of dynamics, i.e., linear or nonlinear in discrete or continuous time domains. The star set method can perform exact and complete analysis of NNCS with linear discrete dynamical plants and neural network controllers with ReLU/Satlin activation functions. The extended star set method has successfully verified safety properties of Advanced Emergency Braking Systems (AEBS) and Adaptive Cruise Control Systems (ACCS), in which the size of the neural network controller ranges from fifty to two hundreds neurons.

A summary of recent verification methods is given in Table 1, which focuses on reachability methods that can provide finite time-horizon guarantees, although there also exist approaches based on barrier certificates (a "continuous" form of the classical inductive invariance proof rule) that may provide infinite time-horizon guarantees [56]. Although verification for NNCS provides sound guarantees for safety, it is usually computationally expensive and suffers from scalability challenges. Importantly, due to scalability limitations, current state-of-art verification techniques cannot deal with NNCS with perception components. In this case, falsification approach plays an important role since it is more scalable and applicable than the verification approaches. Particularly, in [57], a compositional falsification framework for CPS with machine learning components has been developed. In this framework, a temporal logic falsifier cooperates efficiently with a machine learning analyzer to find falsifying executions of the system. The effectiveness of the proposed framework was shown via the falsification of AEBS.

4 Challenges and Future Directions

Scalability vs. Conservativeness. Scalability is still a major challenge for most existing verification techniques. It has been shown that the verification time using exact analysis increases exponentially [10], [14]. Particularly, besides the size of the network, the input set is an important factor affecting the verification time of the exact analysis method. Generally, a large network or a large input set requires more verification time. To improve scalability, a large body of research in neural network verification relies on over-approximation methods. Some recent approaches [58], [59] are optimistic about the scalability of their methods. However, it has been shown that these methods only can deal with a small input set due to the explosion of over-approximation error in the analysis, which leads to conservative reachable sets [14]. In the future, we believe that new hybrid techniques that can combine the advantages of exact and over-approximate analyses are needed to improve both scalability and conservativeness in neural network verification.

Formal specifications and compositional verification. While a large body of research focuses on verifying neural networks and NNCS, fewer works investigate specification formalization for such systems [60], [61], [62]. For neural network verification, most current methods investigate safety and robustness properties, which can be specified as input to output relations of neural networks as illustrated in Figure 1 [60], [62]. For NNCS verification, existing approaches deal with safety specifications defined as predicates over the states of the plant model. In the real-world, learning-enabled CPS are complex in which several LECs, such as perception components and neural network controllers, interact with each other and the physical world, such as between a physical plant and its environment.

Defining meaningful system-level specifications for the whole system is relatively straightforward (such as collision avoidance), but the implications and constraints such system-level specifications place on LECs, especially those for perception, is non-trivial and needs to be investigated deeply. New specification languages for learning-enabled CPS are crucial to formally define the behavior of the systems and their subcomponents, and equally important, is defining libraries of specifications for meaningful perception problems, such as classification, semantic segmentation, and object detection/localization. One promising direction is to utilize hyperproperties for specifying robustness to adversarial perturbations [60], [63]. A further challenge, particularly related to perception, is not only in defining specifications, but in evaluating specifications with respect to meaningful environmental scenarios and data. This challenge is fundamentally different than the typical approach for verification of closed-loop systems, where a plant model generates new inputs for a controller, and instead requires verification with respect to pre-recorded environmental data (such as images/video) or generation thereof. This is partly because it is unreasonable to expect formal models for the environment in which an NNCS operates, and at best, generative models such as GANs and realistic simulators may exist, beyond pre-recorded real-world data. Altogether, it is unclear under what circumstances compositional specification and verification for learning-enabled CPS are achievable, such as by verifying individual LECs and attempting to compose guarantees of individual components into system-level guarantees [64].

Runtime verification for NNCS. Existing verification techniques for NNCS primarily operate offline and are to be performed during the design-time of a system. In practice, it is useful to have techniques that can monitor, if not verify, NNCS specifications online. Based on the online verification

information, a system can perform some intelligent actions to avoid upcoming difficult or catastrophic circumstances, such as hitting an obstacle or colliding with another system, for instance with Simplex architecture approaches [65].

Robust and safe learning. All techniques surveyed in this paper deal with an existing network or NNCS. In the future, new learning methods that integrate verification techniques in the training process to enhance the robustness of a network or the safety of a NNCS are essential for applying neural networks in safety-critical applications, such as in recent approaches for safe reinforcement learning.

Benchmarking and Standardization. A major limiting factor in the development of this area is a lack of standardization for formal models and specifications. There are several ongoing initiatives that aim to address this shortcoming to enable easier, fairer, and more scientific comparisons between the existing verification methods, as well as future ones. Due to this lack of standardization, this article does not make specific claims relating to which methods are most appropriate or scalable, as such answers are not yet known. For the open-loop verification problem (Section 2), the Verification of Neural Networks (VNN) workshop hosted the first competition on neural network verification (VNN-COMP) in 2020[2]. For the closed-loop verification problem (e.g., for NNCS as in Section 3), the Applied Verification of Continuous and Hybrid Systems (ARCH) workshop hosted the first AINNCS category verification competition in 2019 [66]. Other efforts, such as standardization of models (e.g., in Open Neural Network Exchange (ONNX)[3]), specifications, etc., are emerging, such as through the usage of HyST hybrid automata for ARCH-COMP [67], and the development of the VNN-LIB[4], an effort like that of SMT-LIB [68] for satisfiability and SMT problems.

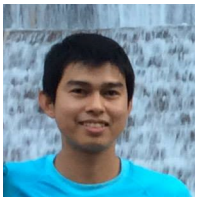
5 Conclusions

This paper has surveyed recent approaches for verifying machine learning components, specifically neural networks, that are crucial to enabling autonomy in CPS, but that suffer from well-known robustness problems. Numerous avenues for future work exist, ranging from runtime verification and assurance approaches to assure autonomy during system operation, to expanded the types of LECs beyond feedforward neural networks for which most existing verification approaches target.

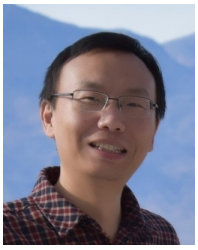
Acknowledgments

The material presented in this paper is based upon work supported by the Air Force Office of Scientific Research (AFOSR) through contract number FA9550-18-1-0122 and the Defense Advanced Research Projects Agency (DARPA) through contract number FA8750-18-C-0089. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFOSR or DARPA.

2. https://sites.google.com/view/vnn20/
3. https://onnx.ai/
4. http://www.vnnlib.org/



Hoang-Dung Tran Dr. Hoang-Dung Tran will join the Department of Computer Science and Engineering at the University of Nebraska, Lincoln, as an Assistant Professor in August 2020. He earned his Ph.D. degree in Computer Science from Vanderbilt University in the same year. His research interests are in formal verification of autonomous cyber-physical systems with learning-enabled components, safe artificial intelligence, hybrid, and switching systems, distributed control systems. He is also interested in robust control, stability analysis of nonlinear control systems, and networked control systems.



Weiming Xiang Dr. Weiming Xiang is currently an Assistant Professor in the School of Computer and Cyber Sciences at Augusta University. Dr. Xiang earned his PhD degree from Southwest Jiaotong University in 2014 with his PhD thesis on formal methods for hybrid traffic systems awarded the Outstanding PhD Dissertation of Southwest Jiaotong University 2014. Before joining Augusta University, Dr. Xiang worked as a Postdoctoral Research Scholar in the Department of Electrical Engineering and Computer Sciences at Vanderbilt University from September 2016 to July 2019, a Postdoctoral Research Associate in the Department of Computer Science and Engineering at the University of Texas at Arlington from November 2015 to August 2016, a Research Associate in the Department of Mechanical Engineering at the University of Hong Kong from May 2015 to October 2015. Dr. Xiang's research interest is developing formal synthesis and verification techniques and software tools for Cyber-Physical Systems (CPS). His current research centers on formal methods on safety, security and reliability of learning-enabled CPS. He is also broadly interested in methods and applications across CPS domains, such as control synthesis, stability analysis, reachable set computation, hybrid systems, power and energy, transportation, fuzzy logic, and neural networks. Dr. Xiang was an Associate Editor of Neurocomputing (2014-2019), and he was the Leading Guest Editor of Special Issue: Recent Advances in Control and Verification for Hybrid Systems in IET Control Theory and Applications and he is currently on the Editorial Board of IET Control Theory and Applications. Dr. Xiang is an IEEE Senior Member.

Taylor T. Johnson Dr. Taylor T. Johnson (S'05M'13) received the M.Sc. and Ph.D. degrees from the University of Illinois at Urbana-Champaign, Urbana, IL, USA, in 2010 and 2013, respectively, both in electrical and computer engineering. He is currently an Assistant Professor of electrical engineering and computer science at the Vanderbilt University, Nashville, TN, USA. His research interests include developing algorithmic techniques and software tools to improve the reliability of cyber-physical systems. Dr. Johnson received the Air Force Office of Scientific Research Young Investigator Program Award in 2018 and the National Science Foundation Computer and Information Science and Engineering Research Initiation Initiative Award in 2015.

REFERENCES

[1] M. Bojarski, D. Del Testa *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[2] K. Julian, J. Lopez, J. Brush, M. Owen, and M. Kochenderfer, "Policy Compression for Aircraft Collision Avoidance Systems," in *35th Digital Avionics Systems Conference. (DASC)*, 2016, pp. 1–10.

[3] K. Julian and M. J. Kochenderfer, "Neural Network Guidance for UAVs," in *AIAA Guidance Navigation and Control Conference (GNC)*, 2017.

[4] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *CoRR*, vol. abs/1312.6199, 2013. [Online]. Available: http://arxiv.org/abs/1312.6199

[5] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning testing: Survey, landscapes and horizons," *IEEE Transactions on Software Engineering*, 2020.

[6] F. Leofante, N. Narodytska, L. Pulina, and A. Tacchella, "Automated Verification of Neural Networks: Advances, Challenges and Perspectives," *ArXiv e-prints*, May 2018.

[7] K. Hornik, M. Stinchcombe, and H. White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[8] W. Xiang, P. Musau, A. A. Wild, D. M. Lopez, N. Hamilton, X. Yang, J. A. Rosenfeld, and T. T. Johnson, "Verification for machine learning, autonomy, and neural networks survey," *CoRR*, vol. abs/1810.01989, 2018. [Online]. Available: http://arxiv.org/abs/1810.01989

[9] C. Liu, T. Arnon, C. Lazarus, C. Barrett, and M. J. Kochenderfer, "Algorithms for verifying deep neural networks," *CoRR*, vol. abs/1903.06758, 2019. [Online]. Available: http://arxiv.org/abs/1903.06758

[10] G. Katz, C. Barrett, D. Dill, K. Julian, and M. Kochenderfer, "Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks," in *International Conference on Computer Aided Verification*. Springer, 2017, pp. 97–117.

[11] W. Xiang, H.-D. Tran, and T. T. Johnson, "Reachable Set Computation and Safety Verification for Neural Networks with ReLU Activations," *arXiv preprint arXiv: 1712.08163*, 2017.

[12] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proceedings of the European Control Conference*, Zürich, Switzerland, July 17–19 2013, pp. 502–510, http://control.ee.ethz.ch/~mpt.

[13] H.-D. Tran, P. Musau, D. M. Lopez, X. D. Yang, L. Nguyen, W. Xiang, and T. T. Johnson, "Parallelizable reachability analysis algorithms for feed-forward neural networks," in *FormaliSE 2019*, 2019.

[14] H.-D. Tran, P. Musau, D. M. Lopez, X. Yang, L. V. Nguyen, W. Xiang, and T. T. Johnson, "Star-based reachability analysis for deep neural networks," in *23rd International Symposium on Formal Methods (FM'19)*. Springer International Publishing, October 2019.

[15] H.-D. Tran, S. Bak, W. Xiang, and T. T. Johnson, "Verification of deep convolutional neural networks using imagestars," in *32nd International Conference on Computer-Aided Verification (CAV)*, 2020.

[16] H. D. Tran, X. Yang, D. M. 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 *32nd International Conference on Computer-Aided Verification (CAV)*, 2020.

[17] S. Bak, H.-D. Tran, K. Hobbs, and T. T. Johnson, "Improved geometric path enumeration for verifying ReLU neural networks," in *32nd International Conference on Computer-Aided Verification (CAV)*, July 2020.

[18] T. Gehr, M. Mirman, D. D. Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev, "AI^2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation," *IEEE Symposium on Security and Privacy*, vol. 39, May 2018.

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[20] W. Xiang, H.-D. Tran, and T. T. Johnson, "Specification-guided safety verification for feedforward neural networks," *arXiv preprint arXiv:1812.06161*, 2018.

[21] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Formal security analysis of neural networks using symbolic intervals," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 1599–1614.

[22] R. J. Vanderbei., *Linear Programming: Foundations & Extensions (Second Edition)*. Springer, 2001.

[23] A. Lomuscio and L. Maganti, "An approach to reachability analysis for feed-forward ReLU neural networks," *CoRR*, vol. abs/1706.07351, 2017. [Online]. Available: http://arxiv.org/abs/1706.07351

[24] V. Tjeng and R. Tedrake, "Verifying Neural Networks with Mixed Integer Programming," *arXiv preprint arXiv:1711.07356*, 2017.

[25] S. Dutta, S. Jha, S. Sanakaranarayanan, and A. Tiwari, "Output Range Analysis for Deep Neural Networks," *arXiv preprint arXiv:1709.09130*, 2017.

[26] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Output Range Analysis for Deep Feedforward Neural Networks," in *NASA Formal Methods*, A. Dutle, C. Muñoz, and A. Narkawicz, Eds. Cham: Springer International Publishing, 2018, pp. 121–138.

[27] S. Dutta, X. Chen, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Sherlock-a tool for verification of neural network feedback systems: demo abstract," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*. ACM, 2019, pp. 262–263.

[28] N. Narodytska, S. P. Kasiviswanathan, L. Ryzhyk, M. Sagiv, and T. Walsh, "Verifying Properties of Binarized Deep Neural Networks," *arXiv preprint arXiv:1709.06662*, 2017.

[29] K. Scheibler, L. Winterer, R. Wimmer, and B. Becker, "Towards Verification of Artificial Neural Networks." in *MBMV*, 2015, pp. 30–40.

[30] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety Verification of Deep Neural Networks," *CoRR*, vol. abs/1610.06940, 2016. [Online]. Available: http://arxiv.org/abs/1610.06940

[31] W. Ruan, M. Wu, Y. Sun, X. Huang, D. Kroening, and M. Kwiatkowska, "Global Robustness Evaluation of Deep Neural Networks with Provable Guarantees for L_0 Norm," *ArXiv e-prints*, Apr. 2018.

[32] R. Ehlers, "Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks," *CoRR*, vol. abs/1705.01320, 2017. [Online]. Available: http://arxiv.org/abs/1705.01320

[33] L. Pulina and A. Tacchella, "NeVer: a tool for artificial neural networks verification," *Annals of Mathematics and Artificial Intelligence*, vol. 62, no. 3, pp. 403–425, Jul 2011.

[34] ——, "An Abstraction-Refinement Approach to Verification of Artificial Neural Networks," in *Computer Aided Verification*, T. Touili, B. Cook, and P. Jackson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 243–257.

[35] ——, "Challenging SMT solvers to verify neural networks," *AI Communications*, vol. 25, no. 2, pp. 117–135, 2012.

[36] M. Fränzle and C. Herde, "HySAT: An efficient proof engine for bounded model checking of hybrid systems," *Formal Methods in System Design*, vol. 30, no. 3, pp. 179–198, Jun 2007.

[37] W. Xiang, H.-D. Tran, and T. T. Johnson, "Output Reachable Set Estimation and Verification for Multi-Layer Neural Networks," *IEEE Transactions on Neural Network and Learning Systems*, vol. 29, no. 11, pp. 5777–5783, 2018.

[38] W. Xiang, D. M. Lopez, P. Musau, and T. T. Johnson, "Reachable set estimation and verification for neural network models of nonlinear dynamic systems," in *Safe, Autonomous and Intelligent Vehicles*. Springer, 2019, pp. 123–144.

[39] W. Xiang and T. T. Johnson, "Reachability analysis and safety verification for neural network control systems," *arXiv preprint arXiv:1805.09944*, 2018.

[40] W. Xiang, H. Tran, X. Yang, and T. T. Johnson, "Reachable set estimation for neural network control systems: A simulation-guided approach," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[41] K. Dvijotham, R. Stanforth, S. Gowal, T. Mann, and P. Kohli, "A Dual Approach to Scalable Verification of Deep Networks," *arXiv preprint arXiv:1803.06567*, 2018.

[42] R. Bunel, I. Turkaslan, P. H. Torr, P. Kohli, and M. P. Kumar, "Piecewise linear neural network verification: A comparative study," *arXiv preprint arXiv:1711.00455*, 2017.

[43] C. Cheng, G. Nührenberg, and H. Ruess, "Verification of Binarized Neural Networks," *CoRR*, vol. abs/1710.03107, 2017. [Online]. Available: http://arxiv.org/abs/1710.03107

[44] R. R. Zakrzewski, "Randomized Approach to Verification of Neural Networks," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, vol. 4, July 2004, pp. 2819–2824 vol.4.

[45] T. Dreossi, S. Jha, and S. A. Seshia, "Semantic Adversarial Deep Learning," *ArXiv e-prints*, Apr. 2018.

[46] T.-W. Weng, H. Zhang, P.-Y. Chen, J. Yi, D. Su, Y. Gao, C.-J. Hsieh, and L. Daniel, "Evaluating the Robustness of Neural Networks: An Extreme Value Theory Approach," *ArXiv e-prints*, Jan. 2018.

[47] Y. Tian, K. Pei, S. Jana, and B. Ray, "DeepTest: Automated testing of deep-neural-network-driven autonomous cars," *CoRR*, vol. abs/1708.08559, 2017. [Online]. Available: http://arxiv.org/abs/1708.08559

[48] K. Pei, Y. Cao, J. Yang, and S. Jana, "DeepXplore: Automated Whitebox Testing of Deep Learning Systems," *CoRR*, vol. abs/1705.06640, 2017. [Online]. Available: http://arxiv.org/abs/1705.06640

[49] W. Xiang, H.-D. Tran, J. A. Rosenfeld, and T. T. Johnson, "Reachable set estimation and safety verification for piecewise linear systems with neural network controllers," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 1574–1579.

[50] 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*. ACM, 2019, pp. 169–178.

[51] 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*. ACM, 2019, pp. 147–156.

[52] S. Dutta, X. Chen, and S. Sankaranarayanan, "Reachability analysis for neural feedback systems using regressive polynomial rule inference," in *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*. ACM, 2019, pp. 157–168.

[53] H.-D. Tran, F. Cei, D. M. Lopez, T. T. Johnson, and X. Koutsoukos, "Safety verification of cyber-physical systems with reinforcement learning control," in *ACM SIGBED International Conference on Embedded Software (EMSOFT'19)*. ACM, October 2019.

[54] 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 (TECS)*, vol. 18, no. 5s, pp. 1–22, 2019.

[55] X. Chen, E. Ábrahám, and S. Sankaranarayanan, "Flow*: An analyzer for non-linear hybrid systems," in *International Conference on Computer Aided Verification*. Springer, 2013, pp. 258–263.

[56] C. E. Tuncali, H. Ito, J. Kapinski, and J. V. Deshmukh, "Invited: Reasoning about safety of learning-enabled components in autonomous cyber-physical systems," in *2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC)*, June 2018, pp. 1–6.

[57] T. Dreossi, A. Donzé, and S. A. Seshia, "Compositional falsification of cyber-physical systems with machine learning components," in *NASA Formal Methods Symposium*. Springer, 2017, pp. 357–372.

[58] G. Singh, T. Gehr, M. Püschel, and M. Vechev, "An abstract domain for certifying neural networks," *Proceedings of the ACM on Programming Languages*, vol. 3, no. POPL, pp. 1–30, 2019.

[59] G. Singh, T. Gehr, M. Mirman, M. Püschel, and M. Vechev, "Fast and effective robustness certification," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 802–10 813.

[60] 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 *International Symposium on Automated Technology for Verification and Analysis*. Springer, 2018, pp. 20–34.

[61] D. J. Fremont, T. Dreossi, S. Ghosh, X. Yue, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, "Scenic: a language for scenario specification and scene generation," in *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2019, pp. 63–78.

[62] T. Dreossi, S. Ghosh, A. Sangiovanni-Vincentelli, and S. A. Seshia, "A formalization of robustness for deep neural networks," *arXiv preprint arXiv:1903.10033*, 2019.

[63] L. V. Nguyen, J. Kapinski, X. Jin, J. V. Deshmukh, and T. T. Johnson, "Hyperproperties of real-valued signals," in *Proceedings of the 15th ACM-IEEE International Conference on Formal Methods and Models for System Design*, ser. MEMOCODE 17. New York, NY, USA: Association for Computing Machinery, 2017, p. 104113.

[64] S. A. Seshia, "Compositional verification without compositional specification for learning-based systems," *University of California at Berkeley*, pp. 1–8, 2017.

[65] S. Bak, T. T. Johnson, M. Caccamo, and L. Sha, "Real-Time Reachability for Verified Simplex Design," in *2014 IEEE Real-Time Systems Symposium*, Dec 2014, pp. 138–148.

[66] D. M. Lopez, P. Musau, H.-D. Tran, S. Dutta, T. J. Carpenter, R. Ivanov, and T. T. Johnson, "Arch-comp19 category report: Artificial intelligence and neural network control systems (ainncs) for continuous and hybrid systems plants," in *ARCH19. 6th International Workshop on Applied Verification of Continuous and Hybrid Systems*, ser. EPiC Series in Computing, G. Frehse and M. Althoff, Eds., vol. 61. EasyChair, April 2019, pp. 103–119.

[67] S. Bak, S. Bogomolov, and T. T. Johnson, "Hyst: A source transformation and translation tool for hybrid automaton models," in *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, ser. HSCC 15. New York, NY, USA: Association for Computing Machinery, 2015, p. 128133.

[68] C. Barrett, A. Stump, and C. Tinelli, "The smt-lib standard - version 2.0," in *Proceedings of the 8th international workshop on satisfiability modulo theories, Edinburgh, Scotland,(SMT '10)*, 2010.