Comprehensive Survey on Formal Verification of Neural Networks: Foundations, Methods, and Future Directions

SurveyForge

Abstract— Formal verification of neural networks serves as a vital tool for ensuring the reliability and safety of AI systems across various domains. This comprehensive survey explores the intersection of formal methods and neural network models, detailing methodological advancements and application-specific challenges. The survey highlights the pivotal role of formal verification in providing mathematical guarantees for safety-critical properties such as robustness, fairness, and interpretability. It examines key techniques including satisfiability modulo theories, numerical optimization, and abstraction methods, emphasizing their contributions and limitations in scaling to complex architectures like transformers and graph neural networks. By integrating verification within learning pipelines, the survey presents novel frameworks designed to marry the synthesis of task performance with verifiability. Additionally, it outlines emerging trends in verification-driven neural architecture design and community-standard benchmarks, which are essential for progressing toward scalable, interpretable, and ethically aligned AI systems. Future research directions underscore the need for enhanced computational strategies, collaborative efforts, and standardized evaluation frameworks to overcome current scalability barriers and to expand the applicability of formal verification in real-world settings.

Index Terms—Formal verification techniques, Neural network scalability, Robustness certification.

1 Introduction

Formal verification of neural networks (NNs) represents a critical intersection of artificial intelligence (AI) and formal methods, aimed at ensuring the reliability, robustness, and safety of deep learning systems through rigorous, mathematical guarantees. This survey begins by grounding readers in the field's motivations, challenges, and historical lineage, providing a roadmap to navigate the evolving landscape of formal neural network verification.

Recent advancements in AI, particularly the application of deep neural networks, have led to transformative breakthroughs across domains as diverse as autonomous driving, healthcare, and finance. However, these successes also underscore a growing reliance on systems whose internal workings are opaque and whose behaviors are inherently uncertain. Formal verification aims to address these shortcomings by providing assurances about defined properties—such as safety, robustness, and fairness—of neural network models. Unlike statistical testing or empirical evaluation, formal verification furnishes guarantees by exhaustively exploring all potential inputs or states under specified conditions, leaving no room for adversarial manipulation or unforeseen failures. These guarantees are particularly indispensable for safety-critical tasks, where failure could lead to catastrophic consequences, as exemplified by incidents in autonomous driving systems [1], [2].

Formal verification in software and hardware development has traditionally leveraged techniques such as satisfiability solving (SAT/SMT), abstract interpretation, or symbolic reasoning. Transferring these mature approaches to neural networks introduces unique and unprecedented challenges. Neural networks are characterized by their immense scale, non-linear activation functions, and high-dimensional input spaces. Unlike traditional systems, modes of failure in NNs might involve adversarially crafted perturbations, sensitivity to noisy inputs, or biases embedded within training data. For instance, verifying robustness to adversarial attacks, such as ensuring the stability of a classifier under bounded L-norm perturbations, is computationally hard and often formulated as an NP-complete problem [3], [4]. Despite diverse theoretical advancements, no singular verification approach achieves a pragmatic balance between scalability, generalizability, and precision.

Central to the landscape of formal NN verification is its application in domains demanding unassailable trust. For autonomous vehicles, formal guarantees about state-space reachability and collision avoidance are pivotal [2]. Similarly, robustness against adversarial examples has implications for ensuring dependability in biometric systems, banking fraud detection, and medical image diagnostics [5], [6]. Despite such progress, practical limitations persist. For example, scalability barriers restrict the feasibility of verifying large-scale architectures, such as transformers or graph neural networks (GNNs), which are becoming increasingly prevalent [7], [8].

The evolution of formal verification techniques for NNs has seen significant milestones. Early methods principally aimed at feed-forward networks with rectified linear unit (ReLU) activations, leveraging their piecewise linearity to encode verification problems into optimization or SAT frameworks [9]. Extensions to tackle broader architectures and activation functions, such as convolutional, recurrent, or even Bayesian neural networks, have since emerged. Interval bound propagation, mixed-integer linear programming (MIP), and semidefinite programming (SDP) have

demonstrated varying degrees of success in handling the inherent complexity of NNs [10], [11], [12]. Furthermore, hybrid strategies, which integrate symbolic reasoning with numerical optimization, have opened pathways to expedite verification tasks without compromising precision [13].

From a historical lens, formal verification represents not merely a transplantation of software verification tools but a fundamental rethinking of how soundness and precision are conceptualized for data-driven learning systems. The benchmarks created by initiatives such as VNN-COMP [14] trace an accelerating trajectory of collaboration within this research community, benchmarking state-of-the-art tools on real-world tasks while fostering standardization efforts like the adoption of ONNX and VNN-LIB formats. Emerging synergies between verification tools (e.g., Marabou, Reluplex) and domain-specific requirements herald a fertile avenue for impactful research and industrial deployment [15].

Looking ahead, the dual objectives of scalability and usability represent key challenges for the field. Research must pivot toward methods that generalize across architectures, incorporate stochastic behaviors, and integrate seamlessly into the neural network development lifecycle. Subfields such as verification-driven architecture design and training-guided verification promise to dissolve conventional barriers between learning and proving [16], [17]. By weaving together insights from formal methods, optimization, and AI, verification research stands poised to reshape societal reliance on machine learning technologies, driving safer and more trustworthy neural network systems.

2 Fundamental Concepts and Formalization

2.1 Core Verification Properties of Neural Networks

Formal verification of neural networks hinges critically on specifying and evaluating key properties that capture the behavior, safety, and fairness of these systems in diverse application contexts. This subsection explores the core verification properties commonly analyzed and guarantees sought in neural networks, emphasizing their application motivations, formal definitions, and challenges.

Safety properties are foundational in ensuring neural networks do not lead to catastrophic failures in safety-critical environments. These properties are typically formalized as constraints on network outputs under a defined set of inputs. For example, in autonomous vehicle systems, safety might require that an obstacle-detecting neural network must never predict a false "no obstacle" within a specific region of input space [2], [6]. Mathematically, for a given property ϕ, the network $f(x)$ satisfies safety if $\forall x \in \mathcal{X}_{safe}\, \phi(f(x))$ holds. Challenges often arise in proving such guarantees because high-dimensional input spaces and the nonlinearity of modern architectures make exhaustive analysis computationally intractable.

Closely related to safety is **robustness against adversarial attacks**, an area of intense focus given neural networks' susceptibility to small perturbations that can drastically alter outputs. Robustness is quantitatively assessed as the maximum perturbation ϵ (measured under norms such as L_1, L_2, or L_∞) that does not cause a misclassification or property violation: $\forall \delta, ||\delta|| \leq \epsilon, f(x+\delta) = f(x)$. Various works have tackled robustness computation, including methods utilizing convex relaxations, optimization, and abstract interpretation [3], [4]. While exact approaches, such as Mixed Integer Programming (MIP), have been shown to compute tight bounds [18], they face scalability issues for large networks. Conversely, relaxed methods are more efficient but inherently approximate, leading to trade-offs between precision and computational cost [4].

Fairness guarantees represent another critical property, especially as neural networks are increasingly deployed in societal decision-making systems. Fairness verification ensures that sensitive attributes (e.g., race, gender) do not introduce discrimination in predictions. Formally, this involves certifying that network output differences for comparable individuals lie within acceptable bounds: $\forall x_1, x_2, |f(x_1) - f(x_2)| \leq \tau$ for x_1 and x_2 differing only in sensitive attributes. Emerging frameworks, such as neuro-symbolic methods, allow for incorporating fairness-specifying constraints into verification pipelines [19]. However, ensuring fairness across intersectional attributes and high-dimensional data remains an unsolved challenge.

Another essential aspect is **correctness and consistency**, which guarantees the neural network's outputs align with domain-specific rules or ground truths. For example, neural networks controlling aircraft must adhere to predefined collision-avoidance protocols to ensure safety mandates [20]. Correctness verification often involves mapping neural network behavior to logical and mathematical frameworks, such as Satisfiability Modulo Theory (SMT) solvers, but suffers from scalability bottlenecks as network complexity grows.

Lastly, **monotonicity and interpretability properties** are vital for deploying neural networks in transparent, controlled environments. Monotonicity mandates outputs vary predictably with input changes (e.g., increasing financial risk factor should increase a credit score), and its violation can be formally ruled out using abstraction techniques [21]. Meanwhile, interpretability is crucial for understanding why outputs are generated—a concern addressed by combining verification techniques with eXplainable Artificial Intelligence (XAI) to generate counterexample regions or symbolic explanations of failures [22], [23].

Despite significant progress, verifying these properties at scale remains challenging. High-dimensional spaces, black-box network models, and nonlinear behaviors necessitate striking careful trade-offs between precision, scalability, and soundness in verification frameworks. Future research avenues include integrating verification into neural network training processes to embed verifiable properties inherently and leveraging community-standard benchmarks, such as VNN-LIB, to catalyze development [24].

In conclusion, formal verification properties such as safety, robustness, fairness, correctness, monotonicity, and interpretability are indispensable for ensuring neural network reliability. Bridging current gaps in scalability and precision will define the future trajectory of this rapidly evolving field, enabling the trustworthy deployment of neural networks across critical applications.

2.2 Formal Specification Languages and Frameworks

The formal specification of neural network properties serves as the cornerstone of rigorous verification processes, offering precise descriptions of desired network behaviors under all possible conditions. By laying out these behaviors in mathematical and logical terms, specification frameworks enable verification methods to rigorously assess whether systems meet given requirements, particularly in safety-critical contexts. This subsection explores the foundational methodologies, strengths, limitations, and emerging trends in formal specification frameworks, bridging the theoretical underpinnings with their practical applications.

At its core, a formal specification defines the properties of a neural network through constraints, relationships, and behaviors involving its inputs and outputs. For example, robustness against adversarial perturbations is often articulated as invariance to small, bounded input changes, formalized as $||x' - x|| \leq \varepsilon \implies f(x') = f(x)$, where f is the neural network function, x denotes the input, and ε specifies the perturbation bound [25]. Similarly, safety properties are formally specified to ensure bounded output conditions critical to application-specific requirements, such as guaranteeing that aircraft collision avoidance networks adhere to predefined safety margins [26].

Mathematical formulations lie at the heart of these frameworks, typically representing input-output relationships through linear inequalities or logical formulas. A notable approach involves piecewise-linear approximations for ReLU-based neural networks, which partition the input space into regions where the network behaves linearly. Techniques such as Mixed Integer Programming (MIP) effectively encode these constraints to verify properties with high precision [13], [27]. However, these methods face scalability challenges owing to the exponential growth of constraints as the network's size and complexity increase.

For systems with temporal dependencies, frameworks like linear temporal logic (LTL) or computation tree logic (CTL) are used to express properties over time, such as "event A must always occur before event B." These temporal logic specifications are especially useful in domains such as autonomous vehicles and robotics, where the timing and order of decisions are critical [28]. While temporal logic enables expressive descriptions of sequential behaviors, verifying these properties often requires computationally expensive model checking methods, posing scalability hurdles in large and intricate networks.

Emerging tools and standardized formats, such as VNN-LIB, have streamlined the specification process by providing a unified platform for defining verification problems and enabling interoperability across verification frameworks. VNN-LIB, for instance, enhances the usability of tools like Marabou and NNV, facilitating benchmarking and promoting research reproducibility [14]. Specification frameworks focusing on fairness, such as CertiFair and Fairify, extend the scope to ethical dimensions by formalizing constraints that systematically evaluate and mitigate biases in neural networks [29], [30].

Graphical and geometric approaches further enrich the specification ecosystem, particularly for high-dimensional input domains. Techniques such as zonotopes, star sets, and polytopes encapsulate input-output constraints geometrically, enabling efficient reasoning about network behaviors in high-dimensional spaces. ImageStars, for instance, have proven effective for scalability in analyzing convolutional neural networks (CNNs) applied to image verification tasks [31]. However, while these approaches can improve scalability and precision by minimizing over-approximation, their computational resource demands remain significant, especially for deep networks.

Despite these advancements, substantial challenges persist in specification frameworks, with scalability emerging as a critical issue. Converting high-level logical or temporal properties into computationally tractable constraints frequently involves trade-offs. Approaches like semidefinite programming (SDP) provide mathematically relaxed formulations to address non-linear neural networks, yet their effectiveness diminishes for highly complex architectures due to gaps in precision [32]. Similarly, convex relaxations yield approximate behaviors but inherently sacrifice completeness to achieve scalability [4].

In response, recent efforts integrate specification languages more tightly with machine learning processes. Neuro-symbolic frameworks embed symbolic property representations directly into neural models, bridging real-world requirements with verifiability [19]. Likewise, tools like CertiFair align fairness constraints with property-driven network design, enforcing ethical guarantees during training [30]. These approaches highlight the potential for co-development, not only using formal specifications to verify models post hoc but also designing neural architectures to inherently satisfy these specifications.

In summary, the formal specification of neural network properties forms a crucial bridge between theoretical rigor and practical constraints in the verification of complex systems. As neural networks expand into increasingly critical domains, the demand for scalable, expressive, and interoperable specification frameworks will only grow. Addressing these challenges through continued innovation and integration with learning processes will be pivotal in advancing verification paradigms and enabling their application to next-generation neural systems.

2.3 Complexity and Scalability Challenges in Neural Network Verification

The formal verification of neural networks faces significant challenges in complexity and scalability, stemming primarily from the structured, high-dimensional representations of these models, their nonlinear activations, and the combinatorial explosion of possible behaviors. This subsection explores these barriers in depth, discussing computational limits, current methodologies, and emerging strategies to mitigate scalability concerns.

At the core of the challenge lies the inherent nonlinearity and dimensionality of neural networks. Deep neural networks (DNNs), especially those with piecewise linear activation functions like ReLU, exhibit behavior that is non-convex and highly dependent on the partitioning of the input space into distinct linear regions. The verification task often requires analyzing whether all possible configurations satisfy specified properties, reducing to decision problems

that are frequently NP-complete—even for relatively simple networks with a shallow structure or limited specifications [3], [9]. For instance, verifying robustness properties under adversarial perturbations necessitates reasoning over an exponential number of piecewise linear regions formed by the ReLU activation. These regions scale exponentially with the number of neurons and layers, making explicit enumeration infeasible in all but the smallest networks.

Another critical factor contributing to scalability challenges is the interaction between nonlinear activation functions and high-dimensional input spaces. For feedforward neural networks, the combinatorial growth of activation phase configurations (e.g., determining whether each ReLU unit is active or inactive) significantly complicates verification tasks. Techniques leveraging linear programming (LP) or mixed-integer linear programming (MIP) partially address this issue by encoding these constraints into tractable formulations; however, such methods are effective only up to moderate-sized networks before the computational burden becomes prohibitive [33], [34].

Additionally, networks containing nonlinear activations such as sigmoid or tanh models exacerbate complexity, as their smooth, non-piecewise structure eludes the direct application of SMT (Satisfiability Modulo Theories) solvers or linear approximations. Approaches combining convex relaxations with semidefinite programming (SDP) attempt to combat this by creating tighter over-approximations of decision boundaries. Nonetheless, these techniques often trade precision for tractability, potentially leading to overly conservative verification results [35], [36].

High-dimensional input spaces further complicate verification due to the need for global guarantees over continuous domains, which are often infinite. Sampling-based or statistical techniques, such as those utilizing interval arithmetic or symbolic intervals, offer some promise by systematically partitioning the input space into manageable subregions and propagating bounds across layers [37]. However, these methods tend to suffer from over-approximation errors, which accumulate with depth and render results increasingly imprecise. This trade-off underscores the limitations of purely over-approximate approaches, necessitating hybrid techniques that balance precision and scalability.

Emerging strategies have sought to mitigate these issues through modular and abstraction-based frameworks. Abstract interpretation has gained prominence as a scalable method for neural network verification, where networks are simplified into "abstract domains" (e.g., intervals, zonotopes, or star sets) while preserving soundness. Such abstractions enable tighter error bounds while handling larger networks but typically require iterative refinement to eliminate spurious counterexamples [38]. Similarly, symbolic propagation methods have demonstrated potential by propagating symbolic constraints through layers rather than explicit numerical computations, allowing analysis of structurally larger networks [39].

Hybrid approaches integrating symbolic and numerical techniques represent a promising avenue for tackling scalability bottlenecks. Efforts such as combining LP-based reachability analysis with geometric constructs like zonotopes or polyhedral approximations offer tighter guarantees while maintaining computational efficiency [40], [41]. Additionally, leveraging parallelism on hardware accelerators (e.g., GPUs) and designing solvers that exploit problem-specific structure have shown empirical success in recent tool competitions like VNN-COMP [14].

The interplay between neural architectures and verification techniques continues to pose critical challenges. Architectures like transformers, graph neural networks (GNNs), and recurrent networks introduce unique scalability and expressiveness issues due to attention mechanisms, node dependencies, or long-term temporal dependencies, respectively [42], [43].

Looking forward, there is a strong impetus to design verifiability-aware neural architectures, where properties conducive to formal reasoning—such as sparsity or modular structure—are embedded at the design stage. Additionally, advancing hybrid symbolic-numeric techniques, adopting adaptive refinement strategies, and introducing domain-specific formulations for scalability represent promising research avenues. Continued progress requires a blend of theoretical insight, practical tool development, and interdisciplinary collaboration to address the daunting computational burdens in verifying complex, deeply layered neural networks.

2.4 Abstraction and Modeling Techniques for Formal Verification

Abstraction and modeling techniques are pivotal for enabling the formal verification of neural networks, addressing the computational challenges posed by their complexity while ensuring soundness. These methods approximate a network's behavior through simplified representations or modular breakdowns that preserve critical characteristics necessary for verification. This subsection delves into the core approaches, their principles, and the trade-offs they navigate between precision, scalability, and guaranteeing properties.

Abstract Interpretation stands out as a foundational technique in formal verification. By mapping input-output relationships of neural networks to clearly defined domains like intervals, zonotopes, or polyhedra, abstract interpretation provides computationally efficient approximations that are sound with respect to specified properties. For example, tools such as ERAN extend formal safety analysis capabilities by integrating abstract interpretation with bounding techniques to scale verification to larger networks while upholding robustness guarantees [2], [44]. However, over-approximation in such methods often leads to false positives, encompassing infeasible behaviors. To mitigate this, refinement techniques like counterexample-guided abstraction refinement (CEGAR) iteratively reduce over-approximation by incorporating counterexamples to enhance precision [45].

Symbolic Propagation represents another powerful approach, advancing constraints through a network layer by layer to reason about input-output relationships symbolically. This technique is particularly effective for networks with linear components and piecewise-linear activations, such as ReLUs. Methods like CROWN utilize efficient linear relaxations to support scalable robust certification across broader properties [46]. Nevertheless, symbolic propaga-

tion faces challenges in handling highly non-linear functions, like Sigmoid or GeLU, where relaxation gaps can reduce precision. Recent advances in tightening relaxation bounds for individual neurons highlight ongoing efforts to overcome these limitations [47].

Reachability Analysis, another abstraction-driven methodology, focuses on rigorously computing all possible network outputs for bounded input spaces. This makes it particularly suitable for analyzing safety-critical properties. Tools employing interval bound propagation (IBP) or geometric constructs like zonotopes have demonstrated scalability when applied to large convolutional and residual network architectures [44], [48]. Despite the computational demands of determining tighter bounds, parallelization strategies and residual-based methods have improved practical efficiency. Additionally, frameworks like INVPROP push the boundaries by addressing preimage verification, which determines input sets corresponding to specific outputs—a critical problem for backward reachability [49].

Decomposition Strategies further tackle scalability by dividing the verification process into smaller, more manageable subproblems. Layer-wise decomposition analyzes each layer independently before aggregating results to provide network-wide guarantees [13]. Similarly, input segmentation divides input domains into subregions, isolating complex regions to improve overall performance [50].

The adoption of **domain-specific modular abstractions** demonstrates the flexibility of modeling techniques. For example, Cnn-Abs abstracts convolutional layers to enable scalable verification, while EEV capitalizes on the binary nature of binarized neural networks (BNNs) to achieve substantial computational speed-ups while maintaining guarantees comparable to traditional networks [51], [52]. Such tailored abstractions reveal how domain-specific optimizations can strike a balance between computational efficiency and precise guarantees.

Despite significant progress, abstraction techniques confront persistent challenges, including over-approximation, structural rigidity, and handling non-determinism inherent in stochastic networks. Convex relaxation, while highly promising, still faces hurdles in achieving the tightness required for non-linear activation functions, underscoring the need for further innovation [47]. Recent efforts leveraging parallel processing on hardware accelerators, such as GPUs, also indicate promising directions for scaling verifications on modern large-scale architectures [44].

In conclusion, abstraction and modeling techniques underpin the scalability of modern neural network verification by tackling the computational intractability of large systems while retaining soundness. However, the trade-offs between precision, scalability, and over-approximation require continued refinement. Future innovations may focus on integrating abstraction frameworks more tightly with data-driven modeling and exploiting advanced hardware paradigms to deliver broader applicability, improved efficiency, and enhanced practical performance.

2.5 Uniqueness and Challenges of Verifying Neural Networks

Formal verification of neural networks presents a distinct set of challenges that stem from the inherent differences between neural networks and traditional software systems. While traditional programs are composed of explicit, rule-based logic, neural networks rely on implicit representations learned empirically. This dichotomy introduces fundamental difficulties in formalizing and analyzing their behavior, with unique implications for verification methodologies and scalability.

One significant challenge arises from the *black-box nature* of neural networks. Unlike traditional systems with transparent and meticulously organized logic, neural networks operate through complex, non-linear transformations distributed across numerous parameters. These parameters, often numbering in the millions, offer no intuitive explanation of the decision-making process, creating barriers to understanding and formal validation. Techniques such as symbolic abstraction and interval propagation attempt to mitigate this opacity, but often sacrifice scalability or precision [13], [53]. Even advanced attempts at symbolic propagation for higher precision often require computational compromises due to the increasing difficulty in reasoning about non-linear interactions at scale [54].

The high-dimensional and *unbounded nature of input spaces* further complicates verification. Neural networks are typically designed to process high-dimensional data, such as images or graphs, within continuous or semi-continuous domains. For instance, verifying robustness under adversarial perturbations involves exhaustively assessing all permissible perturbations within a bounded norm, an inherently intractable task as input dimensions increase [5]. To address these challenges, techniques such as zonotope-based reachability analysis and convex relaxation have been proposed, offering efficient yet approximate solutions. However, these methods often suffer from over-approximation errors, with trade-offs between tractability and the tightness of guarantees [33], [41].

Another critical issue is the *stochasticity introduced by neural network training*. Neural networks are typically trained using non-deterministic techniques such as stochastic gradient descent, making their parameters sensitive to factors like initialization seeds and training data subsets. This introduces variability in behavior across different network instantiations, complicating efforts to verify properties such as robustness and fairness. Emerging studies have proposed approaches like probabilistic verification and Bayesian abstraction to reason about this stochasticity [55], [56]. However, these methods often rely on empirical guarantees rather than exact symbolic reasoning, limiting their application in safety-critical domains.

The *combinatorial explosion in non-linear activation patterns* presents yet another challenge. Activation functions such as ReLU and sigmoid divide the input space into piecewise-linear or non-linear regions, leading to millions of possible activation patterns in deep networks. Verifying properties like robustness or reachability often requires a detailed exploration of these patterns, where even shallow networks are computationally equivalent to solving NP-hard problems [3], [12]. To mitigate this, abstractions such as interval neural networks and mixed-integer linear programming (MIP) formulations have been applied. Although effective in certain scenarios, they often fail to scale to modern architectures with many layers and parameters [13], [57].

Another uniqueness lies in *the hybrid nature of neural network applications*. In real-world systems, neural networks are often components of larger hybrid systems, such as cyber-physical or autonomous systems. Here, the challenge is not only to verify the neural network's input-output properties but also to ensure its safe integration with non-linear dynamical models of the overall system. Verification frameworks such as Verisig and DeepNNC have extended their scope to closed-loop systems, combining neural verification techniques with reachability analysis for plant dynamics [28], [58]. However, such frameworks struggle with non-linearities and time-evolving errors caused by inaccuracies in the system abstraction process.

The way forward requires increasing standardization, tool interoperability, and community benchmarks. VNN-COMP benchmarks have already facilitated comparison across methodologies, but a unification of specification languages and model formats would significantly accelerate progress [59]. Additionally, emerging trends such as integrating verification within training pipelines (e.g., differentiable verification objectives) and leveraging advanced computational resources like GPUs and TPUs for scalability offer promising directions [58], [60].

In summary, neural network verification is uniquely positioned at the confluence of machine learning and formal reasoning. While significant progress has been made in addressing specific challenges, the field remains constrained by issues of scalability and the intrinsic complexity of neural networks. Future advancements would likely come from interdisciplinary efforts that combine innovations in formal methods, optimization techniques, and machine learning to create systems that are both scalable and robust.

3 Techniques and Methods for Formal Verification

3.1 Symbolic Methods and SMT-based Approaches

Symbolic methods and SMT-based approaches to formal verification are foundational techniques that reframe the verification of neural networks into logical reasoning or satisfiability problems. By encoding neural network constraints into formal representations that can be evaluated by Satisfiability Modulo Theories (SMT) solvers, these approaches provide rigorous and generalizable verification frameworks. In particular, SMT-based methods have demonstrated increasing success in verifying properties of piecewise-linear neural networks (e.g., networks with activation functions like ReLU), due to their well-structured, yet mathematically complex, nature.

At the core of SMT-based approaches lies the task of formal encoding. Neural networks are expressed as systems of logical and arithmetic constraints, capturing key elements such as network weights, activation functions, and defining properties (e.g., robustness to adversarial perturbations). For example, piecewise-linear ReLU activation functions are commonly encoded as disjunctions of linear constraints, representing the distinct activation regions for each neuron [3], [9]. This symbolic formulation makes it possible to model verification tasks as query problems: "Does there exist an input that violates a desired output property?" Such queries can then be answered using SMT solvers capable of reasoning about combinations of Boolean logic and linear/numerical constraints.

A significant strength of SMT-based techniques is their ability to provide exact solutions, guaranteeing correctness and safety in settings where approximate methods fall short. Techniques such as Branch-and-Bound (BaB) have proven crucial in scaling SMT problems to larger networks, incrementally exploring activation space partitions while employing heuristics to prune infeasible paths [13]. Optimization-focused SMT extensions, such as those involving conflict clause inference and unit propagation techniques, further improve solver efficiency by deducing tighter bounds on neural activations [3]. However, this exactness comes at a computational cost. Solving SMT queries incurs exponential time complexity in the worst case, an obstacle to verifying networks with even moderately high dimensionality or complex architectures. Despite these challenges, hybrid extensions, which combine symbolic SMT reasoning with faster numerical relaxations, have achieved notable successes in increasing scalability [9], [57].

An emerging area of research involves adapting SMT solvers to the verification of neural network architectures with non-linear or more expressive activation functions, such as sigmoidal or tanh layers. These functions have traditionally presented difficulties due to their non-linear, non-convex nature. Recent efforts in hybrid symbolic approaches integrate piecewise approximations or secondary numerical layers to handle such architectures more effectively [4], [61].

The practical utility of SMT-based approaches extends across diverse applications. For instance, tools such as Marabou demonstrate the effectiveness of SMT in verifying safety properties in autonomous systems like aircraft collision avoidance [15]. Similarly, SMT-based frameworks have shown strong results in validating robustness against adversarial attacks, a key goal in ensuring reliable neural network deployment in high-assurance systems [2]. Despite their computational bottlenecks, SMT methods' ability to produce counterexamples in cases where specifications are violated enhances their utility, offering actionable insights to refine neural network models.

Looking ahead, a pressing challenge involves bridging the scalability gap intrinsic to SMT-based verification. Parallelization strategies and approximation frameworks are promising directions, enabling solvers to exploit modern computational resources like GPUs or distributed systems [62]. Additionally, the integration of SMT with abstract interpretation and reachability methods holds promise for efficiently verifying properties across broader architectures while maintaining soundness [9], [48].

As neural networks evolve in architecture complexity, future research must extend the applicability of SMT-based verification to include dynamic and recurrent systems. Techniques involving invariant generation and iterative re-encoding of temporal dependencies offer potential in addressing these challenges [63], [64]. Finally, there is a growing interest in integrating symbolic verification into real-time machine learning workflows, enabling verification-informed training for inherently safer and verifiable models [8].

In summary, SMT-based formal verification provides rigorous guarantees for safety-critical and robustness prop-

erties of neural networks. While significant scalability hurdles remain, ongoing advancements in hybrid methodologies and computational acceleration techniques are steadily expanding the feasibility of applying symbolic reasoning frameworks to real-world, large-scale neural network models.

3.2 Numerical Optimization and Convex Relaxation Techniques

Numerical optimization and convex relaxation techniques have become integral in the formal verification of neural networks, bridging the gap between computational efficiency and the rigorous handling of non-convexity inherent to these models. By framing verification tasks such as robustness analysis and safety property certification as optimization problems, these approaches leverage mathematical formulations to deliver scalable and practical solutions. This subsection examines the methodologies, trade-offs, and emerging directions within this domain, linking their computational insights to broader verification strategies.

A central challenge in numerical optimization lies in managing activation functions like Rectified Linear Units (ReLUs), which introduce branching behavior due to their piecewise linear nature. Mixed-Integer Programming (MIP) has been particularly effective in precisely encoding this behavior, formulating the verification problem as a set of linear constraints augmented by integer variables to capture activation states [27]. MIP-based approaches ensure exactness and provide rigorous certification of robustness properties. However, the scalability of MIP is constrained by the exponential increase in problem size for networks with many ReLUs. To mitigate this, efforts such as enhanced branching heuristics and pre-solve techniques have expanded the capacity of MIP verifiers to handle larger networks [13].

In contrast, convex relaxation techniques prioritize computational efficiency by simplifying non-convex constraints into convex approximations. Linear and semidefinite relaxations are notable examples, replacing non-convex activation behaviors with computationally tractable convex bounds. Techniques that employ linear programming (LP) relaxations, for example, model ReLU activation functions within tight linear enclosures, facilitating the rapid verification of adversarial robustness [3]. However, while linear relaxations provide efficient solutions, their inherent approximation gaps limit their utility, particularly in high-dimensional networks. Semidefinite programming (SDP) methods extend this framework by capturing richer geometric properties of activation functions, such as quadratic relationships or bounded slopes. By incorporating these additional constraints, SDP-based verification improves precision, proving particularly effective in safety-critical scenarios like flight collision avoidance systems [32]. Nonetheless, the computational cost of SDP increases significantly as network size grows.

A complementary direction involves dual optimization methods, which use Lagrangian relaxations to iteratively derive robustness certificates. These approaches refine bounds on potential property violations through dual objectives, allowing intermediate solutions to offer meaningful guarantees before full convergence [12]. Owing to their scalability and theoretical soundness, dual methods bridge the precision gap left by simpler convex relaxations, aligning well with large-scale verification problems.

More recent innovations incorporate these methods into the training processes themselves, fostering both robust models and efficient verification. For instance, Interval Bound Propagation (IBP) integrates convex bounds during neural network training, aligning models with relaxed constraints to simplify subsequent verification tasks [65]. Similarly, training strategies designed to emphasize ReLU stability and weight sparsity reduce post-training complexity, enabling more efficient property certification [66]. These co-design approaches illustrate the growing convergence between training methodology and formal verification, targeting models that are inherently more verifiable.

However, these methods are not without challenges. Convex relaxations, particularly linear approximations, often underestimate worst-case behaviors, leading to false robustness claims [4]. Furthermore, scalability difficulties persist for deep architectures or networks with non-standard structures like attention mechanisms or graph components. Strategies leveraging parallelization, such as specialized GPU implementations, have yielded promising speed-ups in tools capable of millisecond-scale verification [44], though balancing precision with speed remains an ongoing balancing act.

Looking to the future, hybrid methodologies integrating convex relaxations with symbolic reasoning or sampling-based approximations could bolster both the scalability and precision of verification frameworks. Recent work on scalable convex hull approximations, such as multi-neuron abstractions, exemplifies promising advancements in computational geometry [67]. Meanwhile, adapting convex relaxation frameworks to novel neural architectures like transformers or spiking neural networks could further expand their applicability, enabling the formal verification of increasingly intricate systems [68].

In conclusion, numerical optimization and convex relaxation techniques contribute a vital computational backbone to the field of formal neural network verification, balancing efficiency and rigor. As these approaches evolve, their integration into hybrid methodologies and compatibility with emerging architectures promise to address outstanding challenges, paving the way for the certification of ever more complex and safety-critical systems.

3.3 Abstraction and Refinement Techniques

Abstraction and refinement methods have become foundational in the formal verification of neural networks, aiming to address the scalability challenges imposed by their high dimensionality and non-linear properties. At their core, abstraction techniques create simplified representations of a neural network, or abstract models, that preserve key properties relevant to verification while reducing computational complexity. Refinement methods, in turn, iteratively enhance these models to address inaccuracies introduced by over-approximation, enabling a trade-off between scalability and precision. Together, these methods streamline the verification process, especially for large-scale and deeply layered networks.

Abstract interpretation is one of the most prominent abstraction-based approaches. It systematically constructs over-approximations of neural networks within certain abstract domains, such as intervals, zonotopes, or polyhedra, to bound their output spaces under constrained inputs. For instance, interval arithmetic propagates input intervals across layers, maintaining soundness but resulting in significant over-approximation due to the inability to capture dependencies across dimensions [37]. More expressive domains, such as zonotopes and star sets, allow finer-grained geometric representations of the neural network's behavior, improving precision at the cost of increased computational overhead [38]. Abstraction has also been extended to accommodate piecewise linear networks, such as those with ReLU activations, using polyhedral techniques to track reachable output sets [33]. These methods are foundational for verifying safety-critical systems but often struggle with scaling to larger networks or networks with complex activation functions.

Counterexample-guided abstraction refinement (CEGAR) is a pivotal refinement technique that addresses the limitations of coarse abstractions. In CEGAR, an abstract model that fails to validate a property yields spurious counterexamples, which are analyzed to refine the abstraction iteratively. This process incrementally improves precision by narrowing the over-approximations in areas of interest. The iterative nature of CEGAR achieves a balance between coarse initial abstractions and computational efficiency in refinement [38]. Though computationally demanding in some cases, the success of CEGAR in hardware and software verification has inspired its adaptation to neural networks, particularly in hybrid systems [60].

A specific challenge in abstraction and refinement arises from the heterogeneous nature of neural networks. Methods tailored for simple feed-forward architectures may fail to generalize to more complex configurations such as recurrent or convolutional networks. For example, domain-specific abstractions for binarized or quantized neural networks, which emphasize representation sparsity, have shown promise [69]. Additionally, novel abstraction techniques have emerged for networks integrated into control systems, leveraging techniques like constrained zonotopes to capture the interplay between dynamics of the plant and neural controllers [70].

Emerging research focuses on refining abstraction methods to enhance their integration with symbolic and numerical verification frameworks. For example, hybrid techniques that combine symbolic interval analysis with numerical optimization leverage the strengths of both paradigms to obtain tighter bounds on neural network behavior without excessive computational overhead [37]. Similarly, leveraging neural networks themselves as abstractions of complex dynamics, with formally certified error bounds, presents a novel direction for verification in high-dimensional systems [53]. This counterexample-guided synthesis of neural abstraction models considerably improves verification accuracy while scaling to broader problem domains.

Despite their advancements, abstraction and refinement methods face key challenges, including the exponential growth of verification complexity with deeper abstractions and the difficulty of generalizing abstractions to novel or domain-specific network architectures. Additionally, abstractions must balance precision and scalability to be viable in real-world applications, with excessive refinement cycles often leading to computational infeasibility. Addressing these challenges calls for interdisciplinary insights, such as the incorporation of advanced machine learning techniques into the abstraction process or leveraging emerging computational paradigms, such as distributed systems or quantum computing [55].

In conclusion, abstraction and refinement techniques represent indispensable tools in the formal verification toolbox for neural networks. Their ability to distill complex neural behaviors into manageable representations is critical for advancing verification in high-assurance applications. Continued innovation in abstraction techniques, coupled with scalable refinement strategies, will define the future trajectory of verifying neural networks in increasingly complex and safety-critical settings.

3.4 Reachability Analysis and Interval-based Methods

Reachability analysis plays a fundamental role in formal verification by estimating the set of all possible states a neural network can transition to under specific constraints on the input space. This method is crucial for ensuring that neural networks uphold safety-critical properties across their operating domains. Typically, reachability involves computing a bounded set of reachable outputs for every allowable input, represented through geometric constructs such as intervals, polytopes, or zonotopes. Among these, interval-based methods have gained prominence for their simplicity and computational efficiency, forming a cornerstone for reachability approximations.

The foundation of interval-based approaches lies in interval arithmetic. Here, inputs are treated as ranges (intervals), and operations across network layers propagate these bounds to yield conservative but sound estimates of the output space. Specifically, the method rigorously bounds the output of each neuron by propagating input ranges through weight matrices and activation functions [48]. A notable advantage of interval arithmetic is its guarantee of soundness: all reachable outputs are enclosed within the computed bounds. However, this technique is inherently limited by its tendency toward over-approximation. In deep and complex networks, compounded over-approximations across layers can lead to excessively loose bounds, limiting practical usefulness.

To address these limitations, more expressive geometric representations, such as zonotopes and polyhedra, have been introduced. These representations encapsulate reachable sets with higher precision by preserving linear transformations compactly during propagation [2], [44]. For instance, zonotopes strike a balance between scalability and precision, making them particularly well-suited for networks with modest depth. However, as the complexity of the geometric representation increases, so does the computational overhead, presenting a trade-off between accuracy and efficiency.

Advancements in adaptive reachability techniques have sought to mitigate these trade-offs. Methods like branch-and-bound (BaB) frameworks iteratively partition the input

domain into subregions, focusing computational effort on regions requiring finer-grained exploration [13], [50]. This dynamic refinement approach is particularly effective in cases with non-linear components, such as ReLU activations, enabling a tighter characterization of the reachable set. However, the branching process can become computationally prohibitive for large-scale networks with high-dimensional input spaces.

In parallel, integration with symbolic methods has further refined interval-based reachability approaches. By coupling interval arithmetic with symbolic propagation, these hybrid techniques significantly improve precision without dramatic increases in computational cost. For example, β-CROWN exploits per-neuron split constraints to compute tighter bounds while maintaining scalability for large networks [46]. Additionally, probabilistic sampling methods have been explored as a complementary strategy, approximating reachable sets through high-dimensional sampling to mitigate over-approximations. While these techniques offer computational efficiency, they inherently trade off completeness, limiting their application in safety-critical systems.

A pressing challenge for reachability analysis lies in its scalability to modern, high-capacity architectures like transformers and graph neural networks. The depth and structural complexity of these models exacerbate over-approximation and computational bottlenecks during propagation [7]. Another key obstacle arises in hybrid systems combining neural networks with traditional dynamical components, which demand an integrated approach to reachability analysis within broader verification pipelines.

Emerging advancements aim to address these challenges through the use of advanced computational resources and novel geometric abstractions. GPU-enabled polyhedral methods, such as GPUPoly, leverage parallel processing to tackle the reachability problem in large networks more efficiently [44]. Similarly, frameworks like PRIMA adopt scalable convex hull approximation techniques inspired by computational geometry to provide more precise abstractions for multi-neuron interactions [67].

In conclusion, reachability analysis, particularly interval-based methods, remains a foundational tool for verifying neural network properties by balancing scalability and precision. Recent progress in hybrid techniques, adaptive frameworks, and hardware-accelerated approaches demonstrates promising paths forward. However, challenges in handling state-of-the-art architectures and high-dimensional inputs persist. Future work should focus on the hybridization of symbolic and interval-based methods, exploring hierarchical abstractions, and developing standardized benchmarks to evaluate advancements across diverse neural network architectures and applications.

3.5 Hybrid and Emerging Scalable Methods

Hybrid approaches aim to address the inherent trade-offs in formal verification of neural networks by integrating symbolic and numerical methods, leveraging the respective strengths of each technique. These methods seek to balance precision, scalability, and computational feasibility, particularly for large-scale and modern neural architectures like transformers and graph neural networks. Recent work demonstrates that hybrid frameworks are essential to overcoming the bottlenecks posed by the combinatorial explosion of verification tasks and the high dimensionality of input spaces.

One prominent category of hybrid methods combines symbolic reasoning with numerical optimization techniques. Symbolic methods, often based on Satisfiability Modulo Theories (SMT) solvers, are highly precise but struggle with scalability for deep networks due to the exponential complexity in handling non-linear activation functions. Numerical approaches, such as Mixed-Integer Programming (MIP) or convex relaxation, excel in scalability by approximating the verification problem but may lose precision. Methods like Branch-and-Bound (BaB), which integrate symbolic and numerical strategies, exploit piecewise-linear structures inherent in neural networks to improve efficiency while maintaining soundness [9], [13]. Specifically, BaB-based techniques partition the verification problem into smaller subproblems using symbolic strategies and solve these subproblems using numerical optimization, enabling practical scalability for ReLU-based architectures.

Hybrid frameworks have also demonstrated efficacy through abstraction-refinement mechanisms. Abstraction techniques coarsen neural network representations to simplify verification tasks, while refinement iteratively enhances precision by reintroducing complexity only where necessary. For instance, tools like Marabou integrate abstraction with SMT solvers and numerical optimization to iteratively refine over-approximations of network behavior while maintaining computational efficiency [15]. Similarly, the DeepAbstract framework constructs abstract networks by clustering neurons with similar activation patterns and propagating constraints through these reduced models. Refinements are introduced incrementally based on counterexamples, minimizing computational overhead [21].

Emerging methods have also explored the use of modular verification for distributed computation. Parallelism is employed to decompose large-scale verification tasks into smaller, independently verifiable components. For example, methodologies such as distributed reachability allow computations to be spread across multiple cores or cloud platforms, significantly reducing computation time while ensuring consistent results. Techniques like symbolic interval analysis combined with parallel processing have achieved strong empirical performance, particularly for networks with piecewise-linear or ReLU activations [37], [48].

New trends also aim at developing hardware-aware verification. Leveraging high-performance computational resources like GPUs, TPUs, and quantum hardware accelerates verification pipelines, further enabling scalability. Frameworks such as POLAR employ symbolic propagation combined with interval arithmetic to ensure verifiable guarantees on modern architectures under limited computational resources [71].

Despite these advancements, significant challenges remain. Hybrid methods often face limitations in handling highly non-linear neural functions such as sigmoid or tanh activations, as these introduce additional complexity in optimization and abstraction. Moreover, ensuring interoperability among various hybrid techniques and scaling them to ultra-large models, such as transformers with billions

of parameters, remains an open problem. Progress in this space will likely require innovative integrations of advanced abstraction methods, domain-specific optimization strategies, and emerging hardware-accelerated computation techniques.

Future research should also explore applying hybrid frameworks to novel neural architectures like Neural Ordinary Differential Equations (Neural ODEs) and spiking neural networks. While preliminary efforts have extended reachability analysis to include Neural ODEs using hybrid optimization and symbolic abstractions [42], scalable solutions for these architectures are still in their infancy. Furthermore, establishing universal benchmarking standards, such as those promoted by the VNN-COMP series [59], will facilitate more consistent evaluation and comparison of hybrid methodologies.

In summary, hybrid methods represent a substantial step forward in bridging the gap between precision and scalability in formal verification of neural networks. By unifying the strengths of symbolic and numerical approaches, these techniques address critical challenges posed by modern architectures. However, continued innovation in algorithm design, hardware integration, and benchmarking is essential to fully unlock the potential of hybrid verification frameworks in addressing real-world, safety-critical use cases.

4 Tools, Frameworks, and Benchmarks

4.1 Overview of Formal Verification Tools

Formal verification tools for neural networks are pivotal in ensuring safety and robustness in high-stakes domains. These tools employ diverse methodologies, ranging from satisfiability-based techniques to abstraction and hybrid approaches, to provide guarantees about properties such as adversarial robustness, functional correctness, and output consistency. This subsection explores state-of-the-art tools, assessing their computational frameworks, strengths, limitations, and practical implications.

Prominent tools leveraging SMT (Satisfiability Modulo Theories) solvers include **Marabou** and its predecessor **Reluplex**, which specialize in verifying ReLU-activated networks [3], [15]. Reluplex extends the Simplex algorithm to accommodate the non-linear constraints introduced by piecewise-linear activation functions like ReLU, enabling scalability on medium-sized networks, as demonstrated in case studies addressing aircraft collision-avoidance systems and image classification benchmarks. Marabou further refines this methodology by introducing efficient bound propagation and preprocessor optimizations, though both tools remain constrained by scalability issues for very large or highly complex networks.

Another class of tools emphasizes abstraction techniques to reduce the complexity of verification tasks while retaining formal correctness. For instance, **ERAN (Efficient Robustness Analysis)** employs abstract interpretation techniques, leveraging domains such as zonotopes and polyhedra to soundly over-approximate reachable sets [2]. This method enables the verification of robustness properties across a broader scope of neural network scales compared to SMT-based approaches. Tools like **DeepAbstract** extend this paradigm by clustering neurons with similar activation behaviors to construct smaller, abstracted networks, accelerating verification while maintaining property soundness [21]. Despite their computational efficiency, abstraction-based methods typically trade precision for scalability, introducing potential false positives.

Hybrid approaches like those in **Branch-and-Bound** (BaB)-based methodologies unify multiple verification techniques to address their individual weaknesses. A notable example is the family of algorithms that integrate BaB with Mixed Integer Linear Programming (MILP) formulations to navigate the branching behavior of ReLU activation functions efficiently [13]. By employing strategic branching heuristics and parallelized implementations, BaB frameworks can consistently outperform both purely symbolic and numerical approaches on convolutional neural networks (CNNs) under adversarial perturbations. However, these methods' reliance on iterative computation and decomposition can limit runtime efficiency on deeper architectures.

The expansion of formal methods to quantized and specialized networks has gained traction with tools like **QNNVerifier** and **QVIP**, which employ integer-linear programming to verify quantized neural networks (QNNs) [72]. These tools address the unique computational constraints of QNNs by reducing model size and enabling energy-efficient deployment, yet still face challenges in scaling to larger models due to the increasing complexity of integer optimizations.

Emerging verification tools also incorporate innovative algorithmic advancements. For example, **Alpha-beta CROWN**, a framework built to leverage GPU-accelerated algorithms, sets state-of-the-art benchmarks in verifying robustness under constrained network dynamics [49]. Additionally, **Charon**, a hybrid framework, combines gradient-based optimization with abstract interpretation, achieving a balance between precise counterexample generation and heuristic-based property proving [73]. These tools illustrate the growing deployment of novel computational resources like distributed systems to bridge the scalability gap between theoretical methodologies and industrial applications.

Nevertheless, challenges persist. Scalability remains a critical bottleneck as neural networks, with millions of parameters or specialized architectures like transformers, exceed existing tool capabilities [4], [24]. Recently introduced initiatives, such as the **VNN-COMP**, emphasize the need for standardized benchmarks (e.g., ACAS Xu, CIFAR10) and network formats (e.g., ONNX, VNN-LIB) to enable fair and reproducible evaluations across different tools [59].

Looking forward, integration with machine learning pipelines, particularly during training, is emerging as a promising research direction. Recent frameworks advocate for verification-aware training methods, embedding constraints directly into loss functions for inherently verifiable network designs [8]. Additionally, advancements in hardware acceleration, such as quantum computing and multi-GPU systems, could mitigate computational costs and render formal verification feasible for real-world applications [74].

In conclusion, the field of formal verification tools for

neural networks has progressed significantly, with advancements in SMT solvers, abstraction techniques, and hybrid methodologies driving innovation. However, the trade-offs between precision, scalability, and efficiency require continued refinement, particularly for large-scale, non-linear, or highly complex architectures. Standardized benchmark challenges, coupled with interdisciplinary efforts, will likely play a vital role in shaping the next generation of robust and scalable verification solutions.

4.2 Comparative Analysis of Tool Capabilities and Use Cases

The field of formal verification for neural networks has produced a diverse array of tools tailored to specific problem domains, methodologies, and scalability challenges. A comparative analysis of these tools highlights nuanced trade-offs between precision, scalability, and application scope, while also uncovering emerging synergies that point toward future methodological advancements.

Broadly, verification tools can be categorized by their methodological foundations, including SMT-based solvers, optimization techniques, abstraction frameworks, and reachability analysis. SMT-based tools like *Reluplex* and its successor *Marabou* are widely recognized for their ability to verify ReLU-activated networks. By leveraging the piecewise linear properties of ReLU, these tools encode neural network verification problems as satisfiability modulo theories (SMT), allowing them to precisely confirm safety-critical properties such as collision avoidance in systems like ACAS Xu [26], [63]. Despite their strengths, SMT solvers often face scalability limitations as network depth and size grow, primarily due to the exponential increase in the constraint space. These limitations become especially prominent when dealing with modern architectures, such as large convolutional models or transformers, which necessitate alternative approaches [44].

Optimization-based methods, particularly those employing mixed-integer programming (MIP), offer complementary strengths by combining precise verification capabilities with systematic optimization. Tools like BaB (Branch and Bound) leverage advanced branching strategies and MIP formulations tailored to ReLU activation functions, achieving strong results in verifying adversarial robustness [13], [27]. These methods have demonstrated their effectiveness even for convolutional networks, a challenging domain for many earlier tools. However, optimization-based methods remain computationally intensive, making them less practical for scenarios where real-time analysis or rapid iteration is required.

Abstraction frameworks, such as *Cnn-Abs* and *ERAN*, prioritize scalability by simplifying neural network representations into over-approximations like zonotopes and polyhedra [21], [51]. By employing abstract interpretation techniques, these tools provide conservative safety bounds, enabling verification of large-scale networks or ensembles. For example, these methods have successfully verified the properties of architectures like VGG on datasets such as ImageNet by leveraging geometric constraints [31]. However, the loss of precision introduced by over-approximation can result in false positives, which limits their applicability for systems requiring tight bounds—for instance, closed-loop control systems in cyber-physical applications [58].

Reachability analysis tools, represented by *NNV* and *POLAR*, use input-output set propagation through layers to evaluate safety and robustness properties, particularly for learning-enabled cyber-physical systems [38], [75]. Techniques like the *ImageStar* representation in *NNV* enable reachability analysis for convolutional networks more efficiently [31], [76]. These tools support both exact and approximate analyses, but exact reachability—encompassing the full range of possible behaviors—often proves computationally prohibitive in high-dimensional spaces, such as images or temporal sequence data.

A crucial consideration in evaluating these tools is their alignment with specific neural network architectures or application requirements. For example, tools like *Verisig* specialize in the verification of hybrid systems that integrate neural controllers with nonlinear plant dynamics, leveraging system transformations to verify closed-loop properties [28]. Meanwhile, tools designed for binarized neural networks (BNNs) such as *EEV* achieve significant computational efficiencies by working with compact, binary-weighted networks, yielding order-of-magnitude speedups over traditional floating-point models [52]. Emerging methods, such as convex hull approximations and softmax abstractions used in tools like *PRIMA*, are extending verification capabilities to multi-modal, transformer-based, and recurrent architectures, further expanding the scope of modern verification frameworks [67], [68].

Despite these advancements, enduring challenges remain across all methodologies. The trade-off between precision and scalability continues to dominate the landscape: SMT and optimization approaches excel in precision but scale poorly, while abstraction and reachability methods prioritize scalability at the expense of tight guarantees. Addressing numerical instabilities, improving computational efficiency for non-linear activation functions, and ensuring broad applicability to large-scale industrial benchmarks are persistent hurdles [25], [59]. Additionally, the integration of symbolic-numeric hybrids and probabilistic guarantees via Bayesian methods are promising directions for tackling these limitations [23], [77].

In conclusion, this comparative analysis underscores the absence of a one-size-fits-all verification tool. The choice of tools heavily depends on the neural network architecture, the verified property, and the application domain. As research progresses, combining abstraction techniques, symbolic solvers, and high-performance hardware accelerations holds significant promise for addressing current scalability and precision gaps. These integrations will be key to realizing robust, industrial-scale verification pipelines, particularly in safety-critical domains like autonomous systems and healthcare, where trust in neural networks is paramount.

4.3 Benchmarking Datasets and Performance Evaluation

Benchmarking datasets and performance evaluation in the domain of formal verification of neural networks play a critical role in assessing tool capabilities, fostering progress, and

ensuring comparability across methodologies. This subsection delves into the standardized datasets commonly used for benchmarking, the evaluation metrics that underpin performance analysis, and the challenges associated with establishing meaningful comparisons among tools.

The adoption of standardized datasets such as ACAS Xu [14], [39] has emerged as a cornerstone in the evaluation of safety-critical neural network verification. ACAS Xu, a collision avoidance system for unmanned aerial vehicles, serves as a benchmark due to its real-world significance and intrinsic complexity. Its piecewise-linear nature and small-scale architecture make it an ideal candidate for stress testing a wide range of verification tools, from SMT-based solvers [15] to reachability-based approaches [78]. Tools like Marabou and NNV routinely validate their performance on ACAS Xu to demonstrate efficacy under adversarial robustness, correctness, and safety guarantees.

Datasets focused on image classification tasks, such as MNIST and CIFAR-10, have also been utilized extensively, providing a distinct perspective on evaluating verification tools in high-dimensional input domains. These datasets often serve as proxies to assess robustness to adversarial inputs. Their general-purpose nature allows researchers to study model performance under input perturbations, nonlinear transformations, or bounded distortions [37], [61]. The computational challenges imposed by these benchmarks enable comparative studies of tool scalability, with rapidly increasing dimensions of the problem pushing scalability limits in modern solvers.

Performance evaluation extends beyond the choice of datasets, hinging on robust and standardized metrics. Metrics such as verification time, memory usage, and soundness of results (measuring precision and false positives) form the backbone of empirical assessments. For instance, tools like Reluplex and its successor Marabou have demonstrated their ability to deliver sound results with reduced runtime overhead compared to prior methods [14]. Similarly, verification toolkits like ReluVal leverage symbolic interval arithmetic to address memory-constrained scenarios, achieving up to 200x speedup compared to Reluplex in some benchmarks [37]. A growing focus on multi-objective evaluation metrics—balancing scalability, accuracy, and computational efficiency—is increasingly driving innovative hybrid methodologies [60].

Despite significant progress, challenges abound in achieving broad applicability and universality in formal verification benchmarks. The diversity of neural network architectures complicates evaluations: benchmarks designed for ReLU-based feedforward neural networks (e.g., ACAS Xu) do not necessarily generalize to transformers, spiking neural networks, or recurrent architectures [78], [79]. Moreover, most existing benchmarks do not capture the stochastic and hybrid nature of real-world systems, especially those with probabilistic environmental interactions or multi-modal inputs [80]. This limitation underscores the need for datasets that incorporate diverse dynamical properties while ensuring the mathematical tractability of verification methods.

In recent years, community-driven initiatives like VNN-COMP [14], [59] have set the stage for rigorous tool comparisons by standardizing datasets, model formats (e.g., ONNX), and property specifications (e.g., VNN-LIB). These efforts have significantly reduced ambiguities, enabling fair and transparent evaluations. However, ensuring representativeness across diverse application domains, such as healthcare diagnostics, autonomous vehicles, and finance, remains a formidable challenge.

Emerging trends point toward the integration of more realistic benchmarks, particularly those grounded in control systems with neural controllers. For example, benchmarks involving dynamical system feedback loops [28], [81] are expanding the context in which performance and safety guarantees are assessed. Such benchmarks test the tool's ability to reason about coupled system behavior and verify closed-loop properties, paving the way toward novel metrics that prioritize system-level safety.

Going forward, collaboration across the research community to expand benchmark catalogs and align evaluation practices is imperative. Benchmarks that account for real-world challenges such as data-driven nonlinearity, stochasticity, and scalability beyond theoretical constraints hold the potential to accelerate advancements. Continued innovations in benchmarking frameworks will undoubtedly enhance the robustness and relevance of future verification systems.

4.4 Integrating Verification Tools into ML Development Pipelines

Integrating verification tools into modern machine learning (ML) development pipelines presents distinct challenges and opportunities, fostering a closer alignment between performance optimization and adherence to safety, robustness, and fairness requirements across a model's lifecycle—from training to deployment. Seamlessly embedding verification processes into ML workflows has the potential to enhance system reliability, particularly in safety-critical domains such as autonomous systems, medical diagnostics, and financial decision-making.

A key challenge lies in reconciling the computational demands of verification tools with the resource limitations of large-scale industry-grade ML models. Tools like Marabou and Reluplex [26] excel at verifying ReLU-activated networks but often face scalability constraints when applied to modern architectures such as transformers or convolutional neural networks (CNNs). This incompatibility is further amplified by standard practices in ML engineering, where rapid prototyping and iterative model refinement demand short turnaround times—an area where symbolic reasoning and reachability analysis often fall short.

An emerging approach to address this challenge involves the integration of verification objectives directly into the training process. By incorporating property checks as implicit constraints, this strategy guides the training of neural networks to produce inherently verifiable models. Techniques like differentiable logic [66] enforce specific properties—such as weight sparsity or neuron stability—through modified loss functions. This proactive approach reduces both the complexity of post-training verification and the overall computational burden, ensuring robust models without a significant trade-off in efficiency [66].

Despite advancements in training methodologies, pre-deployment verification remains indispensable for certifying final models against safety and fairness requirements. Tools such as ERAN and β-CROWN [46] excel in this domain by leveraging bound propagation and polyhedral analysis to analyze properties like adversarial robustness and fairness. However, even these improved frameworks often face trade-offs between precision and efficiency, with over-approximations leading to inconclusive verifications in some scenarios.

To mitigate these challenges, hybrid frameworks that combine incomplete and complete verification methods are gaining prominence. For instance, α-β-CROWN [46] adopts a branch-and-bound strategy with GPU acceleration to streamline verification for high-dimensional domains. Similarly, incremental techniques like IVAN [82] focus on localized verification during iterative updates or fine-tuning of models, avoiding redundant computations and further reducing runtime.

Another critical factor for industry adoption is the compatibility of verification tools with widely-used ML frameworks. Standard model formats like ONNX ensure interoperability, creating seamless integration pathways between frameworks such as PyTorch or TensorFlow and verification tools [14]. This interoperability, coupled with growing interest in user-friendly property specifications, enables practitioners to embed fairness, robustness, or monotonicity requirements without significant disruption to their workflows. Innovations in neuro-symbolic verification [19] further enhance usability by translating real-world requirements, such as "the system must stop when detecting a stop sign," into formally verifiable conditions.

Verification tools also play a vital role in debugging and enhancing explainability within ML pipelines. Abstraction-based approaches—such as DeepAbstract [21]—simplify complex models, enabling developers to identify and address vulnerabilities in robustness or fairness. This iterative refinement supports not only model debugging but also provides insights into key decision boundaries, rendering the verification process more interpretable.

Future research must address scaling verification tools for architectures characterized by dynamic attention mechanisms, interdependencies, or large-dimensional spaces found in models like transformers and graph neural networks [7]. Advances in computational infrastructure, such as optimizing verification through GPU utilization or employing cloud-native distributed systems, have shown promise, as demonstrated by systems like GPUPoly [44]. Continued exploration of hybrid verification strategies and statistical methods for approximate guarantees will further expand the scope of practical verification deployments.

In summary, integrating verification into ML development pipelines bridges the gap between formal guarantees and state-of-the-art deep learning practices, ensuring safe and reliable deployment of models in real-world applications. Enhancing scalability, fostering tighter coupling between training and verification processes, and maintaining compatibility with evolving architectures will be essential for broader adoption. By addressing these challenges, verification-driven ML workflows can significantly reduce operational risks while ensuring robust and trustworthy systems throughout their lifecycle.

4.5 Evaluating Scalability and Real-World Applications of Tools

Scalability represents one of the most pressing challenges in the formal verification of neural networks, especially given the increasing complexity of architectures like transformers, deep convolutional models, and graph neural networks. Current verification tools grapple with the combinatorial explosion in constraints when larger, deeply-layered networks or high-dimensional inputs are involved. This subsection discusses advancements in tackling these scalability constraints, examines real-world applications of verification tools, and highlights emerging techniques aimed at bridging the gap between theoretical research and practical deployment.

A major bottleneck in scaling verification tools lies in the computational complexity of handling non-linear activation functions (e.g., ReLU, sigmoid) or piecewise-linear approximations across deeply layered networks. While tools like Reluplex and Marabou [3], [15] excel at solving satisfiability problems for feed-forward ReLU-based networks, their performance can degrade severely for larger architectures, due to exponential branching in constraint solving. Similarly, mixed-integer programming (MIP)-based methods can struggle to scale due to the reliance on explicit enumeration of ReLU activation cases [13]. Techniques like branch-and-bound, integrated into the MIP framework, have demonstrated improvements by exploiting advanced branching strategies on non-linearities, but scalability remains limited when applied to convolutional or recurrent networks.

Some contemporary tools leverage reachability analysis to combat these limitations. Layer-by-layer output reachability estimation, as seen in methods employing polyhedra and zonotopes, has yielded promising results [33], [41]. Reachability techniques function well in specific low-dimensional scenarios, but they face challenges when generalizing to high-dimensional input domains or hybrid models like neural controllers in cyber-physical systems.

Hybrid methods have emerged as a powerful strategy to address scalability. By combining symbolic and numerical approaches, tools such as POLAR effectively reduce error accumulation while enhancing efficiency using symbolic remainders and Bernstein Bézier forms [71]. Additionally, frameworks like DeepNNC integrate Lipschitz continuity for unrolling complex dynamical behaviors, enabling analysis of practical adaptive systems [58]. Hybrid verification pipelines exemplify the growing trend of integrating conceptual advances from verification literature with hardware acceleration techniques, such as GPU parallelism, to handle larger architectures.

In real-world applications, verification tools have demonstrated their utility in ensuring safety in critical domains such as autonomous driving, medical diagnostics, and cyber-physical systems [28], [38]. Specifically, tools like NNV employ exact and over-approximate reachability methods to certify robust behavior in domains such as adaptive cruise control or collision avoidance, underscoring the importance of practical validations. Similarly, Verisig showcases how verification frameworks can adapt sigmoid-

based controllers into hybrid system analysis, further broadening the scope of safe deployments.

Despite these advances, a trade-off persists between depth (accuracy of verification guarantees) and breadth (applicability to large-scale systems). Abstraction-refinement techniques, like those in Cnn-Abs, have proven effective for convolutional architectures through iterative abstraction cycles that simplify the verification problem at each stage [51]. These methods allow high-dimensional verification tasks to become computationally tractable, albeit with limitations in interpretability and runtime consistency. Meanwhile, emerging tools such as Neuro-Symbolic Verification suggest exciting future possibilities by integrating symbolic specifications into verification pipelines, enabling checks on richer, real-world properties [19].

Scaling verification efforts further will require adopting advanced computational strategies, such as transformer-specific optimizations or even leveraging quantum computing for tight constraint solving. Efforts like VNN-COMP provide critical benchmarking standards for evaluating scalability and comparability across tools, driving progress in the field [59]. Furthermore, foundational research into approximate verification introduces the possibility of scalable statistical techniques that complement formal robustness guarantees with probabilistic bounds [56].

In summary, scalability remains the cornerstone challenge in formal verification of modern neural networks. While hybrid methodologies and real-world tools such as NNV and Verisig demonstrate meaningful applications, new advances in hardware optimization, algorithmic abstraction, and neuro-symbolic design hold the key for future breakthroughs. Expanding these tools to address emerging architectures and ensuring their broad adoption in safety-critical domains represent vital next steps in this growing field.

4.6 Future Directions for Tool Development and Standardization

The future development of tools for the formal verification of neural networks (FNNs) must address critical limitations in scalability, standardization, interoperability, and usability, while fostering a collaborative ecosystem of community-driven benchmarks and open-source contributions. As the complexity and ubiquity of neural networks in high-stakes applications grow, verification tools must not only advance to manage modern architectures but also integrate seamlessly into machine learning (ML) pipelines, enabling efficient and practical deployment at scale.

A central avenue for progress lies in enhancing computational efficiency and scalability through specialized hardware and innovative algorithmic designs. Given the sheer scale of contemporary neural networks, tools such as GPUPoly [44], which leverage GPU-based optimizations for polyhedral techniques, highlight the critical importance of hardware acceleration in improving verification efficiency. Similarly, distributed verification frameworks explored in [62] offer promising solutions to computational bottlenecks, enabling verification of deeply layered networks or multi-modal architectures. Furthermore, the potential integration of quantum computing resources into the landscape of neural network verification represents an exciting frontier, offering exponential improvements for specific problem classes. Such breakthroughs could fundamentally reshape the field by making real-time verification of complex and adaptive deep learning models a feasible goal.

Standardizing tool interfaces and model formats is equally crucial for enhancing interoperability and facilitating broader adoption. Initiatives like VNN-LIB [24], which define standardized benchmarks and input-output specifications, have already improved comparability and reproducibility across tools. The extension of these efforts to encompass compatibility with widely-adopted ML frameworks such as ONNX and PyTorch could dramatically reduce friction in integrating verification tools into practical pipelines. Tools like Marabou [15] exemplify this trend, showcasing the potential of adaptable frameworks to bridge traditional formal methods with ML workflows. Moreover, frameworks designed to incorporate hybrid and neuro-symbolic systems [83] offer a path toward capturing the complex behaviors of real-world applications.

Community-driven benchmarking initiatives, such as VNN-COMP [59], play a pivotal role in driving innovation and setting standards for evaluation. However, current benchmarks often lack sufficient diversity in neural network architectures and real-world complexities. Expanding future benchmarks to include emerging architectures, such as transformers, graph neural networks, and spiking neural networks, would better reflect real-world requirements and stress-test verification tools comprehensively. Additionally, the establishment of domain-specific benchmarks capturing ethical and socially meaningful criteria, such as fairness, transparency, and sustainability [19], could catalyze progress in aligning verification practices with broader societal values.

A further challenge lies in bridging the gap between ML training and verification processes. Co-design methodologies, such as those detailed in [66], align training strategies with verification objectives by introducing verifiability constraints during model development. These approaches limit the verification problem's complexity by narrowing the post-training solution space, addressing scalability concerns. Similarly, abstraction-refinement frameworks, such as those described in [45], combine computational efficiency with precision, though additional refinements are necessary to handle dynamic systems, such as lifelong learning or evolving neural networks, at scale.

Finally, improving accessibility and usability remains essential to broadening the utility of verification tools across interdisciplinary teams from academia and industry. Many existing tools require significant expertise in formal methods, creating barriers to widespread adoption. User-friendly interfaces, including interactive visualizations and interpretable proof outputs, could lower these barriers, enabling non-expert users to assess verification results effectively. Additionally, frameworks capable of producing certifiable proofs [23] could enhance trust in verification tools, particularly in safety-critical domains, by providing independently verifiable guarantees.

In conclusion, advancing the formal verification of neural networks necessitates a cohesive research agenda that balances innovation in computational techniques, standard-

ization efforts, seamless integration into ML workflows, and user-centric design. By addressing these priorities, the field can develop robust, scalable, and transparent verification tools that underpin the trustworthy deployment of neural networks in critical, high-assurance applications.

5 Applications of Formal Verification in Neural Networks

5.1 Formal Verification in Autonomous Systems and Robotics

Formal verification has become an indispensable methodology for ensuring the correctness and reliability of neural networks deployed in autonomous systems and robotics. As these systems increasingly operate in safety-critical and uncertain environments, from self-driving cars to industrial robots, it is paramount to provide mathematical guarantees of safe behavior. Neural networks used in perception, decision-making, and control components play a pivotal role in these applications but pose significant verification challenges due to their non-linear and high-dimensional properties.

In the domain of autonomous vehicles, formal verification techniques are crucial for certifying properties such as collision avoidance, robust lane tracking, and adherence to environmental constraints. For example, reachability analysis, a technique that computes the set of all possible outputs given an input range, has been widely applied to verify safety properties of autonomous maneuvers. Tools like NNV [38] have shown efficacy in analyzing neural networks integrated into safety-critical subsystems, such as the adaptive cruise control in vehicles, where precise control and accurate perception are imperative. Similarly, formal methods have been adapted to verify network properties in the context of aircraft collision avoidance systems, as demonstrated in methodologies leveraging mixed-integer programming for certifying bounded perturbation robustness [3], [10].

A core strength of formal verification in this setting is its ability to detect subtle and catastrophic failures arising from neural network misclassification or decision errors. Methods such as satisfiability modulo theories (SMT), abstract interpretation, and optimization-based approaches have been tailored for the unique requirements of robotics and autonomous systems. SMT-based methods, used for analyzing ReLU-activated networks, have been employed to certify neural-network-based motion planning algorithms, ensuring safe navigation under environmental constraints [13], [15]. These approaches are particularly effective in discrete decision frameworks, such as obstacle avoidance or route selection.

Beyond perception and decision-making, the use of neural networks in real-time control has prompted the need for verifying closed-loop behaviors under continuous dynamics. For example, techniques combining formal reachability analysis with differential dynamic logic (dL) enable the verification of neural network controllers embedded in cyber-physical systems over infinite-time horizons [84]. Such approaches bridge neural network verification and hybrid system analysis, providing guarantees on the stability and safety of autonomous robots in dynamic and adversarial settings. Moreover, abstraction-refinement methods tailored for convolutional architectures used in image-based perception tasks help address the scalability issues inherent in verifying large-scale networks [51].

Industrial robotics is another domain where formal verification has demonstrated utility. Robots operating in manufacturing and warehouse settings must comply with stringent safety regulations and demonstrate consistent behavior despite high variability in operational conditions. Ensuring these properties often involves a mix of simulation-based testing and abstraction-driven certification. Simulation-based falsification, for instance, has been successfully integrated with counterexample analysis to refine neural network models and eliminate failure cases in robotic systems [20].

However, the application of formal verification in large-scale robotics comes with notable trade-offs. Although exact methods yield high-precision guarantees, they often suffer from scalability limitations, particularly for deeply layered neural networks with non-linearities, such as those used in modern autonomous systems. Approximation techniques, though computationally efficient, may produce conservative bounds, resulting in reduced confidence or false negatives [2], [73]. Emerging hybrid approaches that integrate symbolic and numerical methods offer promising advances, balancing precision with computational feasibility [12].

Looking forward, a key challenge lies in aligning formal verification frameworks with real-world application scales and dynamics. The increasing adoption of complex architectures, such as transformers and graph neural networks, within autonomous systems will necessitate extending current verification methods to address their unique properties [7]. Additionally, the development of hardware acceleration (e.g., GPUs, TPUs) and distributed verification platforms holds the potential to overcome the computational inefficiencies that currently hinder large-scale adoption [74]. Finally, collaborative efforts involving standardized benchmarks, such as those supported by VNN-COMP [59], will be essential to foster innovation and establish trust in neural network-driven autonomous systems.

In sum, formal verification plays a critical role in ensuring the safe operation of autonomous systems and robotics. While significant progress has been made, emerging trends in neural network design and the growing complexity of deployment environments continue to drive research toward scalable, robust, and adaptive verification frameworks.

5.2 Healthcare Applications of Neural Network Verification

Formal verification of neural networks has emerged as a critical approach in healthcare, addressing the stringent demands for precision, safety, and equity in medical diagnostics and decision-support systems. As neural networks (NNs) are increasingly integrated into domains such as medical imaging, drug discovery, assistive robotics, and personalized medicine, ensuring their robustness and correctness is of paramount importance. Unlike conventional empirical methods, formal verification provides mathematical guarantees, making it particularly suited for healthcare applications where failure can result in life-threatening out-

comes and where compliance with regulatory and ethical standards is imperative.

A key application of formal verification in healthcare is the **safety assurance of diagnostic systems**, such as those analyzing medical imaging for tasks like cancer detection or cardiovascular risk assessment. These systems must reliably classify images even when subject to perturbations, such as input noise or variations in data quality. For instance, interval bound propagation (IBP) methods have demonstrated scalability in certifying large networks while maintaining computational efficiency [65]. By establishing tight bounds on the output deviations under certified constraints, IBP ensures that diagnostic models remain robust to minor variations in high-dimensional medical images, such as X-rays and MRIs. Furthermore, tools like NNV [38] use reachability analysis to compute precise output bounds, verifying safety properties critical to imaging-based systems and sensor-driven classifiers alike.

Another impactful area of application is in **personalized medicine and drug discovery**, where NNs are leveraged to model highly nonlinear chemical and biological interactions. Errors in these models can have far-reaching consequences, such as predicting adverse drug reactions or recommending inappropriate treatments. Formal verification ensures that predictive models adhere to safety constraints, even in the context of high-throughput analytics. Mixed-integer programming (MIP), as illustrated in [27], rigorously certifies piecewise-linear neural networks, making it particularly effective for validating predictions of drug interactions in sizable datasets with up to millions of combinations. This capability is essential for ensuring the reliability of models tasked with designing or optimizing personalized treatment protocols grounded in genomic and proteomic data.

Beyond robustness, **fairness verification** in healthcare focuses on preventing bias in critical models, particularly in applications where outcomes have direct societal and ethical implications. Diagnostic models used for patient screening or resource allocation must ensure fairness across demographic subgroups to avoid perpetuating or exacerbating health inequities. Frameworks like those proposed in [30] and [29] evaluate fairness constraints at both the individual and global levels. These methodologies ensure alignment with ethical mandates by verifying that similar patients receive consistent outcomes and that biases across group-level metrics are minimized.

The scope of formal verification extends further to **medical robotics and assistive devices**, where neural networks control systems such as surgical robots, exoskeletons, and prosthetics. These safety-critical systems, particularly when integrated into feedback loops, demand rigorous analysis to prevent malfunctions or unsafe behaviors. Techniques like Lipschitz constant-based robustness verification [85] provide certifiable bounds on the impact of small disturbances in sensor or controller feedback systems. Meanwhile, hybrid approaches such as Verisig [28] translate NN-based controllers into hybrid systems for reachability analysis, proving effective for assistive systems operating in dynamic, real-world settings involving biomechanical interactions.

Despite these advancements, challenges remain in scaling formal verification to meet the demands of large and complex models, such as those used in genomics or precision oncology. Optimization frameworks like convex relaxation [4] have achieved notable progress but still face high computational costs that escalate with model complexity. Moreover, emerging architectures like transformers and graph-based neural networks, frequently employed in protein folding and drug design, present additional challenges due to their intricate dependencies and dynamic structures [68]. To address these issues, advances in high-performance computing, including GPU-accelerated pipelines [44], and the development of hybrid verification methodologies [13] will be critical.

As regulatory bodies increasingly emphasize transparency and accountability in medical AI, the role of formal verification in healthcare is set to expand. Future directions include integrating verification directly into training workflows, such as predictor-verifier co-training [86], to develop models that are inherently robust and verifiable. Probabilistic approaches to verification are also gaining traction, particularly in addressing uncertain or noisy datasets [77]. Furthermore, embedding domain-specific ethical standards into verifiable specifications, as seen in [84], can ensure compliance with regulatory and societal expectations. By tackling these challenges, formal verification is poised to become an indispensable pillar in the deployment of safe, equitable, and reliable neural network-driven systems within healthcare.

5.3 Finance and Risk-sensitive Decision Systems

Formal verification in finance and risk-sensitive decision systems represents a pivotal area where robust guarantees are essential for ensuring trust, compliance, and operational efficiency. Neural networks (NNs) have increasingly emerged as integral components in financial systems, being employed in tasks ranging from fraud detection to credit scoring, and high-frequency trading. However, their reliability, interpretability, and alignment with stringent regulatory frameworks demand rigorous formal verification methods.

One critical domain of application is fraud detection, wherein NNs are utilized to identify anomalous patterns in transactions. Although effective, these systems face challenges stemming from adversarial attacks aiming to bypass detection. Formal verification provides rigorous guarantees against such perturbations by analyzing the robustness of neural networks under adversarial scenarios, ensuring that fraudulent transactions cannot manipulate model outputs into false negatives. For instance, symbolic interval techniques [37], which compute tight bounds on NN behavior without relying on computationally intensive SMT solvers, have shown exceptional scalability. In the financial setting, these techniques enable robust certification of transaction classifiers even when handling dynamically changing adversarial inputs. Nonetheless, these methods suffer from over-approximation trade-offs, wherein the guaranteed bounds can remain overly conservative, leaving room for further refinement.

Credit scoring and risk evaluation systems have also gained attention due to their significant societal implications. NNs employed for assessing creditworthiness can inadvertently embed biases, such as discrimination based

on race or socioeconomic factors. Ensuring fairness is thus paramount. Formal verification frameworks specifically targeting fairness aspects, such as adversarial robustness and monotonicity of scores, have been explored. Hybrid approaches combining counterexample-guided abstraction refinement (CEGAR) and fairness certificate generation, as highlighted in [87], offer an efficient mechanism to analyze whether a credit scoring model produces outputs that systematically disadvantage any demographic group. These refinements drive computational efficiency but remain challenged by non-linear activation mechanisms common in deeper architectures.

In high-frequency algorithmic trading, formal verification ensures that NN controllers deployed in volatile trading environments satisfy safety properties under real-time constraints. For instance, ensuring that bid-ask spreads remain within predefined bounds or that trading strategies avoid cascading losses forms the backbone of regulatory compliance. Reachability analysis methods, like those discussed in [33], can compute the limits of market-safe operations for NN-driven trading algorithms. Despite their promise, the capacity of these methods is often limited by scalability challenges, particularly when neural architectures interface with time-sensitive systems, such as predicting market volatility.

An emerging trend involves integrating formal verification directly into training workflows. This design philosophy aligns with the paradigm of "verification-aware training," where constraints derived from regulatory norms or economic safety are embedded during model optimization. For example, differentiable formulations of fairness, monotonicity, or boundedness can be encoded as auxiliary loss terms, ensuring that the learned models not only fit the data but also adhere to verifiable safety properties. This idea resonates with findings in [79], where constrained optimization aids the design of safe decision systems, and could be adapted to financial systems to achieve similar objectives.

Looking to the future, the primary challenge in formal verification of financial NNs is balancing precision, scalability, and interpretability. Current SMT-related methods offer high precision but often falter in handling large-scale models. Conversely, computationally scalable methods based on symbolic intervals or interval arithmetic excel in runtime efficiency but sacrifice analytical tightness. Moreover, ensuring domain-centered explainability—vital for stakeholder trust and regulatory audits—requires enhancements beyond black-box robustness analysis. Emerging neurosymbolic approaches, which combine symbolic reasoning with numerical domains [19], promise a way forward by bridging this interpretability gap.

In conclusion, formal verification mechanisms are indispensable tools for ensuring the dependability and ethical deployment of NNs in financial tasks. By addressing robustness against adversarial attacks, fairness in credit scoring, and safety in trading algorithms, these tools foster trust and enable compliance with high-assurance regulatory environments. Future trajectories must focus on scalability innovations, such as GPU acceleration or distributed computing [41], and standardization in specifying fairness and safety requirements. A community-driven benchmarking effort parallel to VNN-COMP could catalyze the dissemination and validation of cutting-edge verification methods targeting financial systems.

5.4 Adversarial Defense Mechanisms

Adversarial attacks pose a significant threat to the deployment of neural networks in safety-critical domains, where even minor perturbations to input data can lead to catastrophic outcomes. When aligned with the objectives of ethical AI deployment and fairness assurance discussed in the surrounding subsections, formal verification emerges as a powerful tool for ensuring resilience against adversarial examples through rigorous and systematic analysis. This subsection delves into the applications of formal verification in adversarial defense, the methodologies utilized, their strengths and limitations, and the emerging trends that promise to advance this critical area.

Formal verification-based adversarial defense mechanisms aim to certify robustness against adversarial examples by providing mathematical guarantees within specific perturbation norms (e.g., L_∞, L_2). These guarantees either prove that no adversarial example exists within a defined bound or identify vulnerabilities for subsequent mitigation. Mixed-integer programming (MIP) has gained traction in this domain, with formulations that precisely encode L_∞ constraints to evaluate neural network robustness [27]. While MIP-based approaches deliver highly precise outputs, their scalability is limited when applied to deeper and more complex neural networks due to the exponential growth of constraints.

To address scalability challenges, convex relaxation techniques have been widely adopted. These methods approximate non-linear activation functions, such as ReLU, using linear constraints to facilitate more tractable robustness certification. Advances in this area, such as those discussed in [4] and [67], refine the bounding process for perturbations propagated through network layers, yielding stronger guarantees for piecewise linear networks. However, this strategy poses an inherent trade-off: tighter convex approximations improve precision but increase computational overhead, whereas coarser relaxations may simplify computation at the expense of overly conservative guarantees.

A complementary paradigm, interval-bound propagation, has demonstrated particular promise for high-dimensional and large-scale networks. Techniques such as β-CROWN [46] and adaptive relaxation methods [12] combine computational efficiency with enhanced bound tightness using parallelization optimized for GPUs. For instance, β-CROWN efficiently accelerates robustness analysis while ensuring tighter bounding operations, enabling the verification of larger and more complex architectures compared to conventional approaches reliant on linear programming.

Beyond post-training verification, recent advancements have emphasized integrating verification mechanisms directly into the training process. These frameworks, often termed verification-aware training, establish a synergistic feedback loop between training and robustness analysis. Approaches such as inducing ReLU stability during optimization significantly reduce verification complexity while enhancing adversarial robustness [66]. This co-design philosophy not only improves robustness but aligns with the

forward-looking principles highlighted in other subsections, such as verification-aware design for fairness and safety-critical domains.

Probabilistic robustness verification has also emerged as an intriguing alternative to deterministic verification frameworks. Incorporating stochastic considerations, Bayesian neural networks (BNNs) model uncertainty in weights to yield rigorous probabilistic robustness bounds [88]. While this paradigm offers nuanced insights into robustness under stochastic perturbations, the associated computational cost—stemming from the high dimensionality of probabilistic distributions—limits its scalability, especially for industrial-scale systems.

Despite these advancements, several challenges persist in achieving a balance between scalability and precision. Verifying architectures with sophisticated, non-linear activations (e.g., Sigmoid, Tanh), as well as multi-modal frameworks like Transformers, remains difficult due to the highly non-linear computation graphs [89]. Furthermore, scaling verification techniques to match the vast complexity of real-world networks demands innovative computational solutions. Promising avenues include GPU-accelerated polyhedral techniques [44] and abstraction-based scalability techniques [90], both capable of reducing computational bottlenecks while maintaining rigorous guarantees.

Looking ahead, hybrid verification approaches harmonizing symbolic and numerical methods may define the next frontier by mediating trade-offs between scalability and precision [73]. Collaborative benchmarks like VNN-COMP, which have successfully galvanized development in other verification domains, provide a model for advancing innovation in adversarial robustness verification [14]. Furthermore, embedding robustness directly into model training through differentiable constraints or adversarially augmented datasets represents a paradigm shift that promises seamless scalability while maintaining rigorous guarantees in real-world systems.

In conjunction with the broader goals of fairness and safety outlined in surrounding sections, formal verification for adversarial defense mechanisms offers critical contributions to safe and trustworthy neural network deployment. By leveraging rigorous methodologies to ensure robustness against perturbations, formal verification acts as both a safeguard and a driver of innovation for applications requiring high-assurance performance. As research continues to refine this interplay between computational feasibility and real-world applicability, integrating robustness guarantees into the model design and training phases will become an indispensable strategy for achieving security and resilience in modern AI systems.

5.5 Ensuring Ethical and Fair AI Decisions

In recent years, the ethical and social implications of neural network-driven decision systems have drawn substantial attention, particularly in high-stakes areas such as hiring, credit scoring, legal decision-making, and healthcare. These systems risk perpetuating or amplifying existing societal biases due to imbalances in training data, opaque learning processes, and non-linear decision boundaries. Ensuring fairness and equity, therefore, is a critical goal—one that formal verification methods are uniquely capable of addressing. This subsection examines how formal verification frameworks can rigorously certify fairness properties and align neural networks with ethical guidelines, highlighting their strengths, challenges, and future research directions.

Fairness in neural networks can be formalized in terms of mathematical criteria that ensure equitable outcomes across sensitive attributes (e.g., race, gender, socioeconomic status). Such criteria often include notions like statistical parity (equal acceptance rates across demographic groups), equalized odds (equal error rates across groups), and counterfactual fairness (consistent decisions for an individual regardless of hypothetical changes in sensitive attributes). Formal verification approaches establish these properties by proving whether all possible inputs and computational pathways in a neural network adhere to these fairness constraints [87]. For example, the specification of fairness constraints can be encoded into satisfiability modulo theories (SMT), as demonstrated in SMT-based methods tailored to binarized neural networks [91].

State-of-the-art verification tools such as Marabou [15] and ReluVal [37] have been adapted to certify fairness properties through symbolic and interval-based methods. These tools analyze piecewise linear neural networks, such as those employing ReLU activation functions, by encoding fairness constraints into their reachability or feasibility exploration processes. However, one inherent challenge is scalability. Constraining a model to satisfy fairness requirements over high-dimensional input spaces often results in combinatorial explosion. Hybrid methods, such as the combination of abstraction and numeric reasoning in DeepAbstract [21], provide a compelling way forward by simplifying neural network complexity while maintaining soundness in fairness guarantees.

An emerging trend in the verification of fair AI systems is the integration of quantitative fairness metrics directly into the training process. By embedding fairness properties as differentiable constraints in training objectives, it is possible to align optimization routines with ethically sound network behaviors. For example, ReachNN employs symbolic propagation techniques to propagate sensitive attributes across network layers, detecting subtle disparities in decision boundaries [54]. Complementary to these methods, data-driven verification frameworks employ probabilistic guarantees, leveraging PAC learning principles to certify group fairness within statistical limitations [92]. However, these empirical approaches trade off formal soundness for scalability, posing challenges when applied to safety-critical or regulated environments.

Fairness verification is not merely an algorithmic challenge; it also demands interpretability. Expanding on the explainable AI (XAI) paradigm, verification outputs must provide actionable insights into why a model fails certain fairness constraints and how such failures arise. Techniques like residual reasoning, proposed in [61], utilize intermediate abstractions to isolate fairness violations, enabling a more targeted refinement of the network. Similarly, advancements in concept-based analysis [93] redefine specifications in terms of high-level semantic concepts, aligning human interpretable descriptions with formal guarantees.

Despite these advances, guaranteeing fairness in neu-

ral networks remains fraught with challenges. The tension between fairness and utility underscores the complexity of applying rigorous verification standards. As fairness constraints are integrated, there is often a degradation in model performance—a trade-off requiring careful calibration. Moreover, fairness criteria themselves vary in interpretation, making reconciling multiple fairness notions (e.g., statistical parity versus individual fairness) a non-trivial formalization task [87]. Furthermore, auditing fairness over non-linear architectures like transformers or multi-modal neural networks, which are becoming increasingly prevalent in decision-making applications, exacerbates both the computational and theoretical challenges [79].

Future research must address scalability by leveraging advancements such as distributed computing architectures [59] and hybrid symbolic-numeric solutions, which have already shown promise in optimizing fairness verifications [13]. Moreover, an interdisciplinary approach that incorporates ethical guidelines, legal principles, and technical constraints will shape the development of fairness standards. Initiatives such as VNN-COMP [59] offer a blueprint for establishing benchmarking frameworks and shared tools to foster collaboration in this burgeoning field.

The quest for ethical and fair neural networks poses a dual challenge: advancing technical rigor while accommodating evolving societal expectations. Formal verification, with its mathematical guarantees, serves as both a safeguard and an enabler, ensuring that AI systems can uphold equity and justice in real-world decision-making.

5.6 Emerging Domains and Real-world Implementations

Formal verification of neural networks is progressively extending its reach beyond traditional domains, finding applications in emerging and real-world settings where robustness, safety, and ethical compliance are paramount. Building upon foundational advancements in fairness and interpretability, this subsection investigates these pioneering application areas, highlighting distinctive implementations and exploring their associated challenges.

Energy and power systems are among the foremost emerging domains benefiting from formal verification techniques. Neural network models are increasingly used in energy grid management, renewable energy optimization, and fault detection tasks, where the reliable operation of these systems is crucial under diverse and potentially volatile conditions. Verification ensures that these models operate robustly even in scenarios involving power fluctuations, hardware degradation, or adversarial perturbations. Robustness verification methods such as mixed-integer programming (MIP) [27] and reachability analysis [48] provide guarantees against extreme or adversarial input scenarios, thereby reinforcing the stability of system performance. Yet, scalability remains a significant challenge when addressing the high-dimensional neural networks used in such domains. Hybrid techniques that integrate symbolic reasoning with interval-based approximations have shown promise in bridging this gap [44].

In wildlife conservation and disaster management, neural networks have been deployed for tasks such as species identification, forest fire detection, and early warnings for hurricanes or floods. These mission-critical applications depend on the reliability of models under constrained computational resources and noisy, real-world data inputs. Formal verification plays a pivotal role in such scenarios by enabling quantitative verification frameworks that certify probabilistic robustness [90] or interval-based reachability methods [94]. However, practical deployment is hindered by the inherent difficulty in defining adaptable formal specifications that account for diverse, region-specific environmental conditions, as underscored by similar challenges in safety-critical system verification [2].

Transportation infrastructure captures another critical domain where verified neural networks have demonstrated potential in tasks like predictive maintenance, railway signal control, and air traffic management. Failures in these applications can lead to catastrophic consequences, emphasizing the importance of ensuring stringent safety verification. Closed-loop verification techniques, exemplified by tools such as Verisig [28], extend verification capabilities by explicitly accounting for real-world dynamics in hybrid systems. Scalability, especially in latency-sensitive applications like air and rail traffic management, remains an ongoing challenge. Hardware-efficient verification approaches leveraging GPUs have shown considerable promise in mitigating these constraints [44], though the trade-off between computational efficiency and verification precision persists as a critical limitation.

Interactive AI systems, including educational assistants, medical chatbots, and customer service agents, are also becoming a burgeoning area for formal verification. Beyond robustness and safety, these systems necessitate the verification of properties such as fairness, monotonicity, and correctness to ensure ethical compliance and reliable functionality. Techniques like differential verification, which analyzes relationships between pairs of neural models [95], are particularly well-suited for identifying regressions in fairness, robustness, or other critical properties during model updates. However, the challenge of aligning real-time verification mechanisms with continuously learning and evolving neural models complicates their practical deployment in interactive systems [96].

Despite these advances, formal verification in these emerging domains faces notable hurdles, with scalability being among the most pressing. Neural networks employed in real-world implementations are often large and intricate, requiring verification techniques to balance computational efficiency and precision. Abstraction-refinement frameworks [45] and optimization-based approaches that exploit sparsity and neuron stability [66] offer promising pathways to address these challenges. At the same time, developing standardized benchmarks and tools, akin to VNN-LIB established for safety-critical domains [59], is crucial for fostering reproducibility and facilitating robust comparisons across different approaches.

By addressing scalability constraints, refining domain-specific specifications, and integrating interdisciplinary techniques, formal verification researchers are poised to unlock the full potential of these methods in ensuring safe, reliable, and ethical deployment of neural networks across a wide range of real-world scenarios. Building on prior

advancements in fairness and interpretability, these efforts promise to broaden the applicability of formal verification while responding to the unique demands and challenges presented by emerging application areas in the 21st century.

6 Challenges and Open Problems

6.1 Scalability Challenges in Modern Neural Network Architectures

Verifying modern neural network architectures, such as transformers, graph neural networks (GNNs), and recurrent neural networks (RNNs), poses significant scalability challenges due to their inherent complexity, size, and architectural nuances. These challenges stem from the exponential growth in computational demands as networks increase in depth, layer size, and structural interdependencies. This subsection examines the barriers to scalability in verifying these architectures, evaluates current progress, and identifies opportunities for future breakthroughs.

Transformers, widely used in tasks such as natural language processing and vision, represent a pinnacle of architectural complexity. Central to their design is the self-attention mechanism, which computes dependencies between each pair of input elements, inherently requiring $O(n^2)$ operations where n is the input sequence length. This quadratic complexity amplifies the difficulty of encoding verification constraints in satisfiability modulo theories (SMT) or mixed-integer programming (MIP) frameworks, especially in long-sequence tasks like text summarization or multi-modal inputs. Moreover, the dynamic nature of the attention mechanism introduces non-linear dependencies that are difficult to capture in traditional formal verification tools. Current SMT-based verifiers, such as Marabou [15], face bottlenecks when applied to transformers, limiting their applicability to small-scale or heavily simplified networks. While approximation methods marginally mitigate these barriers, they often compromise verification precision, leaving properties such as adversarial robustness or fairness underexplored.

Similarly, graph neural networks (GNNs) bring unique scalability challenges due to their reliance on graph-structured data. Properties such as node robustness under perturbations or sensitivity to adversarially modified edge connections require verification techniques capable of reasoning over graph topologies with potentially unbounded size. Unlike Euclidean data, graph-structured inputs introduce combinatorial dependencies among nodes and edges, exponentially increasing the complexity of reachability or safety verification. Standard piecewise-linear analysis approaches struggle with the non-Euclidean relational nature of GNNs, as noted in [7]. Although recent abstraction-refinement techniques have shown promise in verifying bounded-degree input graphs, nascent progress has not addressed scalability for larger or more expressive graph classes, such as those used in molecular drug design or large social network analysis.

Recurrent neural networks (RNNs) present verification challenges primarily due to the recursive nature of their computations. As RNNs process sequential data, their capacity to represent long-term dependencies results in a verification task that scales with the temporal depth of the input sequence. Properties such as output consistency and temporal safety are particularly challenging to verify, as the state transitions of these models encode non-deterministic behaviors unique to sequential processing. Furthermore, the recurrence introduces cyclic dependencies that standard reachability analysis fails to address effectively. Progress in invariant inference has allowed limited verification of RNNs [63], but existing methods struggle when applied to architectures like LSTMs or GRUs, which incorporate gating mechanisms that increase non-linearity and computational burden.

Another significant scalability barrier across architectures lies in the sheer size and depth of modern models. State-of-the-art networks such as GPT models or deep convolutional GNNs contain millions, if not billions, of parameters. Verification tools reliant on symbolic propagation or branch-and-bound frameworks [13] face exponential blow-ups in the computational graph when handling such networks, making even approximate verification infeasible for practical timescales. Relaxation-based strategies—e.g., interval bound propagation [48]—often result in significant over-approximations, leading to high false positive rates.

Efforts to address scalability typically involve either simplifying the verification task or improving computational efficiency. For instance, abstraction techniques such as DeepAbstract [21] achieve reductions in network size by clustering neurons with similar behaviors, albeit at the cost of over-approximation. Parallelization [62] has demonstrated practical gains, distributing computational workloads across hardware accelerators. Meanwhile, hybrid verification methods that combine numerical optimization with symbolic reasoning [12] have begun to explore trade-offs between precision and tractability, allowing partial scalability to larger networks.

Despite progress, much work remains to achieve scalable formal verification for all modern architectures. Future efforts may involve developing specialized verification tools for transformers and GNNs, leveraging domain-specific insights to reduce computational burdens without sacrificing rigor. Incorporating techniques like neuro-symbolic reasoning [19] or exploiting advances in quantum or edge computing [59] could also push the scale of verifiable models. Such interdisciplinary strategies will be critical to addressing the growing complexity of neural architectures in high-stakes applications.

6.2 Verification of Dynamic and Adaptive Systems

The formal verification of dynamic and adaptive neural networks, such as lifelong learning models and reinforcement learning (RL) policies, represents a critical yet underexplored frontier in the broader context of ensuring the reliability of neural networks. Unlike static models, these systems dynamically evolve over time, adapting to new inputs, tasks, or environmental conditions. While this adaptability enhances their utility in non-stationary and complex environments, it simultaneously introduces unique challenges to formal verification frameworks, which have been predominantly designed for static, pre-trained networks. This subsection delves into the specific obstacles posed by dynamic and adaptive systems, assesses recent progress, and explores avenues for future research.

Dynamic systems like lifelong learning models continuously incorporate new data while retaining prior knowledge—a feature essential for tasks where environmental conditions or objectives shift over time. Verifying such systems necessitates dynamic specification frameworks that evolve alongside the model's learning trajectory. Static, predefined specifications used effectively for traditional networks are inadequate to capture the temporal and evolving behaviors exhibited by lifelong learning models. As a result, novel verification paradigms, such as iterative or online verification frameworks, are needed to continuously monitor and ensure correctness throughout the model's evolution. Existing verification techniques often assume static architectures [13], [63], which results in substantial gaps when applied to systems that undergo adaptation. Hybrid methods integrating offline verification during training with runtime verification during deployment have demonstrated initial promise in addressing these challenges [75]. However, such approaches require further advancements to scale to complex, real-world applications while maintaining formal rigor.

Reinforcement learning (RL) policies present distinct challenges in their interaction with stochastic, high-dimensional environments. Verifying these policies involves an intricate analysis spanning the learned policy, state transitions, and the underlying reward functions. Traditional methods often focus on static analysis of policies without accounting for the dynamic and non-deterministic nature of RL environments. Techniques such as reachability analysis in closed-loop systems have emerged as promising tools to bridge this gap by capturing the interplay between RL policies and their operating environments, as demonstrated by frameworks like Verisig [28] and Reachability Analysis of Neural Network Control Systems [58]. Nevertheless, RL's characteristic long training horizons and the complexity of temporal safety properties, such as bounded control errors, significantly expand the state space, making comprehensive verification computationally prohibitive. Recent approaches leveraging Lipschitz continuity analysis [32] and probabilistic guarantees through conformal predictions [77] represent critical steps forward but remain constrained in scalability for large-scale or highly stochastic environments.

A growing area of interest lies in dynamic, evolving specifications, which adapt with the system's learning trajectory or task objectives. Temporal logics and scenario-based optimization provide a foundation for representing such specifications, but ensuring their scalability and compatibility with runtime constraints remains an open challenge. Runtime verification approaches, illustrated in tools like POLAR-Express [75], have gained traction by combining real-time monitoring with strategies like model switching to enforce safe behavior in changing environments. Despite these advances, refining and efficiently scaling dynamic specifications for neural systems deployed in real-world, safety-critical applications represents a critical and unresolved bottleneck.

Another fundamental concern in lifelong learning systems is catastrophic forgetting, where the model loses prior knowledge while adapting to new tasks. Verification in this context must extend beyond assessing transient correctness after adaptation to ensuring that previously validated knowledge remains intact. Techniques such as residual reasoning [61] and abstraction-refinement [21] offer promising pathways to address this issue. However, their utility is often constrained by the same scalability challenges faced by more conventional verification approaches, limiting their applicability to large-scale neural systems or rapidly evolving workloads.

Looking forward, the field of formal verification for dynamic and adaptive systems stands to benefit from modular and compositional methods. Rather than treating the system as a monolithic entity, divide-and-conquer strategies could verify individual components and synthesize global properties dynamically. Modular verification combined with counterexample-guided abstraction refinement [51] could enable more scalable and precise verification processes. Additionally, integrating machine learning techniques within the verification pipeline, as seen in predictor-verifier co-design frameworks [86], offers a novel opportunity to improve both efficiency and robustness. The adoption of neuro-symbolic methods for dynamically adapting and verifying high-level specifications [19] also provides a promising direction that merits further exploration.

In conclusion, the inherent complexity and evolving nature of dynamic and adaptive neural networks pose formidable challenges to traditional verification paradigms. Bridging the divide between verification and learning will require scalable frameworks capable of iteratively adapting to evolving models and stochastic environments. By advancing runtime monitoring, modular analysis techniques, and integration with neuro-symbolic reasoning, the field has an opportunity to extend verification capabilities to this crucial but underexplored frontier. These efforts will be pivotal in ensuring the safety and reliability of next-generation neural systems in dynamic, real-world deployments.

6.3 Stochastic and Probabilistic Behaviors in Neural Networks

The formal verification of neural networks in the presence of stochastic or probabilistic behaviors represents a significant challenge, given the increasing adoption of robust models like Bayesian Neural Networks (BNNs) and applications in noisy or uncertain environments. While deterministic neural networks have been the primary focus of verification efforts, real-world systems often operate under probabilistic constraints, ranging from model uncertainty to stochastic input distributions. This subsection explores these challenges, evaluates existing approaches, and identifies opportunities for future advancements in the rigorous verification of neural networks under stochastic settings.

Stochasticity in neural networks arises primarily from two sources: (1) intrinsic uncertainties in the model parameters, as seen in Bayesian Neural Networks, and (2) extrinsic uncertainties in the environment, such as noisy, adversarial, or incomplete input data. Bayesian Neural Networks explicitly model uncertainty by learning parameter distributions, providing probabilistic outputs rather than fixed predictions. While this property allows for calibrated decision-making under uncertainty, it complicates verification by introducing distributions over the output space. Techniques based on symbolic interval analysis or probabilistic abstraction over Bayesian models [34], [80] have

shown promise but remain computationally intensive for large-scale networks. Efficient abstractions, such as interval Markov chains [97], rely on simplifying the stochastic process while ensuring probabilistic soundness, offering partial solutions for specific stochastic systems.

Further challenges emerge when dealing with neural networks exposed to noisy or uncertain real-world inputs. For example, sensor noise, adversarial manipulation, and input variability introduce fundamental difficulties in verifying robustness. Interval arithmetic and symbolic interval methods, as employed in tools like ReluVal [37], provide some guarantees by constructing tight bounds on possible outputs under bounded noise. However, these methods often over-approximate distributions, compromising precision. Recent innovations in reachability analysis [58], [81] and Lipschitz-based optimization [58] address scalability but remain limited to deterministic approximations of stochastic phenomena.

The transition from deterministic guarantees to probabilistic ones introduces additional trade-offs. Probabilistic formal verification approaches often rely on bounding distributions of outputs or computing probabilistic guarantees for safety-critical properties. For example, abstractions grounded in convex optimization or Markov Decision Processes (MDPs) [80] construct finite probabilistic models that over-approximate neural network behaviors. Although these methods are sound with respect to probabilistic criteria (e.g., computing bounds on failure probabilities or deriving probabilistic safety envelopes for neural controllers), they impose strong assumptions about the network's smoothness or continuity, limiting their applicability to heavily nonlinear systems. Adaptive refinement techniques [97] have demonstrated scalability by iteratively refining abstraction granularity based on dynamic uncertainty, yet these methods still encounter difficulties when extended to high-dimensional input spaces or deep architectures.

Moreover, the interplay between noise and complex dynamics in systems such as hybrid neuro-symbolic architectures compounds verification complexity. Closed-loop systems often require joint validation of the neural network controller and its surrounding environment, necessitating hybrid stochastic frameworks. For instance, dynamic network models such as Neural Ordinary Differential Equations (Neural ODEs) embed stochasticity into their parameterized dynamics, complicating reachable set calculations [42]. Techniques leveraging polynomial approximations, zonotopes [41], and Lipschitz-based constraints [58] aim to mitigate over-approximations but require further unification across hybrid and stochastic settings.

Additionally, a critical challenge lies in balancing the expressive capacity of stochastic models with the computational tractability of verification algorithms. Models incorporating stochastic noise require probabilistic bounds for interpretability while respecting computational constraints. Algorithms such as those based on statistical conformal predictions [98] introduce confidence metrics to quantify and bound the likelihood of unsafe behavior in large-scale systems, but their reliance on approximations underscores the broader challenge of establishing both soundness and efficiency in verification pipelines.

Future research must address several pressing gaps. First, the development of more scalable probabilistic abstractions is vital for complex architectures like deep BNNs and systems with high-dimensional noise distributions. Second, hybrid frameworks are needed to bridge probabilistic and deterministic verification methods, enabling seamless integration of neural networks into broader cyber-physical systems. Third, benchmarking stochastic verification tools against real-world safety-critical scenarios remains essential for advancing this frontier. Progress in this domain could unlock more reliable guarantees for systems operating under uncertainty, spanning applications such as autonomous driving, medical diagnostics, and financial decision-making [34], [40].

In summary, formal verification of stochastic and probabilistic neural network behavior remains a formidable yet critical challenge. Bridging the gap between probabilistic model representations and verification frameworks involves not only theoretical advancements but also substantial innovations in computational techniques. By refining abstraction methods, integrating noise modeling into verification pipelines, and building stochastic verification standards, the community can move closer to achieving practical and reliable guarantees for neural networks in uncertain environments.

6.4 Integration into Machine Learning Pipelines

The integration of formal verification tools into the machine learning (ML) pipeline represents a critical yet underexplored challenge in advancing the reliability and accountability of neural networks. Despite substantial progress in designing algorithms to verify properties such as robustness, fairness, and interpretability, these efforts often remain disconnected from the iterative and high-throughput nature of real-world ML workflows. This subsection examines the complexities of embedding formal verification within the lifecycle of neural network design, training, and deployment, evaluating existing approaches, discussing limitations, and proposing future directions.

A compelling research focus lies in embedding verification constraints directly into the training process to produce "inherently verifiable" networks. Differentiable representations of verification objectives allow verifiable properties to be integrated as constraints or regularizations within the loss function. For example, enforcing ReLU stability during training can significantly reduce the computational overhead required for post-training verification, improving scalability [66]. Similarly, co-design frameworks that optimize for weight sparsity and neuron stability demonstrate a dual benefit of robustness and easier verifiability without significant trade-offs in model performance [66]. However, a persistent challenge across these approaches remains balancing generalizability with strict constraint satisfaction, underscoring the need for further refinement to align training objectives with post-training verification needs.

In the post-training phase, standalone verification tools are increasingly adopted but remain difficult to integrate smoothly into modern ML pipelines. Interoperability has been a key bottleneck due to the lack of standardized model specifications and formats. Recent initiatives like ONNX

and VNN-LIB standards have made strides towards resolving this, improving compatibility with ML frameworks such as TensorFlow, PyTorch, and JAX, but have yet to achieve widespread adoption [14]. Meanwhile, the high computational cost of verification continues to limit its applicability, particularly for large-scale networks or those with high-dimensional inputs. Techniques such as GPU-accelerated bound-propagation algorithms [46] and scalable polyhedral transformations [44] offer promising advances in computational efficiency. Nonetheless, further improvements are necessary to address the burgeoning model sizes and complexity typical of cutting-edge architectures.

The challenges do not end at post-training verification but extend into testing and certification processes prior to deployment. Tools like Marabou and ERAN have proven effective at verifying safety properties and identifying counterexamples, but fail to align seamlessly with iterative CI/CD pipelines used in many ML production workflows [2], [26]. Iterative model updates—characteristic of processes such as online learning or transfer learning—aggravate these shortcomings, given the exhaustive or inflexible nature of many verification tools. Incremental verification frameworks show promise in addressing this challenge by reusing intermediate verification artifacts from prior iterations, significantly improving efficiency. For example, recent methods have achieved 2.4×–3.8× speedups by leveraging intermediate results for updated models, underscoring their potential in deployment-centric scenarios [82].

A closely related challenge involves the interpretability of verification outputs, which is vital for fostering their adoption by practitioners. Many existing tools produce outputs that are mathematically rigorous but lack the actionable insights necessary for developers to apply verification results effectively. Integrating Explainable AI (XAI) techniques, such as feature attribution or relevance propagation, into verification processes could bridge this gap by grounding abstract results in accessible, domain-specific contexts. Emerging methods like neuro-symbolic verification combine logical specifications with neural representations, presenting a step forward, but their current implementations remain at an early stage, with limited practical applications [19].

Looking ahead, integrating verification-aware model design, scalable post-training verification, and deployment-standard practices into hybrid pipelines may offer a systematic pathway for embedding formal verification into ML workflows. Advances in hardware acceleration, exemplified by GPU-powered bound propagation [44] and exploratory efforts in quantum-inspired verification [99], also provide a key technological foundation for overcoming computational barriers. Furthermore, community-driven benchmarks such as VNN-COMP [59] continue to set the stage for measuring and propagating best practices across the field.

Despite these advances, several open problems remain. Establishing generalized APIs to enable seamless communication between verification tools and widely used ML frameworks will be essential for improving accessibility. Handling uncertainty in stochastic or evolving networks alongside scalability challenges posed by increasingly intricate architectures remains another critical area of focus. By addressing these gaps, embedding robust verification frameworks into ML pipelines can become a cornerstone for achieving trustworthy AI, driven by unified efforts across academia, industry, and standardization bodies.

6.5 Balancing Interpretability and Verification Precision

Balancing interpretability and verification precision in neural networks poses fundamental challenges, particularly due to the inherent complexity and opacity of modern deep learning systems. Interpretability focuses on rendering verification results accessible and comprehensible to practitioners, enabling actionable insights. Conversely, precision aims to provide accurate, tight guarantees on the satisfaction or violation of formal properties. However, achieving optimal outcomes in both dimensions is fraught with trade-offs, as improving one often compromises the other, especially in high-dimensional, large-scale networks.

One of the central tensions arises from over-approximations, which are commonly employed in scalable verification methods. Over-approximation ensures soundness, as it avoids missing potential violations of safety properties, but it frequently leads to imprecise bounds, causing false positives that undermine practical utility. Techniques such as symbolic interval analysis [37] and convex relaxations [58] exemplify this trade-off. While these methods reduce computational costs by abstracting parts of the model, they often overestimate feasible regions in the input or output space, creating verification results that are overly cautious and less actionable.

Furthermore, the trade-off between interpretability and precision becomes particularly stark in systems utilizing abstraction-based techniques. For instance, abstraction-refinement frameworks like Cnn-Abs [51] simplify complex convolutional networks by temporarily removing certain connections or layers during the analysis process. Although abstraction enables interpretability by reducing the scale of the verification problem and aligning with domain-specific intuition, the inherent over-approximation during abstraction often necessitates subsequent refinements. This iterative refinement not only impacts computational scalability but also risks introducing additional complexity that reduces the interpretability of intermediate results.

Emerging techniques, such as residual reasoning [61], highlight innovative approaches to partially mitigate this trade-off. By reusing verification insights from lower-fidelity abstractions during the refinement process, residual reasoning reduces redundant computations, improving both interpretability and precision. However, applying such approaches remains contextually dependent, as the suitability of residual reasoning techniques varies with the architecture and input-output relationships of the underlying neural network.

The field of explainable AI (XAI) provides another perspective on bridging the gap between interpretability and precision in verification. Formal XAI techniques, such as those discussed in [100], leverage interpretability frameworks to generate provably correct explanations for neural network behavior. These techniques align verification outputs with human-understandable features or explanations,

reducing the cognitive burden associated with comprehending abstract or mathematical verification results. However, integrating such methods often introduces computational overhead and risks a reduction in precision, particularly when heuristics are used to simplify the explanations. For example, the use of bundles to group related explanations in [100] improves interpretability but may omit rare edge cases critical for certain safety-critical applications.

Recent advancements in hybrid frameworks also hold promise for addressing this dual challenge. Techniques that combine symbolic and numerical methods, such as interval applications integrated into gradient-based optimization [101], demonstrate that tailored combinations of verification methods can improve precision without significantly compromising interpretability. These systems employ symbolic over-approximation for coarse reasoning followed by numerical optimization for tighter bounds. Still, seamless integration of such hybrid frameworks into verification pipelines remains an open problem, particularly in ensuring their accessibility for non-experts.

Another dimension of this trade-off is rooted in the complexity of network architectures. For example, piecewise linear networks using ReLU activations have well-studied verification tools due to their mathematical tractability [3]. However, contemporary architectures, such as transformers, graph neural networks, or spiking neural networks, present more intricate dependencies that challenge interpretability [5]. Precisely analyzing such models often requires complex specifications, limiting the accessibility and comprehensibility of verification outcomes.

Looking forward, addressing this trade-off will require interdisciplinary strategies. One promising direction is the integration of concept-based specifications, as explored in [93]. By translating verification results into familiar, high-level domain concepts, such approaches may simultaneously enhance interpretability while maintaining rigorous analysis. Additionally, incorporating user-guided refinement into verification pipelines, wherein practitioners interactively refine over-approximations and adjust interpretability expectations, could improve practical applications of verification tools.

In conclusion, the balance between interpretability and verification precision remains a critical open problem, necessitating innovations at the intersection of formal methods, machine learning, and explainable AI. Advances in residual reasoning, hybrid symbolic-numerical frameworks, and concept-based abstractions are paving the way for meaningful progress. However, achieving this balance in increasingly complex neural networks will require ongoing efforts to design scalable, user-friendly verification tools that address both conceptual clarity and mathematical rigor.

6.6 Formal Verification of Hybrid and Multi-Modal Systems

The formal verification of hybrid and multi-modal systems, where neural networks are integrated with other components or modalities, represents a critical frontier, raising unique technical challenges. These systems are increasingly deployed in high-stakes domains such as autonomous vehicles, robotics, and cyber-physical systems, where ensuring safety and robustness is paramount. This subsection delves into the specific challenges these systems pose and surveys existing methods and potential opportunities in this intricate verification landscape.

Hybrid systems consist of neural network controllers operating in conjunction with traditional dynamical systems, requiring verification approaches to bridge the discrete, piecewise-linear nature of neural networks and the continuous, often non-linear dynamics of the plant or environment. A prominent approach to address this challenge involves state-space abstraction to model the interactions between these components. For instance, Verisig [28] converts sigmoid-activated neural networks into equivalent hybrid automata, thereby enabling the application of traditional verification tools for analyzing closed-loop safety properties. While such methods demonstrate initial promise, they are often limited by their dependence on specific activation functions and architectures, excluding more widespread models such as those relying on ReLU activations. Furthermore, scalability remains a critical bottleneck, particularly for large-scale hybrid systems featuring high-dimensional input spaces.

Multi-modal systems, which integrate inputs from heterogeneous sources such as vision, language, and sensor data, introduce additional verification complexities. These systems require tools capable of analyzing interactions and dependencies among diverse modalities while managing the inherent trade-offs between precision and scalability. A commonly adopted but limiting approach involves simplifying assumptions, such as treating modalities independently or aggregating them into a unified format, which exacerbates over-approximations and undermines verification accuracy. Emerging techniques like abstraction-refinement frameworks offer a more nuanced approach by incrementally refining multi-modal dependencies, as exemplified in [45]. However, these methods remain computationally intensive and face challenges in scaling to deep multi-modal architectures. Moreover, specification difficulties increase when systems integrate symbolic and sub-symbolic components, as conventional input-output constraints fail to fully capture the semantic richness and cross-modal interactions.

A further layer of complexity arises from compositional reasoning, where the behavior of hybrid and multi-modal systems is influenced by interactions not only within the neural network but also between its outputs and other system components. While classical compositional reasoning techniques have progressed in non-neural contexts, their adaptation to systems combining neural networks with classical components is still in early stages. Promising approaches, such as those leveraging neural abstractions [53]—which approximate complex dynamics using bounded neural ODEs—provide a foundation for enabling compositional reasoning. Nonetheless, their applicability to multi-modal systems remains underexplored, pointing to a rich avenue for future research.

A fundamental obstacle to advancing the verification of hybrid and multi-modal systems is the lack of standardized interfaces between tools designed for verifying neural network components and those specialized in traditional dynamical systems. Incremental progress is being made through efforts to integrate neural reachability al-

gorithms, such as NNV [38], with classical verification platforms. While these integrated frameworks have demonstrated promising early successes in analyzing safety-critical closed-loop systems, significant work remains in refining interoperability to accommodate a wide range of hybrid configurations and architectures.

To move the field forward, several research directions can be prioritized. First, the development of scalable, closed-loop verification frameworks that effectively address the interplay between non-linear plant dynamics and neural network behaviors is essential. Techniques such as reachability analysis, as explored by [94], could be expanded to handle increasingly complex activations and dynamics. Second, enhancing the verification of dependencies in multi-modal systems requires harmonizing domain-specific abstractions with probabilistic reasoning, as demonstrated in some robustness verification methods for Bayesian neural networks [88]. This hybridization of symbolic and statistical reasoning provides a promising pathway to better capture the nuanced interactions between modalities. Finally, the establishment of benchmarking suites and community standards tailored to hybrid and multi-modal verification—building on efforts such as VNN-COMP [59]—can serve to unify research efforts and encourage reproducibility, while addressing the wide diversity of configurations in real-world applications.

In summary, despite meaningful advances in the verification of hybrid and multi-modal systems, significant challenges related to scalability, compositional reasoning, and tool integration persist. Progress in this domain will rely on developing innovative abstractions, advancing algorithmic interoperability, and fostering the standardization of methodologies. These endeavors are pivotal for building trustworthy, reliable neural network systems that can be deployed with confidence in high-stakes, safety-critical environments.

7 Emerging Trends and Future Directions

7.1 Verification-Driven Neural Architecture Design

Integrating verifiability into the design of neural network architectures represents a paradigm shift in artificial intelligence (AI) engineering. Traditional neural networks, optimized primarily for task-specific performance, pose significant challenges for formal verification due to their high complexity, non-linearities, and black-box nature. The emerging trend of verification-driven neural architecture design seeks to address these issues by prioritizing properties like simplicity, interpretability, and structure during the architecture development phase, ensuring systems are inherently more amenable to verification techniques.

One core strategy in verification-driven design is the use of piecewise-linear activation functions, such as Rectified Linear Units (ReLU), as they enable more tractable verification methods, including satisfiability modulo theory (SMT) solvers and mixed-integer linear programming (MIP) formulations. ReLU-based networks exhibit a piecewise-linear structure that allows formal verification algorithms to decompose the problem space into smaller, manageable components [3], [13]. However, as network sizes and architectural depths increase, the branching factor in verification grows exponentially, presenting scalability bottlenecks. This limitation motivates ongoing research into modified activation functions and sparse connections that inherently optimize architectures for verification [9].

Another promising approach is modular design, where neural networks are decomposed into smaller, independently verifiable sub-modules. Modular architectures make it easier to establish property-specific guarantees while simplifying the overall verification task. For instance, small feedforward blocks interfaced with interpretable symbolic layers facilitate both interpretability and property verification [19]. However, modularization must strike a balance between design flexibility and verification fidelity, as coarse-grained modular abstractions may introduce false positives or overly conservative bounds during verification.

A critical development in this domain is the integration of verifiability constraints within neural architecture search (NAS) frameworks. By embedding formal verification objectives, such as robustness or safety guarantees, into the NAS optimization process, it is possible to identify architectures that excel in both task performance and verifiability. Recent advancements employ symbolic reasoning techniques that evaluate a potential architecture's verification cost dynamically during the search phase [74]. Emerging tools like VerifAI incorporate simulation-based falsification and counterexample-guided architecture refinement, creating iterative feedback loops that improve both the design and verification simultaneously [20].

Sparse and low-precision networks, such as quantized and binarized neural networks, further exemplify architectures designed with verifiability in mind. Binarization reduces computational overhead, making SMT-based and MIP-based verification both faster and more scalable while maintaining adequate performance for specific tasks [18], [91]. Despite their reduced expressiveness, these architectures offer a compelling trade-off for applications requiring real-time performance and formal guarantees.

Another noteworthy direction focuses on designing architectures to ensure monotonicity, fairness, or safety constraints directly at the structural level. For example, networks constrained to enforce monotonicity in input-output relationships can simplify verification as specific properties can be encoded a priori rather than verified post hoc [79]. Similarly, explicit architectural constraints that preserve fairness or ethical considerations mitigate potential biases and ensure compliance with social and legal norms [102].

Even with these advancements, challenges remain, particularly scalability, robustness across varying tasks, and generality of verification guarantees. Novel computational frameworks aim to address these limitations by leveraging parallel processing and emerging hardware, such as GPUs or TPUs, to accelerate verification-aware training processes [2], [62]. Furthermore, combining verification-driven design with adaptive or hybrid learning frameworks, such as learning-enabled cyber-physical systems, promises to expand the impact of verifiability-first architectures beyond static decision-making domains into dynamic, real-world systems [38].

Looking ahead, advancing verification-driven neural architecture design will require stronger collaborations between formal methods, hardware optimization, and AI research fields. Establishing standardized metrics and bench-

marks for verifiability-aware architectures, alongside incentives for community-led innovation via competitions like VNN-COMP, could catalyze further transformative progress [24]. Ultimately, bridging performance efficiency with provable guarantees will cement verification-first design as a linchpin of safe and trustworthy AI deployment in safety-critical contexts.

7.2 Integration of Verification and Training Pipelines

The integration of verification techniques with training pipelines represents a pivotal advancement in the formal verification of neural networks, shifting the focus from post hoc validation to embedding formal guarantees within the learning process. This paradigm enhances trustworthiness, robustness, and scalability by circumventing limitations associated with decoupling training and verification stages. Recent progress has illuminated several promising approaches and frameworks that unify these traditionally separate tasks, enabling the design of models that are not merely accurate but inherently verifiable.

A key trend in this direction involves encoding verification properties directly into loss functions used during training. By leveraging differentiable constraints representing safety, robustness, or fairness properties, neural networks are trained to minimize violations of these properties while simultaneously optimizing task-specific objectives. For example, interval bound propagation (IBP) during training allows networks to adapt parameters throughout optimization to tighten robustness bounds [65]. This ensures provable robustness against adversarial perturbations once certified bounds are verified post-training. However, balancing verification-compatible training constraints with task performance remains challenging; excessively tight bounds, while enhancing verifiability, can degrade predictive accuracy.

Predictor-verifier frameworks offer another innovative pathway where a verifier co-trains with the primary predictor, dynamically ensuring compliance with formal specifications and iteratively refining the predictor's training trajectory [86]. This collaborative optimization strengthens the feedback loop between learning and verification, enabling significant reductions in training time and notable improvements in verified robustness, particularly on benchmarks like MNIST and SVHN [86]. However, scalability remains a limiting factor, as the computational overhead of co-training grows with model and dataset complexity, presenting obstacles to broader adoption in more intricate real-world scenarios.

A complementary strategy involves inducing structural properties during training that simplify subsequent verification processes. Techniques like enforcing ReLU stability—ensuring that activation patterns remain consistent across similar inputs—can drastically reduce the complexity of verification post hoc [66]. Similarly, promoting weight sparsity not only benefits verification scalability by limiting the search space but also improves computational efficiency [66]. Although these methods can streamline verification and enhance computational performance, imposing these structural constraints may curtail the expressivity of neural networks, potentially limiting their suitability for tasks requiring high complexity or nuance.

Emerging hybrid frameworks further bridge formal methods and training. For instance, neuro-symbolic verification techniques integrate logical specifications directly into training processes, accommodating complex real-world criteria such as fairness and safety [19]. Modular co-design strategies, where neural architectures are explicitly optimized for intrinsic verifiability, also offer exciting prospects. Enforcing low Lipschitz constants during training produces smoother networks that simultaneously exhibit improved verifiability and robustness to adversarial disturbances [85]. These strategies highlight a shift toward refining architectural design to balance expressiveness and verifiability, reflecting a broader trend to unify learning and verification.

Despite their potential, these integrated frameworks face notable challenges. Differentiable verification constraints or verifier models add substantial computational overhead, often requiring advanced hardware or approximate methods that risk compromising precision. Embedding guarantees during training can also limit a network's adaptability to new or unforeseen deployment scenarios, particularly in dynamic environments requiring evolving specifications. Furthermore, while most efforts have concentrated on robustness against adversarial examples, ensuring compliance with dynamic objectives—such as reinforcement learning policies or lifelong learning systems—remains an open research frontier.

Addressing these issues requires scalable and efficient optimization strategies that reduce computational demands without compromising verifiability. Distributed architectures and GPU acceleration, as evidenced in tools like GPUPoly [44], offer promising avenues to alleviate resource constraints. Similarly, conformal prediction frameworks could supplement these methods, enabling probabilistic certification of adherence to specifications within the training pipeline [77]. Such innovations could enhance the feasibility of these approaches for high-dimensional, real-world applications, broadening their impact across diverse domains.

By embedding verification principles into training, neural networks attain both task performance and provable adherence to safety, robustness, and fairness standards. This convergence is vital for deploying reliable, trustworthy machine learning systems, especially as neural networks increasingly underpin safety-critical applications. Integrated training-verification pipelines thus represent a crucial step toward fulfilling the promise of AI systems that are not only powerful but also demonstrably secure and reliable.

7.3 Expanding Verification to Emerging Neural Architectures

Emerging neural architectures such as spiking neural networks (SNNs), graph neural networks (GNNs), and transformers have introduced unique modeling paradigms aimed at solving specialized tasks across domains like neuroscience, social network analysis, and natural language processing. However, these architectures pose significant challenges to formal verification methodologies, necessitating novel advancements to ensure robustness, safety, and reliability in their deployment, particularly in high-stakes settings. This subsection explores the current landscape of

verification for these architectures, detailing advancements, challenges, and future directions.

Spiking neural networks represent a biologically inspired class of models that process information temporally, relying on spikes rather than continuous values for computation. Their inherent time-dependent dynamics and event-driven nature introduce complexities that extend beyond traditional architectures. Temporal properties critical to SNN behavior, such as spatiotemporal spike patterns and bounded delays, present unique verification problems demanding new specifications and methods. Standard reachability analyses designed for feedforward models are ill-suited to the non-linear and asynchronous behavior of SNNs. Techniques such as encoding the dynamics of SNNs as hybrid automata have shown potential [78], but scalability challenges remain due to the combinatorial growth of configurations in event-based scenarios. There is also an emerging interest in developing temporal logic frameworks specifically tailored for SNNs, such as Signal Temporal Logic (STL), to capture their rich temporal semantics [103]. These efforts aim to integrate time-sensitive analysis into verification pipelines while overcoming the limitations of traditional methods used for static architectures.

Graph neural networks, used to analyze graph-structured data, pose another verification challenge due to their reliance on message-passing mechanisms and aggregation functions, which induce a combinatorially rich input space of graphs with variable sizes and structures. Structural perturbations (e.g., node or edge manipulations) form an adversarial space that must be considered to ensure robustness and safety properties. Formal verification efforts for GNNs are increasingly leveraging symbolic and abstraction-based techniques to handle these topological complexities. For example, abstraction frameworks have been explored to model GNNs' operations over equivalence classes of graph structures, reducing the dimensionality of the verification task [69]. Advanced forms of reachability and sampling-based methods have further been proposed for verifying graph-based perturbations, balancing computational efficiency with precision. However, scalability beyond small and moderately sized graphs remains a challenge, particularly when verifying properties that capture global graph features, such as submodularity or connectivity constraints [87]. Hybrid approaches combining symbolic verification of message-passing functions with probabilistic bounds over structural modifications may emerge as a promising direction.

Transformers and large-scale language models, including large language models (LLMs), have become foundational architectures in natural language processing and are starting to play critical roles in domains requiring explainability and reliability. Transformers' self-attention mechanism introduces quadratic complexity with respect to input sequence length, complicating the verification of safety properties such as robustness to adversarial token perturbations or the monotonicity of attention weights. Current approaches to transformer verification include abstraction-based techniques that reduce the complexity of attention mechanisms while preserving verification fidelity [9]. Another promising development lies in symbolic interval analysis, where bounds over sequence semantics are derived to identify properties violation without exhaustively enumerating sequence combinations [37]. However, these methods are still limited in scaling to the massive parameter sizes and contextual interdependencies found in modern LLMs. Integrating these methods with efficient distributed computational frameworks may alleviate some of the scalability issues while ensuring sound analysis.

One common trend across all three architectures is the shift toward specification refinement and modular verification. Rather than treating each architecture monolithically, techniques that decompose the verification task into smaller, interpretable modules are gaining momentum. For instance, leveraging neuro-symbolic representations that fuse neural components with logical specifications has shown promise for both temporal SNN verification and task-driven transformer analysis [19]. Similarly, compositional reasoning frameworks are being developed to handle GNNs in isolation from their graph inputs, enabling scalable property verification using reduced-order models [69].

Despite these advances, significant gaps persist. Verification frameworks for emerging neural architectures often require domain-specific tools, which limits generalizability. Furthermore, as architectures like SNNs and transformers shift toward real-world implementation (e.g., in neuromorphic hardware or conversational AI systems), ensuring compliance with application-specific safety regulations remains underexplored. Future efforts should focus on the unification of verification approaches across architectures, the creation of standardized benchmarks (e.g., extending VNN-COMP benchmarks [59] to include these architectures), and leveraging advanced computational resources such as GPUs and TPUs for scalable analysis [102].

The expanding scope of neural verification into these architectures necessitates interdisciplinary collaborations across formal methods, machine learning, and application-specific domains. By integrating insights from each, the field can address its current challenges while enabling the safe and reliable deployment of next-generation neural models.

7.4 State-of-the-Art Computational Resources for Verification Scalability

The scalability of formal verification techniques for neural networks remains a critical bottleneck, especially as model architectures grow larger and more complex. Recent advances in computational resources and methodologies offer promising avenues to address these challenges, leveraging specialized hardware, parallelization strategies, and algorithmic refinements. This subsection explores the latest computational solutions that have enabled breakthroughs in scalability, evaluates their trade-offs, and highlights emerging opportunities for future development.

High-performance parallel processors, such as GPUs and TPUs, have been instrumental in improving the scalability of neural network verification. GPUs, with their inherent ability for massive parallelism, have accelerated key tasks like bound propagation and branch-and-bound (BaB) traversal [44], [46]. For instance, tools like GPUPoly utilize GPU-specific algorithms for polyhedral analysis, enabling robust verification of networks with up to 1 million neurons and 34 layers in approximately 34.5 ms [44]. Similarly,

α,β-CROWN employs GPU-optimized bound propagation in BaB frameworks, achieving neuron-split computations at significantly higher speeds, with up to three orders of magnitude improvement compared to traditional linear programming solvers [46].

Key advancements in computational efficiency also come from innovative relaxation techniques. Convex relaxations, which approximate non-linear activation functions using linear or convex forms, provide a scalable mechanism for verification. Techniques such as PRIMA leverage convex hull approximations to manage arbitrary activation functions while maintaining precision and scalability [67]. Further, sum-of-infeasibilities methods adapt stochastic optimization strategies, such as DeepSoI, to iteratively refine verification bounds, exploiting parallelism more effectively [104].

Algorithmic innovations, including the use of Lagrangian relaxation, have further contributed to the scalability of verification methods. These approaches tightly integrate problem structures into parallelizable computational graphs, enabling efficient bound computation without sacrificing precision [99]. Incorporating proximal algorithms into verification frameworks has also improved the equilibrium between computational overhead and accuracy. Insights from these techniques further enhance traditional constraint-solving methods within complete verification workflows [50].

In addition to hardware-centric approaches, structured model design offers a complementary path to achieve scalability. Verification-aided frameworks that promote neuron stability—such as sparsity-aware ReLU stability—reduce computational complexity during verification [66]. By optimizing model architectures during training, neural networks can be designed to inherently streamline verification processes, cultivating co-design principles between the development and verification phases.

Quantum computing presents an exciting frontier for tackling scalability barriers in verification, though it remains in its infancy. Early theoretical studies suggest that verification tasks involving Boolean satisfiability problems might benefit from exponential speedups using quantum approaches [62]. Hybrid hardware systems that integrate quantum processors with classical computational frameworks could provide tractable solutions to previously intractable verification problems.

The adoption of distributed systems further expands opportunities for scalability. Cloud-based verification solutions, such as those utilized in the International Verification of Neural Networks Competition (VNN-COMP), standardize resource allocation and tool evaluation under consistent computational settings [24]. However, ensuring fairness across distributed infrastructures, especially when hardware computational efficiencies vary, remains an unresolved issue.

Despite these advancements, challenges remain. Resource-intensive frameworks often trade off tightness in verification bounds to favor computational feasibility [4]. Furthermore, verification methods optimized for specific neural architectures, such as piecewise-linear models, face hurdles when generalizing to emerging architectures like Vision Transformers or spiking neural networks [89].

Future efforts must address these limitations by developing adaptive verification pipelines that dynamically scale with network size and complexity. Co-design strategies that integrate verifiability constraints into training, alongside abstraction frameworks generalized to diverse architectures, present promising directions. Continued advancements in GPU and TPU-optimized relaxations, in conjunction with the potential of quantum-enhanced algorithms, represent fertile ground for tackling scalability challenges.

The intersection of hardware innovation, algorithmic refinement, and interdisciplinary collaboration will shape the evolution of scalable verification. As neural networks continue to advance in complexity and application diversity, computational resources and methods must adapt to sustain formal verification as a critical tool for ensuring the reliability and safety of machine learning systems in high-stakes domains.

7.5 Community Standards and Benchmarking for Verification

With the increasing complexity and widespread adoption of neural networks (NNs) in safety-critical systems, establishing community standards and robust benchmarking practices for formal verification has become a cornerstone for advancing the field. Effective benchmarking supports the systematic evaluation of verification tools, fosters reproducibility, and provides meaningful comparisons across approaches. However, despite substantial progress in verification methods, the lack of standardized benchmarks, interoperable frameworks, and universally agreed-upon performance metrics continues to hinder the field's development and real-world applicability.

To address this challenge, community efforts in recent years have focused on developing universal benchmarks and shared datasets for verification tasks. Initiatives like VNN-COMP, the International Verification of Neural Networks Competition, have been particularly transformative. VNN-COMP 2022 and 2023 [14], [59] defined standardized network formats (e.g., ONNX) and formal specification libraries (e.g., VNN-LIB), creating a common ground for evaluating verification tools. These benchmarks often include tasks like robustness verification under adversarial perturbations and reachability analysis, enabling tool developers to assess their methods systematically across equal computing resources. Such competitions highlight the importance of unifying the research community through shared goals but also underscore the challenges of achieving scalability and verifiability for diverse networks and architectures.

The design of benchmarks poses several trade-offs. Standardized datasets, such as ACAS Xu [14], [38], are widely used to test the robustness and safety properties of neural networks but are limited in their representation of real-world scenarios. Although such datasets provide a controlled environment for performance measurement, expanding benchmarks to accommodate emerging architectures like transformers and graph neural networks (GNNs) remains an open problem [7]. The balance between representativeness, scalability, and accessibility is crucial for designing benchmarks that reflect practical constraints and challenges.

Equally important is the standardization of specification formats, which has been catalyzed by the adoption of frameworks like VNN-LIB. These formats enable researchers to express diverse properties, such as safety, robustness, and fairness, in a machine-interpretable manner. Tools such as Reluplex [3] benefit from integrating these specifications to automate verification tasks. However, standardized frameworks alone are insufficient unless accompanied by intuitive interfaces and tools that seamlessly integrate into machine learning pipelines. The ONNX ecosystem provides a promising foundation for bridging this gap through interoperability with popular deep learning frameworks, facilitating both benchmarking and real-world adoption [14], [87].

The quest for reproducibility further necessitates the open sourcing of verification tools and datasets. Platforms like Marabou [15] and DeepNNC [58] offer valuable resources for researchers while exemplifying how open-source implementations can drive innovation and collaboration. Open repositories lower the barrier to entry for new researchers, enabling them to replicate experiments and build upon prior work. However, significant challenges remain in ensuring comparability, especially in the absence of universally agreed evaluation metrics like precision, scalability, and computational efficiency, which vary widely across different tools and applications.

As the field matures, community-driven initiatives must address pressing issues, such as accommodating multi-modal tasks and hybrid systems within benchmarking frameworks [41], [60]. There is also a growing need to evaluate ethical and fairness guarantees alongside traditional robustness and correctness properties [87], [100]. These requirements demand a concerted effort to expand benchmarks and specifications to reflect diverse applications and societal considerations.

Looking ahead, the development of adaptive benchmarks capable of evolving alongside advances in neural network architectures and application domains is essential. Drawing insights from competitions like VNN-COMP and open tools such as NNV [38], future efforts should focus on building modular, extensible benchmarking pipelines that ensure compatibility while minimizing overhead. Establishing cross-disciplinary collaborations, integrating advanced computational frameworks like quantum computing [12], and promoting inclusive participation from academia and industry are pivotal to advancing this domain. The roadmap for verification standards and benchmarking hinges on fostering global standards that blend rigor, flexibility, and accessibility to empower foundational breakthroughs in formal verification research.

7.6 Towards Verification of Ethical and Fair AI Systems

The formal verification of ethical and fair AI systems represents a critical frontier in the domain of neural network verification, aligning closely with broader goals of ensuring safety, robustness, and societal alignment in machine learning applications. As machine learning models increasingly mediate high-stakes decision-making across diverse domains—such as employment, healthcare, and judicial systems—their potential to propagate social biases or violate ethical norms underscores the urgent need for rigorous verification frameworks that prioritize fairness. This subsection explores the intersection of formal verification and fairness, highlighting existing methodologies, their inherent limitations, and emerging trends that address these pressing challenges.

Formal verification in fairness primarily involves certifying that neural networks satisfy explicitly defined fairness criteria, such as demographic parity, equalized odds, or individual fairness, under rigorous mathematical guarantees. These properties, often expressed as statistical inequalities or logic-based constraints, aim to ensure that sensitive attributes, such as race or gender, do not unduly influence outcomes. Quantitative verification frameworks have been leveraged to probabilistically verify compliance with such fairness properties, offering bounded error guarantees via methods such as PAC (Probably Approximately Correct) analysis. For example, one prominent approach [105] provides a structured mechanism to estimate the proportion of inputs satisfying predefined fairness conditions, creating a foundation for operationalizing fairness verification across a variety of neural network models and demographic scenarios.

A central challenge in fairness verification arises from a triad of competing priorities: fairness, accuracy, and robustness. Efforts to optimize fairness often result in performance trade-offs, particularly in scenarios where fairness constraints conflict with the inherent data distributions. Hybrid frameworks have emerged to address these tensions by uniting fairness optimization with verifiability during the training process. For instance, the predictor-verifier co-training paradigm [86] integrates fairness constraints directly into the learning phase, simultaneously training a neural network to prioritize fairness objectives while ensuring post-training verifiability. This paradigm exemplifies how fairness-aware training and formal verification can collaborate to bridge the gap between model development and ethical accountability.

The intersection of fairness verification and ethical imperatives accentuates the importance of transparency and interpretability in verification outputs. Just as explainability is vital in traditional AI applications, fairness guarantees must be accessible to both technical and non-technical stakeholders. Emerging methods aim to generate interpretable certificates of fairness that not only validate compliance but also pinpoint causal factors underlying potential bias [19]. Techniques rooted in neuro-symbolic reasoning show promise in encoding complex fairness and ethical constraints, enabling the articulation of multifaceted conditions that go beyond simple statistical metrics, particularly in addressing nuanced real-world ethical dilemmas.

Despite these advances, the field continues to grapple with significant computational challenges. The increasing complexity of neural networks, coupled with fairness conditions involving high-dimensional demographic attributes, amplifies the difficulty of scalability. Advances in verification techniques, such as mixed-integer programming [13] and symbolic constraint propagation [26], offer scalable solutions for specific fairness cases. However, verifying nonlinear and probabilistic models remains challenging due to the reliance on coarse over-approximations to maintain

computational feasibility.

An important evolution in fairness verification lies in its extension to multi-modal and compositional systems. Modern AI pipelines often integrate neural networks with traditional algorithms and other decision-making processes, necessitating fairness verification at the system level. Hybrid frameworks that analyze the interactions of symbolic and neural components, such as those employed in the formal analysis of neuro-symbolic systems [83], provide promising methodologies for tackling system-wide fairness. Such approaches account for the complexity of real-world AI systems, further advancing the capabilities of fairness verification.

Looking ahead, the formal verification of fairness and ethics in AI systems demands a multidisciplinary approach. Collaborations with sociologists and ethicists are essential to define fairness criteria that go beyond rigid statistical constraints and incorporate broader societal insights. To address scalability challenges, leveraging distributed systems and advanced computational resources, such as GPUs, is paramount [44]. Additionally, the development of benchmarking initiatives—similar to the role played by VNN-COMP in robustness verification—specific to fairness verification would help standardize methodologies, foster reproducibility, and catalyze development in this domain.

In conclusion, the formal verification of ethical and fair AI systems is an essential step toward building equitable and socially-aligned AI solutions. While advances in fairness guarantees demonstrate the feasibility of incorporating ethical considerations into AI development, challenges surrounding scalability, interpretability, and the inclusion of nuanced ethical principles remain significant. Addressing these challenges will not only ensure the reliability and accountability of AI systems but also mitigate the societal risks posed by their widespread deployment in critical decision-making contexts.

8 Conclusion

This survey has provided a comprehensive examination of the field of formal verification for neural networks, underscoring its crucial role in ensuring the reliability, safety, and trustworthiness of neural systems across diverse application domains. Formal verification serves as a mathematical and computational pillar for deriving guarantees on critical properties such as safety, robustness, fairness, and interpretability. Through the meticulous synthesis of foundational methods, emerging techniques, application-driven challenges, and future opportunities, this work bridges the technical advancements with their broader implications.

A key takeaway is the diversity in verification methodologies and tools, each addressing distinct challenges posed by neural network architecture and application settings. Symbolic methods, such as satisfiability modulo theories (SMT)-based solvers, provide rigorous proofs for properties but often face scalability challenges for deep or complex architectures [3], [91]. Numerical optimization techniques, including mixed-integer programming (MIP) and convex relaxation, have enabled the verification of larger networks and more nuanced properties, though often with trade-offs in precision [10], [11]. Abstract interpretation and over-approximation methods strike a balance between computational efficiency and soundness but may suffer from conservative bounds [48], [57]. Hybrid verification frameworks, which combine symbolic, numerical, and abstraction strategies, have emerged as promising solutions to address the scalability bottleneck without compromising rigor [13], [74].

Despite these advances, several limitations persist. Scalability remains a critical barrier, particularly for state-of-the-art architectures such as transformers, graph neural networks (GNNs), and Bayesian neural networks (BNNs), which exhibit complex dynamics and high-dimensional input spaces. Recent work has demonstrated that even under optimized conditions, relaxation methods face inherent tightness gaps for certain architectures [4], and the verification of stochastic systems, such as BNNs, poses additional challenges due to polynomial non-linearities [88]. Moreover, integrating verification techniques into dynamic and adaptive learning pipelines presents unresolved questions, requiring methodologies that can accommodate continual learning, evolving specifications, and real-time constraints [64].

Applications of formal verification in safety-critical contexts such as autonomous systems, healthcare, and finance reflect its transformative potential. Tools like Marabou and NNV demonstrate the ability to verify robustness and safety in autonomous vehicle navigation or collision-avoidance systems [2], [38]. Similarly, advancements in fairness verification indicate progress toward ensuring ethical AI systems in domains like recruitment and judicial decision-making [5]. However, the maturity of these tools falls short of widespread adoption, particularly when robustness and interpretability need to be balanced against computational overheads [79].

Emerging trends highlight exciting possibilities for advancing the domain. Verification-aware neural network design, which incorporates guarantees at the architectural level, is poised to revolutionize the field, making models inherently more amenable to analysis [21], [84]. Likewise, the integration of explainability and verification opens the door to providing intuitive, actionable insights alongside rigorous guarantees [19], [22]. Scalable computation frameworks, leveraging GPUs, quantum accelerators, or distributed architectures, represent another frontier for overcoming the scalability challenge [74].

To propel formal verification forward, collaborative efforts are essential. Standardization initiatives like VNN-LIB and competitions such as VNN-COMP have provided fertile ground for method comparison and tool development [24], [59]. However, more robust benchmarks, reflective of real-world complexity, are necessary to foster meaningful progress and adoption.

In conclusion, the formal verification of neural networks remains a dynamic and interdisciplinary field, straddling theoretical depth, practical innovation, and ethical imperative. Addressing existing challenges in scalability, integration, and interpretability will unlock new dimensions in safe and responsible AI deployment. By converging robust methodologies, computational advancements, and domain-specific collaborations, formal verification can realize its full potential in underpinning the trustworthiness of tomor-

row's AI systems.

REFERENCES

[1] X. Huang, D. Kroening, W. Ruan, J. Sharp, Y. Sun, E. Thamo, M. Wu, and X. Yi, "A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability," *Comput. Sci. Rev.*, vol. 37, p. 100270, 2018. 1

[2] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Efficient formal safety analysis of neural networks," *ArXiv*, vol. abs/1809.08098, 2018. 1, 2, 4, 6, 8, 10, 15, 19, 23, 25, 30

[3] R. Ehlers, "Formal verification of piece-wise linear feed-forward neural networks," in *Automated Technology for Verification and Analysis*, 2017, pp. 269–286. 1, 2, 4, 5, 6, 7, 10, 13, 15, 24, 25, 29, 30

[4] H. Salman, G. Yang, H. Zhang, C.-J. Hsieh, and P. Zhang, "A convex relaxation barrier to tight robustness verification of neural networks," in *Neural Information Processing Systems*, 2019, pp. 9832–9842. 1, 2, 3, 6, 7, 10, 16, 17, 28, 30

[5] M. H. Meng, G. Bai, S. Teo, Z. Hou, Y. Xiao, Y. Lin, and J. Dong, "Adversarial robustness of deep neural networks: A survey from a formal verification perspective," *ArXiv*, vol. abs/2206.12227, 2022. 1, 5, 24, 30

[6] T. Zhao, E. Yurtsever, J. Paulson, and G. Rizzoni, "Formal certification methods for automated vehicle safety assessment," *IEEE Transactions on Intelligent Vehicles*, vol. 8, pp. 232–249, 2022. 1, 2

[7] M. Salzer and M. Lange, "Fundamental limits in formal verification of message-passing neural networks," 2022. 1, 9, 13, 15, 20, 28

[8] S. Chen, L. Molu, and M. Fazlyab, "Verification-aided learning of neural network barrier functions with termination guarantees," *2024 American Control Conference (ACC)*, pp. 3610–3617, 2024. 1, 6, 10

[9] R. Bunel, I. Turkaslan, P. H. S. Torr, P. Kohli, and P. Mudigonda, "A unified view of piecewise linear neural network verification," in *Neural Information Processing Systems*, 2017, pp. 4795–4804. 1, 4, 6, 9, 25, 27

[10] C.-H. Cheng, G. Nührenberg, and H. Ruess, "Maximum resilience of artificial neural networks," *ArXiv*, vol. abs/1705.01040, 2017. 2, 15, 30

[11] R. Brown, E. Schmerling, N. Azizan, and M. Pavone, "A unified view of sdp-based neural network verification through completely positive programming," in *International Conference on Artificial Intelligence and Statistics*, 2022, pp. 9334–9355. 2, 30

[12] K. Dvijotham, R. Stanforth, S. Gowal, T. A. Mann, and P. Kohli, "A dual approach to scalable verification of deep networks," in *Conference on Uncertainty in Artificial Intelligence*, 2018, pp. 550–559. 2, 5, 7, 15, 17, 20, 29

[13] R. Bunel, J. Lu, I. Turkaslan, P. H. S. Torr, P. Kohli, and M. P. Kumar, "Branch and bound for piecewise linear neural network verification," *ArXiv*, vol. abs/1909.06588, 2019. 2, 3, 5, 6, 7, 9, 10, 11, 13, 15, 16, 19, 20, 21, 25, 29, 30

[14] M. N. Müller, C. Brix, S. Bak, C. Liu, and T. T. Johnson, "The third international verification of neural networks competition (vnncomp 2022): Summary and results," *ArXiv*, vol. abs/2212.10376, 2022. 2, 3, 4, 12, 13, 18, 23, 28, 29

[15] H. Wu, O. Isac, A. Zelji'c, T. Tagomori, M. Daggitt, W. Kokke, I. Refaeli, G. Amir, K. Julian, S. Bassan, P. Huang, O. Lahav, M. Wu, M. Zhang, E. Komendantskaya, G. Katz, and C. W. Barrett, "Marabou 2.0: A versatile formal analyzer of neural networks," *ArXiv*, vol. abs/2401.14461, 2024. 2, 6, 9, 10, 12, 13, 14, 15, 18, 20, 29

[16] D. Corsi, G. Amir, A. Rodríguez, C. Sánchez, G. Katz, and R. Fox, "Verification-guided shielding for deep reinforcement learning," *ArXiv*, vol. abs/2406.06507, 2024. 2

[17] A. Kumar, "Pragmatic formal verification of sequential error detection and correction codes (eccs) used in safety-critical design," *ArXiv*, vol. abs/2404.18270, 2024. 2

[18] C. Lazarus and M. J. Kochenderfer, "A mixed integer programming approach for verifying properties of binarized neural networks," *ArXiv*, vol. abs/2203.07078, 2022. 2, 25

[19] X. Xie, K. Kersting, and D. Neider, "Neuro-symbolic verification of deep neural networks," *ArXiv*, vol. abs/2203.00938, 2022. 2, 3, 13, 14, 17, 20, 21, 23, 25, 26, 27, 29, 30

[20] D. J. Fremont, J. Chiu, D. Margineantu, D. Osipychev, and S. Seshia, "Formal analysis and redesign of a neural network-based aircraft taxiing system with verifai," *Computer Aided Verification*, vol. 12224, pp. 122 – 134, 2020. 2, 15, 25

[21] P. Ashok, V. Hashemi, J. Křetínský, and S. Mohr, "Deepabstract: Neural network abstraction for accelerating verification," in *Automated Technology for Verification and Analysis*, 2020, pp. 92–107. 2, 9, 10, 11, 13, 18, 20, 21, 30

[22] J. Marques-Silva, "Disproving xai myths with formal methods – initial results," *2023 27th International Conference on Engineering of Complex Computer Systems (ICECCS)*, pp. 12–21, 2023. 2, 30

[23] O. Isac, C. W. Barrett, M. Zhang, and G. Katz, "Neural network verification with proof production," *2022 Formal Methods in Computer-Aided Design (FMCAD)*, pp. 38–48, 2022. 2, 11, 14

[24] C. Brix, M. N. Muller, S. Bak, T. T. Johnson, and C. Liu, "First three years of the international verification of neural networks competition (vnn-comp)," *International Journal on Software Tools for Technology Transfer*, vol. 25, pp. 329–339, 2023. 2, 10, 14, 26, 28, 30

[25] X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, "Safety verification of deep neural networks," in *International Conference on Computer Aided Verification*, 2016, pp. 3–29. 3, 11

[26] G. Katz, C. W. Barrett, D. Dill, K. D. Julian, and M. J. Kochenderfer, "Reluplex: An efficient smt solver for verifying deep neural networks," *ArXiv*, vol. abs/1702.01135, 2017. 3, 11, 12, 23, 29

[27] V. Tjeng, K. Y. Xiao, and R. Tedrake, "Evaluating robustness of neural networks with mixed integer programming," in *International Conference on Learning Representations*, 2017. 3, 7, 11, 16, 17, 19

[28] R. Ivanov, J. Weimer, R. Alur, G. Pappas, and I. Lee, "Verisig: verifying safety properties of hybrid systems with neural network controllers," *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, 2018. 3, 6, 11, 12, 13, 16, 19, 21, 24

[29] S. Biswas and H. Rajan, "Fairify: Fairness verification of neural networks," *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pp. 1546–1558, 2022. 3, 16

[30] H. Khedr and Y. Shoukry, "Certifair: A framework for certified global fairness of neural networks," in *AAAI Conference on Artificial Intelligence*, 2022, pp. 8237–8245. 3, 16

[31] H.-D. Tran, S. Bak, W. Xiang, and T. T. Johnson, "Verification of deep convolutional neural networks using imagestars," *Computer Aided Verification*, vol. 12224, pp. 18 – 42, 2020. 3, 11

[32] M. Fazlyab, M. Morari, and G. Pappas, "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming," *IEEE Transactions on Automatic Control*, vol. 67, pp. 1–15, 2019. 3, 7, 21

[33] W. Xiang, H.-D. Tran, and T. T. Johnson, "Reachable set computation and safety verification for neural networks with relu activations," *ArXiv*, vol. abs/1712.08163, 2017. 4, 5, 8, 13, 17

[34] A. Venzke and S. Chatzivasileiadis, "Verification of neural network behaviour: Formal guarantees for power system applications," *IEEE Transactions on Smart Grid*, vol. 12, pp. 383–397, 2019. 4, 21, 22

[35] V. Magron, G. Constantinides, and A. F. Donaldson, "Certified roundoff error bounds using semidefinite programming," *ACM Transactions on Mathematical Software (TOMS)*, vol. 43, pp. 1 – 31, 2015. 4

[36] C. Qin, K. Dvijotham, B. O'Donoghue, R. Bunel, R. Stanforth, S. Gowal, J. Uesato, G. Swirszcz, and P. Kohli, "Verification of non-linear specifications for neural networks," *ArXiv*, vol. abs/1902.09592, 2019. 4

[37] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana, "Formal security analysis of neural networks using symbolic intervals," in *USENIX Security Symposium*, 2018, pp. 1599–1614. 4, 8, 9, 12, 16, 18, 22, 23, 27

[38] 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," *Computer Aided Verification*, vol. 12224, pp. 3 – 17, 2020. 4, 8, 11, 13, 15, 16, 25, 28, 29, 30

[39] X. Yang, H.-D. Tran, W. Xiang, and T. Johnson, "Reachability analysis for feed-forward neural networks using face lattices," *ArXiv*, vol. abs/2003.01226, 2020. 4, 12

[40] Y. Zhang and X. Xu, "Safety verification of neural feedback systems based on constrained zonotopes," *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 2737–2744, 2022. 4, 22

[41] N. Kochdumper, C. Schilling, M. Althoff, and S. Bak, "Open- and closed-loop neural network verification using polynomial zonotopes," *ArXiv*, vol. abs/2207.02715, 2022. 4, 5, 13, 17, 22, 29

[42] D. M. Lopez, P. Musau, N. P. Hamilton, and T. T. Johnson, "Reachability analysis of a general class of neural ordinary differential equations," in *International Conference on Formal Modeling and Analysis of Timed Systems*, 2022, pp. 258–277. 4, 10, 22

[43] W. Merrill, "Sequential neural networks as automata," *ArXiv*, vol. abs/1906.01615, 2019. 4

[44] C. Müller, F. Serre, G. Singh, M. Püschel, and M. T. Vechev, "Scaling polyhedral neural network verification on gpus," in *Conference on Machine Learning and Systems*, 2020. 4, 5, 7, 8, 9, 11, 13, 14, 16, 18, 19, 23, 26, 27, 30

[45] J. Liu, Y. Xing, X. Shi, F. Song, Z. Xu, and Z. Ming, "Abstraction and refinement: Towards scalable and exact verification of neural networks," *ACM Transactions on Software Engineering and Methodology*, vol. 33, pp. 1 – 35, 2022. 4, 14, 19, 24

[46] 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 verification," *ArXiv*, vol. abs/2103.06624, 2021. 4, 9, 13, 17, 23, 27, 28

[47] C. Tjandraatmadja, R. Anderson, J. Huchette, W. Ma, K. Patel, and J. Vielma, "The convex relaxation barrier, revisited: Tightened single-neuron relaxations for neural network verification," *ArXiv*, vol. abs/2006.14076, 2020. 5

[48] W. Ruan, X. Huang, and M. Kwiatkowska, "Reachability analysis of deep neural networks with provable guarantees," in *International Joint Conference on Artificial Intelligence*, 2018, pp. 2651–2659. 5, 6, 8, 9, 19, 20, 30

[49] S. Kotha, C. Brix, Z. Kolter, K. Dvijotham, and H. Zhang, "Provably bounding neural network preimages," *ArXiv*, vol. abs/2302.01404, 2023. 5, 10

[50] A. Palma, R. Bunel, A. Desmaison, K. Dvijotham, P. Kohli, P. H. S. Torr, and M. P. Kumar, "Improved branch and bound for neural network verification via lagrangian decomposition," *ArXiv*, vol. abs/2104.06718, 2021. 5, 9, 28

[51] M. Ostrovsky, C. W. Barrett, and G. Katz, "An abstraction-refinement approach to verifying convolutional neural networks," *ArXiv*, vol. abs/2201.01978, 2022. 5, 11, 14, 15, 21, 23

[52] K. Jia and M. Rinard, "Efficient exact verification of binarized neural networks," *ArXiv*, vol. abs/2005.03597, 2020. 5, 11

[53] A. Abate, A. Edwards, and M. Giacobbe, "Neural abstractions," *ArXiv*, vol. abs/2301.11683, 2023. 5, 8, 24

[54] P. Yang, J. Liu, J. Li, L. Chen, and X. Huang, "Analyzing deep neural networks with symbolic propagation: Towards higher precision and faster verification," in *Sensors Applications Symposium*, 2019, pp. 296–319. 5, 18

[55] Z. Zhang, C. Ma, S. Soudijani, and S. Soudjani, "Formal verification of unknown stochastic systems via non-parametric estimation," *ArXiv*, vol. abs/2403.05350, 2024. 5, 8

[56] S. Webb, T. Rainforth, Y. Teh, and M. P. Kumar, "A statistical approach to assessing neural network robustness," *ArXiv*, vol. abs/1811.07209, 2018. 5, 14

[57] P. Prabhakar and Z. R. Afzal, "Abstraction based output range analysis for neural networks," *ArXiv*, vol. abs/2007.09527, 2020. 5, 6, 30

[58] C. Zhang, W. Ruan, and P. Xu, "Reachability analysis of neural network control systems," *ArXiv*, vol. abs/2301.12100, 2023. 6, 11, 13, 21, 22, 23, 29

[59] C. Brix, S. Bak, C. Liu, and T. T. Johnson, "The fourth international verification of neural networks competition (vnn-comp 2023): Summary and results," *ArXiv*, vol. abs/2312.16760, 2023. 6, 10, 11, 12, 14, 15, 19, 20, 23, 25, 27, 28, 30

[60] C. Schilling, M. Forets, and S. Guadalupe, "Verification of neural-network control systems by integrating taylor models and zonotopes," in *AAAI Conference on Artificial Intelligence*, 2021, pp. 8169–8177. 6, 8, 12, 29

[61] Y. Elboher, E. Cohen, and G. Katz, "Neural network verification using residual reasoning," in *IEEE International Conference on Software Engineering and Formal Methods*, 2022, pp. 173–189. 6, 12, 18, 21, 23

[62] H. Wu, A. Ozdemir, A. Zeljić, A. Irfan, K. D. Julian, D. Gopinath, S. Fouladi, G. Katz, C. Păsăreanu, and C. W. Barrett, "Parallelization techniques for verifying neural networks," *2020 Formal Methods in Computer Aided Design (FMCAD)*, pp. 128–137, 2020. 6, 14, 20, 25, 28

[63] Y. Jacoby, C. W. Barrett, and G. Katz, "Verifying recurrent neural networks using invariant inference," in *Automated Technology for Verification and Analysis*, 2020, pp. 57–74. 6, 11, 20, 21

[64] G. Amir, M. Schapira, and G. Katz, "Towards scalable verification of deep reinforcement learning," *2021 Formal Methods in Computer Aided Design (FMCAD)*, pp. 193–203, 2021. 6, 30

[65] S. Gowal, K. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelović, T. A. Mann, and P. Kohli, "On the effectiveness of interval bound propagation for training verifiably robust models," *ArXiv*, vol. abs/1810.12715, 2018. 7, 16, 26

[66] K. Y. Xiao, V. Tjeng, N. M. M. Shafiullah, and A. Madry, "Training for faster adversarial robustness verification via inducing relu stability," *ArXiv*, vol. abs/1809.03008, 2018. 7, 12, 14, 17, 19, 22, 26, 28

[67] M. N. Müller, G. Makarchuk, and G. Singh, "Prima: general and precise neural network certification via scalable convex hull approximations," *Proceedings of the ACM on Programming Languages*, vol. 6, pp. 1 – 33, 2021. 7, 9, 11, 17, 28

[68] Z. Shi, H. Zhang, K.-W. Chang, M. Huang, and C.-J. Hsieh, "Robustness verification for transformers," *ArXiv*, vol. abs/2002.06622, 2020. 7, 11, 16

[69] J. Törnblom and S. Nadjm-Tehrani, "Formal verification of input-output mappings of tree ensembles," *ArXiv*, vol. abs/1905.04194, 2019. 8, 27

[70] A. Clavière, E. Asselin, C. Garion, and C. Pagetti, "Safety verification of neural network controlled systems," *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pp. 47–54, 2020. 8

[71] C. Huang, J. Fan, X. Chen, W. Li, and Q. Zhu, "Polar: A polynomial arithmetic framework for verifying neural-network controlled systems," in *Automated Technology for Verification and Analysis*, 2021, pp. 414–430. 9, 13

[72] Y. Zhang, Z. Zhao, F. Song, M. Zhang, T. Chen, and J. Sun, "Qvip: An ilp-based formal verification approach for quantized neural networks," *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, 2022. 10

[73] G. Anderson, S. Pailoor, I. Dillig, and S. Chaudhuri, "Optimization and abstraction: a synergistic approach for analyzing neural network robustness," *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2019. 10, 15, 18

[74] 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," *ArXiv*, vol. abs/2011.13824, 2020. 10, 15, 25, 30

[75] W. Xiang and Z. Shao, "Safety verification of neural network control systems using guaranteed neural network model reduction," *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 1521–1526, 2022. 11, 21

[76] J. A. Vincent and M. Schwager, "Reachable polyhedral marching (rpm): A safety verification algorithm for robotic systems with deep neural network components," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9029–9035, 2020. 11

[77] A. Lin and S. Bansal, "Verification of neural reachable tubes via scenario optimization and conformal prediction," *ArXiv*, vol. abs/2312.08604, 2023. 11, 16, 21, 26

[78] W. Xiang and T. T. Johnson, "Reachability analysis and safety verification for neural network control systems," *ArXiv*, vol. abs/1805.09944, 2018. 12, 27

[79] W. Xiang, P. Musau, A. Wild, D. M. Lopez, N. P. Hamilton, X. Yang, J. A. Rosenfeld, and T. T. Johnson, "Verification for machine learning, autonomy, and neural networks survey," *ArXiv*, vol. abs/1810.01989, 2018. 12, 17, 19, 25, 30

[80] S. Adams, M. Lahijanian, and L. Laurenti, "Formal control synthesis for stochastic neural network dynamic models," *IEEE Control Systems Letters*, vol. 6, pp. 2858–2863, 2022. 12, 21, 22

[81] 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," *2018 Annual American Control Conference (ACC)*, pp. 1574–1579, 2018. 12, 22

[82] S. Ugare, D. Banerjee, S. Misailovic, and G. Singh, "Incremental verification of neural networks," *Proceedings of the ACM on Programming Languages*, vol. 7, pp. 1920 – 1945, 2023. 13, 23

[83] M. Daggitt, W. Kokke, R. Atkey, N. Slusarz, L. Arnaboldi, and E. Komendantskaya, "Vehicle: Bridging the embedding gap

in the verification of neuro-symbolic programs," *ArXiv*, vol. abs/2401.06379, 2024. 14, 30

[84] S. Teuber, S. Mitsch, and A. Platzer, "Provably safe neural network controllers via differential dynamic logic," *ArXiv*, vol. abs/2402.10998, 2024. 15, 16, 30

[85] Y. Tsuzuku, I. Sato, and M. Sugiyama, "Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks," in *Neural Information Processing Systems*, 2018, pp. 6542–6551. 16, 26

[86] K. Dvijotham, S. Gowal, R. Stanforth, R. Arandjelović, B. O'Donoghue, J. Uesato, and P. Kohli, "Training verified learners with learned verifiers," *ArXiv*, vol. abs/1805.10265, 2018. 16, 21, 26, 29

[87] F. Leofante, N. Narodytska, L. Pulina, and A. Tacchella, "Automated verification of neural networks: Advances, challenges and perspectives," *ArXiv*, vol. abs/1805.09938, 2018. 17, 18, 19, 27, 29

[88] B. Batten, M. Hosseini, and A. Lomuscio, "Tight verification of probabilistic robustness in bayesian neural networks," *ArXiv*, vol. abs/2401.11627, 2024. 18, 25, 30

[89] Z. Shi, Q. Jin, Z. Kolter, S. Jana, C.-J. Hsieh, and H. Zhang, "Neural network verification with branch-and-bound for general nonlinearities," *ArXiv*, vol. abs/2405.21063, 2024. 18, 28

[90] T. Baluta, Z. L. Chua, K. S. Meel, and P. Saxena, "Scalable quantitative verification for deep neural networks," *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, pp. 312–323, 2020. 18, 19

[91] G. Amir, H. Wu, C. W. Barrett, and G. Katz, "An smt-based approach for verifying binarized neural networks," *Tools and Algorithms for the Construction and Analysis of Systems*, vol. 12652, pp. 203 – 222, 2020. 18, 25, 30

[92] Y.-F. Chen, C.-E. Hsieh, O. Lengál, T.-J. Lii, M. Tsai, B.-Y. Wang, and F. Wang, "Pac learning-based verification and model synthesis," *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*, pp. 714–724, 2015. 18

[93] R. Mangal, N. Narodytska, D. Gopinath, B. C. Hu, A. Roy, S. Jha, and C. S. Păsăreanu, "Concept-based analysis of neural networks via vision-language models," *ArXiv*, vol. abs/2403.19837, 2024. 18, 24

[94] S. Jafarpour, A. Harapanahalli, and S. Coogan, "Interval reachability of nonlinear dynamical systems with neural network controllers," *ArXiv*, vol. abs/2301.07912, 2023. 19, 25

[95] B. Paulsen, J. Wang, and C. Wang, "Reludiff: Differential verification of deep neural networks," *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, pp. 714–726, 2020. 19

[96] A. Albarghouthi, "Introduction to neural network verification," *ArXiv*, vol. abs/2109.10317, 2021. 19

[97] J. Skovbekk, L. Laurenti, E. Frew, and M. Lahijanian, "Formal abstraction of general stochastic systems via noise partitioning," *IEEE Control Systems Letters*, vol. 7, pp. 3711–3716, 2023. 22

[98] L. Lindemann, Y. Zhao, X. Yu, G. J. Pappas, and J. V. Deshmukh, "Formal verification and control with conformal prediction," *ArXiv*, vol. abs/2409.00536, 2024. 22

[99] R. Bunel, A. Palma, A. Desmaison, K. Dvijotham, P. Kohli, P. H. S. Torr, and M. P. Kumar, "Lagrangian decomposition for neural network verification," in *Conference on Uncertainty in Artificial Intelligence*, 2020, pp. 370–379. 23, 28

[100] S. Bassan and G. Katz, "Towards formal xai: Formally approximate minimal explanations of neural networks," in *International Conference on Tools and Algorithms for Construction and Analysis of Systems*, 2022, pp. 187–207. 23, 24, 29

[101] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Output range analysis for deep neural networks," *ArXiv*, vol. abs/1709.09130, 2017. 24

[102] J. Renkhoff, K. Feng, M. Meier-Doernberg, A. Velasquez, and H. H. Song, "A survey on verification and validation, testing and evaluations of neurosymbolic artificial intelligence," *IEEE Transactions on Artificial Intelligence*, vol. 5, pp. 3765–3779, 2024. 25, 27

[103] D. Li, M. Cai, C. Vasile, and R. Tron, "Learning signal temporal logic through neural network for interpretable classification," *2023 American Control Conference (ACC)*, pp. 1907–1914, 2022. 27

[104] H. Wu, A. Zelji'c, G. Katz, and C. W. Barrett, "Efficient neural network analysis with sum-of-infeasibilities," in *International Conference on Tools and Algorithms for Construction and Analysis of Systems*, 2022, pp. 143–163. 28

[105] T. Baluta, S. Shen, S. Shinde, K. S. Meel, and P. Saxena, "Quantitative verification of neural networks and its security applications," *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019. 29