

3.14 Tools, Frameworks, and Verification Workflows Overview of Neural Network Verification Tools

Neural network verification tools analyze whether neural networks satisfy properties such as safety and robustness. Combining formal verification with systematic testing, risk management, and continuous monitoring could strengthen the technical evidence supporting certification. The authors implemented a prototype Neuro-Symbolic Verifier (NSV) based on the Marabou verification framework and evaluated it using neural networks trained on the MNIST handwritten-digit dataset and the German Traffic Sign Recognition Benchmark (GTSRB). Standardized benchmarks, such as those used in the International Verification of Neural Networks Competition (VNN-COMP), facilitate more consistent comparisons by using common network and specification formats and controlled evaluation conditions. However, comparative studies show that verification performance depends on the characteristics of individual problem instances. The authors reported that the integration of structured requirements management, technical evaluation methods, and automated MLOps workflows supports traceability, reproducibility, and the systematic assessment of trustworthy AI objectives.

Evaluation Metrics, Benchmark Datasets, and Experimental Design Liu et al. (2024) evaluated their NARv approach using verification time, the number of successfully verified properties, and the sizes of abstract and refined neural networks. Xie et al. (2022) proposed a neuro-symbolic verification framework that addresses a limitation of traditional neural network verification methods, which mainly support properties expressed as logical constraints over network inputs and outputs. Their method uses Property Specification Patterns (PSP) to generate formal properties and observer automata, while Property Adaptation Patterns (PAP) enable these observers to be safely modified during execution without restarting the verification process. Kwiatkowska et al. (2011) introduced PRISM 4.0, which extended quantitative verification to priced probabilistic timed automata and supported techniques such as explicit-state model checking, statistical model checking, and optimal strategy generation. Brix et al. (2023) analyzed the first three editions of the International Verification of Neural Networks Competition (VNN-COMP), which evaluated tools on safety and robustness problems involving image classification, neural control, reinforcement learning, and autonomous systems. At the software engineering level, Chong et al. (2021) demonstrate how formal proofs can be integrated into continuous integration (CI), where verification artifacts are maintained with the codebase and checked as part of routine development workflows. Second, the evaluation focused on robustness against adversarial input perturbations and used Marabou and Planet as backend verification engines; the authors did not evaluate NARv with abstract-interpretation-based verification approaches. Henzinger et al. (2021) showed that verifying quantized neural networks with bit-vector specifications is PSPACE-hard, highlighting the computational difficulty of reasoning about rounding and fixed-point arithmetic. To improve scalability, they introduced three SMT-encoding optimizations: removing unreachable activation branches, minimizing bit-vector widths, and eliminating redundant multiplications. Future research should extend certification frameworks to cover additional trustworthiness dimensions, establish more widely accepted assessment metrics, and develop methods suited to increasingly complex systems, including foundation models and large language models. Gross (2023) examined the reproducibility of three studies and identified practical challenges involving software dependencies, hardware compatibility, missing implementations, and incomplete experimental documentation.

Runtime Validation of Generative

AI Outputs : Large language models (LLMs) can generate inconsistent or erroneous outputs, which raises reliability concerns in high-stakes applications. Overall, the findings demonstrate the potential of abstraction-refinement techniques to improve the scalability of formal neural network verification without sacrificing soundness or completeness under the approach's assumptions. Pulina and Tacchella (2012) introduced a Counter-Example Triggered Abstraction-Refinement (CETAR) approach that uses Quantifier-Free Linear Arithmetic (QF_LRA) to verify abstract network representations. Kaminwar et al. (2023) describe a structured approach that considers data, code, and model verification across development phases, emphasizing that evaluating predictive performance alone is insufficient to establish the reliability of an ML system. Isolated environments, such as Anaconda environments or Docker containers, can help manage software dependencies, while recording random seeds supports reproducibility when training involves randomness. POLAR provides polynomial arithmetic methods, whereas NNV supports multiple neural network architectures and learning-enabled control systems. PRISM is an open-source probabilistic model checker.