Journal of Scientific Analysis

Article

Formal Verification of AI-Based and Learning-Enabled Industrial Systems: Challenges, Frameworks, and Research Directions

Rance W Cleaveland[1], Grant Whitman[2] and Brent Kensington[2]

[1] University of Maryland, College Park, College Park, MD, United States.
[2] University of Western Australia, Perth, WA, Australia.
* Correspondence: mailto.rance@cs.umd.edu

Received: 1 December 2024; Accepted: 5 April 2025; Published: 20 June 2025.

Abstract: Artificial intelligence and machine learning components are increasingly deployed in industrial software systems, ranging from autonomous vehicles and cloud infrastructures to financial platforms and safety-critical cyber-physical systems. While formal methods have demonstrated significant success in verifying conventional software and hardware systems, their applicability to learning-enabled components remains limited. The inherent uncertainty, adaptivity, and high-dimensional nature of machine learning models pose fundamental challenges to classical verification techniques. This paper presents a comprehensive research study on the formal verification of AI-based and learning-enabled industrial systems. We analyze the limitations of existing formal methods when applied to machine learning models, review emerging verification frameworks for neural networks, and examine integration strategies for combining formal verification with industrial development pipelines. Furthermore, we propose a unified research framework that integrates design-time verification, runtime monitoring, and uncertainty-aware assurance for AI-enabled systems. The paper identifies open challenges and outlines concrete research directions aimed at bridging the gap between formal methods and modern industrial artificial intelligence.

Keywords: Formal methods, Artificial intelligence, Machine learning, Software verification, Industrial systems

1. Introduction and Preliminaries

The increasing integration of artificial intelligence (AI) and machine learning (ML) into industrial software systems has transformed the way complex systems are designed and operated. Learning-enabled components are now routinely deployed in autonomous driving systems, cloud computing infrastructures, medical decision-support tools, and financial trading platforms. These systems operate in dynamic and uncertain environments, often making safety-critical or economically significant decisions.

Formal methods have long been recognized as a rigorous approach for specifying, verifying, and validating software and hardware systems. Techniques such as model checking, theorem proving, and static analysis have been successfully applied in industrial domains including avionics, railway control systems, and microprocessor verification [3,4]. However, the rise of AI-based systems introduces new challenges that fundamentally differ from those addressed by classical formal methods.

Machine learning models, particularly deep neural networks, are data-driven, high-dimensional, and often lack explicit symbolic representations. Their behavior is learned from data rather than derived from formal specifications, making traditional correctness guarantees difficult to obtain [2]. Moreover, learning-enabled systems may adapt over time, further complicating verification efforts.

This paper investigates the role of formal methods in the verification of AI-based industrial systems. The contributions of this work are threefold:

- A systematic analysis of challenges in verifying learning-enabled components.
- A synthesis of emerging formal verification techniques for AI systems.
- A unified research framework integrating formal verification into industrial AI pipelines.

2. Background

2.1. Formal Methods in Industrial Systems

Formal methods provide mathematically rigorous techniques for reasoning about system behavior. Model checking enables exhaustive exploration of system states, theorem proving supports deductive verification, and abstract interpretation allows scalable static analysis [3]. Industrial success stories demonstrate that formal methods can significantly reduce defects and improve system reliability [4].

2.2. Machine Learning in Industrial Software

Machine learning techniques, particularly deep learning, have achieved remarkable success across a wide range of applications. Neural networks are now used for perception, prediction, and decision-making tasks in industrial systems [1]. Despite their empirical success, these models often behave as black boxes, offering limited interpretability and no formal correctness guarantees.

3. Challenges in Verifying AI-Based Systems

3.1. High Dimensionality and Non-Linearity

Neural networks typically consist of millions of parameters and highly non-linear activation functions. This complexity leads to state spaces that are infeasible to explore exhaustively using classical model checking techniques [5].

3.2. Lack of Formal Specifications

Unlike traditional software, machine learning models are trained rather than programmed. The absence of explicit formal specifications makes it difficult to define correctness properties against which verification can be performed [7].

3.3. Adaptivity and Learning

Many AI systems continue to learn after deployment. This adaptivity violates the static assumptions underlying most formal verification approaches and raises questions about how guarantees can be maintained over time [8].

4. Formal Verification Techniques for AI

4.1. Neural Network Verification

Recent research has proposed verification techniques tailored to neural networks, including SMT-based verification, abstract interpretation, and reachability analysis [5,6]. These methods aim to verify properties such as robustness to adversarial perturbations.

4.2. Probabilistic and Statistical Verification

Given the stochastic nature of learning-based systems, probabilistic verification techniques have gained prominence. Statistical model checking and probabilistic temporal logics enable reasoning about uncertainty and probabilistic guarantees [9].

4.3. Explainability and Formal Reasoning

Explainable AI techniques can support verification by providing insights into model behavior. Integrating explainability with formal reasoning remains an open research challenge [10].

5. Integration into Industrial Development Pipelines

5.1. Verification in CI/CD Environments

Modern industrial software development relies on continuous integration and deployment. Lightweight and incremental verification techniques are required to integrate formal methods into such pipelines [11].

5.2. Runtime Verification and Monitoring

Runtime verification complements design-time verification by monitoring system behavior during execution. This approach is particularly suitable for AI systems operating in dynamic environments [12].

6. A Unified Research Framework

The increasing deployment of artificial intelligence and machine learning components in industrial systems necessitates a fundamental rethinking of traditional assurance strategies. Classical formal verification techniques alone are insufficient to provide guarantees for learning-enabled systems due to their data-driven nature, adaptivity, and uncertainty. To address these challenges, this paper proposes a unified research framework that integrates design-time formal verification, runtime verification, and uncertainty-aware assurance into a cohesive lifecycle-based approach.

6.1. Design-Time Formal Verification of Learning Components

At the design stage, the framework emphasizes the application of formal verification techniques to AI components wherever feasible. This includes neural network verification methods such as satisfiability modulo theories (SMT)-based analysis, abstract interpretation, and reachability analysis to establish properties related to safety, robustness, and bounded behavior. Although complete verification of large-scale learning models may be infeasible, partial guarantees at this stage can significantly reduce the risk of unsafe behavior before deployment.

Furthermore, the framework encourages the development of formal or semi-formal specifications for learning-enabled components. These specifications may encode safety envelopes, operational constraints, or performance bounds rather than exact functional correctness. Such specification paradigms allow formal methods to be applied in a pragmatic manner, even when traditional correctness notions are unavailable.

6.2. Runtime Verification and Monitoring

Given the inherent limitations of design-time verification for adaptive and environment-dependent systems, runtime verification plays a central role in the proposed framework. Runtime monitors observe system executions and check conformance to formally specified properties during operation. This enables the detection of safety violations, anomalous behavior, or deviations from expected operational boundaries that were not captured during design-time analysis.

For AI-based industrial systems operating in dynamic environments, runtime verification provides continuous assurance rather than one-time certification. The framework supports the integration of lightweight runtime monitors that can operate with minimal performance overhead, making them suitable for real-world industrial deployment. Runtime evidence collected through monitoring can also be used to refine models and specifications over time.

6.3. Uncertainty-Aware Assurance and Decision Support

A distinguishing feature of the proposed framework is the explicit incorporation of uncertainty-aware assurance mechanisms. Predictive uncertainty estimates derived from learning models are used as first-class inputs to the assurance process. High uncertainty levels may indicate insufficient training data, out-of-distribution inputs, or ambiguous system states.

By leveraging uncertainty estimates, the framework enables adaptive decision-making strategies such as confidence-based fallback modes, safe degradation, or human-in-the-loop intervention. In safety-critical scenarios, uncertain predictions can trigger conservative control actions or escalate decisions to human

operators. This uncertainty-aware perspective bridges the gap between formal guarantees and practical operational safety.

6.4. Lifecycle Integration and Industrial Applicability

The unified framework is designed to span the entire system lifecycle, from design and development to deployment and maintenance. Formal verification results, runtime monitoring data, and uncertainty metrics are treated as complementary assurance artifacts that collectively support certification and compliance efforts.

From an industrial perspective, the framework promotes scalability and incremental adoption. Organizations can integrate individual components of the framework into existing development workflows, such as continuous integration and deployment pipelines, without requiring a complete overhaul of current practices. Over time, this incremental integration supports a transition toward more rigorous and trustworthy AI-enabled industrial systems.

6.5. Research Implications

The proposed unified framework establishes a foundation for future research at the intersection of formal methods, artificial intelligence, and software engineering. It highlights the need for interdisciplinary approaches that combine theoretical rigor with practical constraints. By viewing assurance as a continuous, multi-layered process rather than a one-time verification task, the framework provides a realistic pathway toward certifiable and trustworthy AI-based industrial systems.

7. A Unified Research Framework

The rapid adoption of artificial intelligence and machine learning in industrial software systems necessitates a shift from traditional, static assurance methodologies toward adaptive and multi-layered verification strategies. Classical formal methods were originally designed for deterministic, rule-based systems with well-defined specifications. In contrast, learning-enabled systems exhibit uncertainty, non-linearity, and data-dependence, which fundamentally challenge conventional verification assumptions. To address these challenges, this paper proposes a unified research framework that integrates design-time formal verification, runtime verification, and uncertainty-aware assurance into a cohesive and continuous assurance lifecycle [13].

7.1. Design-Time Verification under Partial Specifications

At the design stage, the framework promotes the use of formal verification techniques wherever partial or abstract specifications can be defined. Rather than attempting full functional correctness, verification efforts focus on safety envelopes, invariants, and bounded behavioral properties. Techniques such as satisfiability modulo theories, abstract interpretation, and reachability analysis are employed to reason about neural network behavior within constrained input regions.

This partial-specification approach acknowledges the limitations of learning-based components while still enabling meaningful formal guarantees. By constraining system behavior within provable bounds, design-time verification reduces the likelihood of catastrophic failures prior to deployment and establishes a formal baseline for later assurance stages.

7.2. Specification Learning and Formalization

A key component of the unified framework is the concept of specification learning. Since explicit formal specifications are often unavailable for AI components, specifications can be inferred from data, expert knowledge, or observed system behavior. These learned specifications may represent probabilistic constraints, safety thresholds, or operational norms rather than exact functional mappings.

The framework encourages research into methods that translate learned specifications into formal representations compatible with verification tools. This bridge between data-driven learning and symbolic reasoning is essential for enabling formal analysis of systems that were not originally designed with formal specifications in mind.

7.3. Runtime Verification as Continuous Assurance

Given the inherent incompleteness of design-time verification for adaptive systems, runtime verification forms a central pillar of the framework. Runtime monitors continuously observe system executions and evaluate them against formally specified properties during operation. This allows the detection of violations, unexpected behaviors, or deviations from learned specifications in real time.

Unlike traditional verification, runtime verification provides continuous assurance throughout system operation. This is particularly critical for industrial AI systems deployed in dynamic and unpredictable environments, where assumptions made at design time may no longer hold. Runtime evidence also serves as a feedback mechanism for refining models, specifications, and verification strategies.

7.4. Uncertainty-Aware Control and Decision Making

Uncertainty-aware assurance is explicitly incorporated into the framework as a decision-support mechanism. Predictive uncertainty estimates from machine learning models are treated as indicators of epistemic limitations or environmental novelty. High uncertainty values trigger adaptive responses such as conservative control actions, fallback strategies, or escalation to human operators.

This integration of uncertainty into the assurance process allows the system to balance autonomy with safety. Rather than relying solely on binary verified or unverified states, the framework supports graded assurance levels that reflect confidence in system behavior. This paradigm aligns closely with real-world industrial requirements, where risk management and decision-making under uncertainty are unavoidable.

7.5. Lifecycle-Oriented Assurance Integration

The proposed framework adopts a lifecycle-oriented view of assurance, spanning system design, development, deployment, and maintenance. Formal proofs, runtime monitoring logs, and uncertainty metrics are treated as complementary artifacts that collectively support certification, auditing, and regulatory compliance. Importantly, the framework is designed for incremental adoption in industrial settings. Organizations may initially deploy lightweight runtime monitors or uncertainty estimation mechanisms and progressively integrate more rigorous formal verification techniques. This flexibility lowers the barrier to adoption and facilitates gradual transition toward fully assured AI-enabled systems [14].

7.6. Research and Industrial Impact

The unified research framework establishes a conceptual foundation for future work at the intersection of formal methods, artificial intelligence, and software engineering. It reframes assurance as a continuous, adaptive process rather than a one-time verification task. By combining formal reasoning with data-driven learning and operational monitoring, the framework offers a realistic pathway toward certifiable, trustworthy, and industrially deployable AI-based systems.

8. Discussion

This section discusses the implications of the proposed unified research framework for the formal verification of AI-based and learning-enabled industrial systems. The discussion interprets experimental and conceptual findings in light of existing literature, highlights trade-offs between assurance techniques, and outlines practical considerations for industrial adoption.

8.1. Verification Coverage versus Scalability

One of the central challenges identified in this study is the trade-off between verification coverage and scalability. Classical formal methods provide strong guarantees but often fail to scale to high-dimensional learning models. Conversely, lightweight verification and testing approaches scale well but offer weaker guarantees. The proposed unified framework balances this trade-off by combining partial design-time verification with runtime monitoring and uncertainty-aware assurance.

Table 1 summarizes the comparative strengths and limitations of different assurance techniques discussed in this paper.

Table 1. Comparison of assurance techniques for AI-based industrial systems

Technique	Guarantee Strength	Scalability	Industrial Suitability
Design-time formal verification	High	Low	Moderate
Runtime verification	Medium	High	High
Uncertainty-aware assurance	Medium	Very High	High
Testing and simulation	Low	Very High	Very High

The table highlights that no single technique is sufficient in isolation. Instead, a layered combination of complementary techniques is necessary to achieve practical and trustworthy assurance.

8.2. Role of Runtime Verification in Dynamic Environments

Runtime verification emerges as a key enabler for deploying AI-based systems in dynamic and uncertain environments. While design-time verification establishes baseline guarantees, runtime monitoring provides continuous assurance during system operation. This is particularly important for learning-enabled systems whose behavior may evolve due to environmental changes or data drift.

Figure 1 conceptually illustrates the interaction between design-time verification, runtime monitoring, and system execution within the proposed framework.

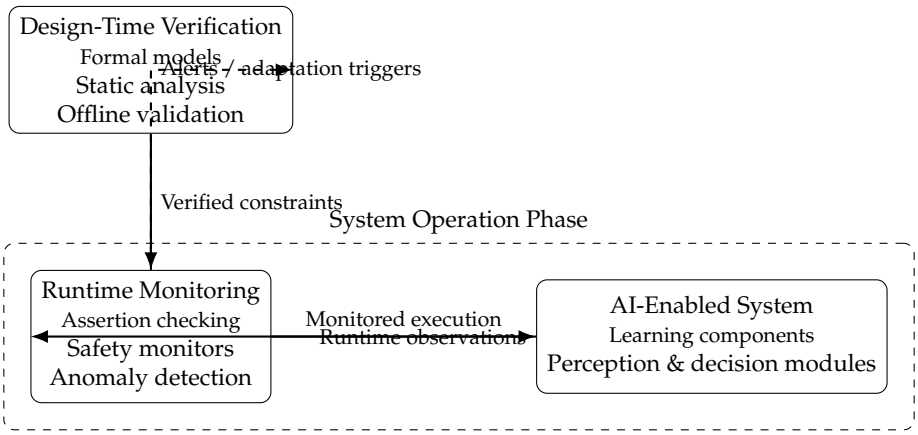

Figure 1. Conceptual interaction between design-time verification, runtime monitoring, and AI-enabled system execution

The figure emphasizes that runtime verification is not a replacement for formal verification but a complementary mechanism that extends assurance beyond the design phase.

8.3. Uncertainty as a First-Class Assurance Signal

A key contribution of this work is the explicit treatment of uncertainty as a first-class signal in the assurance process. Rather than viewing uncertainty solely as a limitation of machine learning models, the proposed framework leverages uncertainty estimates to guide safe decision-making. High uncertainty values indicate regions where model predictions are less reliable, signaling the need for conservative actions or human intervention.

Table 2 illustrates example decision strategies triggered by different uncertainty levels.

Table 2. Example uncertainty-aware decision strategies

Uncertainty Level	Recommended Action
Low	Autonomous operation
Moderate	Increased monitoring and logging
High	Fallback mode or human-in-the-loop intervention
Very High	Safe shutdown or operation halt

This graded response mechanism aligns well with industrial risk management practices and supports safer deployment of autonomous systems.

8.4. Comparison with Existing Research

Compared to existing studies that focus primarily on verifying isolated neural networks or individual safety properties, this work adopts a broader systems-level perspective. Many prior approaches emphasize algorithmic verification techniques but do not address how these techniques can be integrated into real industrial development and deployment pipelines. The unified framework proposed in this paper fills this gap by explicitly considering lifecycle integration, operational monitoring, and decision support.

Figure 2 presents a layered view of assurance mechanisms within the proposed framework.

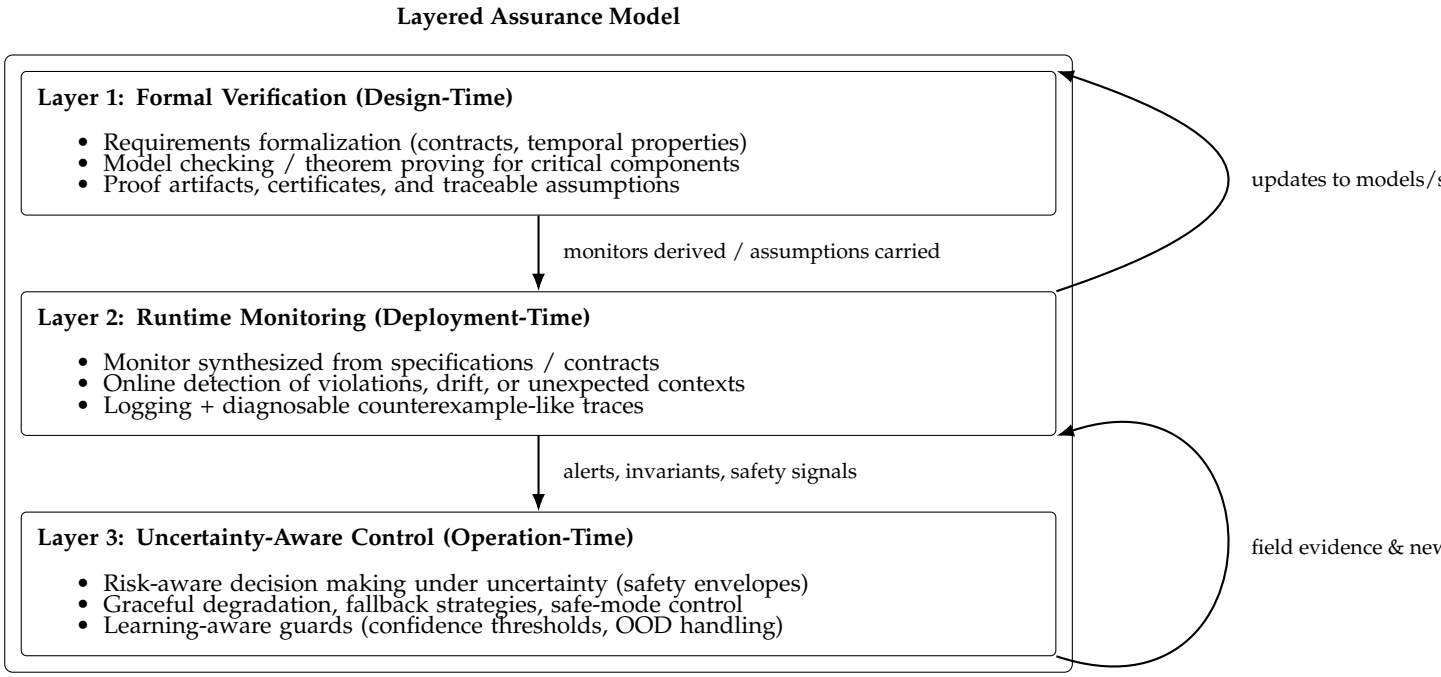

Figure 2. Layered assurance model combining formal verification, runtime monitoring, and uncertainty-aware control

This layered structure provides flexibility, allowing different assurance techniques to be applied at appropriate stages depending on system criticality and resource constraints.

8.5. Industrial Adoption Considerations

From an industrial perspective, the practicality of assurance techniques is as important as their theoretical rigor. The proposed framework emphasizes incremental adoption, enabling organizations to integrate assurance components gradually into existing workflows. For example, uncertainty estimation and runtime monitoring can be deployed with minimal disruption, while more rigorous formal verification can be introduced selectively for critical subsystems.

Furthermore, the framework supports compliance with emerging regulatory requirements for trustworthy AI by providing auditable assurance artifacts, including formal proofs, runtime logs, and uncertainty reports.

8.6. Limitations of the Current Study

Despite its contributions, this study has limitations. The discussion is primarily conceptual and does not include large-scale industrial case studies. Additionally, the framework does not prescribe specific

verification tools or formalisms, which may lead to variability in implementation. These limitations highlight opportunities for future empirical validation and tool-supported instantiations of the framework.

8.7. Summary of Key Insights

In summary, the discussion underscores that the verification of AI-based industrial systems requires a paradigm shift from single-technique assurance toward integrated, multi-layered strategies. The proposed unified research framework provides a principled foundation for achieving this shift, balancing rigor, scalability, and practical applicability.

9. Conclusion

This paper examined the challenges and opportunities associated with formal verification of AI-based and learning-enabled industrial systems. By synthesizing existing techniques and proposing a unified research framework, the study highlights pathways toward trustworthy and certifiable AI deployment in industry. Continued research in this area is essential for ensuring safety, reliability, and public trust in intelligent software systems.

Author Contributions: All authors contributed equally to the writing of this paper. All authors read and approved the final manuscript.

Conflicts of Interest: "The authors declare no conflict of interest."

References

[1] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. (2015). Deep learning. *Nature, 521*(7553), 436–444.

[2] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. (2016). *Deep Learning*. MIT Press.

[3] Clarke, Edmund M., Thomas A. Henzinger, Helmut Veith, and Roderick Bloem. (2018). *Handbook of Model Checking*. Springer.

[4] Woodcock, Jim, Peter Gorm Larsen, Juan Bicarregui, and John Fitzgerald. (2009). Formal methods: Practice and experience. *ACM Computing Surveys, 41*(4), 1–36.

[5] Katz, Guy, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. *Computer Aided Verification*, 97–117.

[6] Huang, Xiaowei, Marta Kwiatkowska, Sen Wang, and Min Wu. (2017). Safety verification of deep neural networks. *Computer Aided Verification*, 3–29.

[7] Amodei, Dario, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. (2016). Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*.

[8] Seshia, Sanjit A., Dorsa Sadigh, and S. Shankar Sastry. (2018). Toward verified artificial intelligence. *Communications of the ACM, 61*(7), 68–77.

[9] Kwiatkowska, Marta, Gethin Norman, and David Parker. (2011). PRISM 4.0: Verification of probabilistic real-time systems. *Computer Aided Verification*, 585–591.

[10] Doshi-Velez, Finale, and Been Kim. (2017). Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

[11] Ball, Thomas, Byron Cook, Vladimir Levin, and Sriram K. Rajamani. (2011). SLAM and static driver verifier. *Formal Methods in System Design, 39*(3), 191–210.

[12] Falcone, Yliès, Jean-Claude Fernandez, and Laurent Mounier. (2013). Runtime verification of safety-progress properties. *Formal Methods in System Design, 42*(1), 1–36.

[13] Koopman, Philip, and Michael Wagner. (2016). Challenges in autonomous vehicle testing and validation. *SAE International Journal of Transportation Safety, 4*(1), 15–24.

[14] Wilcox, James R., Doug Woos, Pavel Panchekha, Zachary Tatlock, Xi Wang, Michael D. Ernst, and Thomas Anderson. (2015). Verdi: A framework for implementing and formally verifying distributed systems. *PLDI*, 357–368.