LICENTIATE THESIS PROPOSAL

Explainable Machine Learning for Predictive Modeling and Abstraction in Industrial-Scale Systems

Edin Jelačić

School of Innovation, Design and Engineering (IDT)

Mälardalen University

edin.jelacic@mdu.se

March 11, 2026

Main supervisor: Prof. Tiberiu Seceleanu

Co-supervisors: Prof. Cristina Seceleanu, Prof. Ning Xiong, Assoc. Sen. Lecturer Peter Backeman

Contents

Included Papers

This thesis proposal is based on the following included papers:

- **Paper A** ”Abstraction-based Reduction of Input Size for Neural Networks”. Published in the First International Conference on Bridging the Gap between AI and Reality (AISoLA 2023). Authors: Peter Backeman, Edin Jelačić, Cristina Seceleanu, Ning Xiong, Tiberiu Seceleanu.

- **Paper B** ”A Conformal Prediction-Based Framework for CPU Load Forecasting: A Black-Box Approach”. Published in the 49th IEEE International Conference on Computers, Software, and Applications (COMPSAC 2025) in July, 2025. Authors: Edin Jelačić, Cristina Seceleanu, Peter Backeman, Tiberiu Seceleanu, Axel Jantsch.

- **Paper C** ”Machine learning-based cache miss prediction”. Published in the International Journal on Software Tools for Technology Transfer (STTT) in April, 2025. Authors: Edin Jelačić, Cristina Seceleanu, Ning Xiong, Peter Backeman, Sharifeh Yaghoobi, Tiberiu Seceleanu.

- **Paper D** “HASCo: Hybrid AI Simulation Compiler - Evaluating Agentic LLMs for Usable Scenario Generation in Road-Traffic Simulation”. In preparation for submission to the IEEE Computational Intelligence Magazine (CIM), 2025, SI on Privacy- Preserving Machine and Deep Learning. Authors: Edin Jelačić, Cristina Seceleanu, Peter Backeman, Tiberiu Seceleanu, Rong Gu, Ali Naur, Zhennan Fei, Ammara Asif.

Abstract

Machine learning is increasingly called upon to guide decisions in critical industrial applications. Its predictive power promises efficiency and adaptability, yet its black-box nature and lack of guarantees pose risks in contexts where behavior must remain analyzable and safe. This thesis asks how machine learning can be strengthened so that it becomes not only powerful, but also accountable, explainable, and usable by engineers in practice with a need for understanding the reasoning for system outputs. This is done by acknowledging some of the key gaps hampering more widespread adoption of machine learning in industrial-scale systems. Firstly, there is a comparative scarcity of work in providing guarantees for machine learning-based systems. Secondly, there is an outstanding requirement for data-driven models of industrial devices that generalize outside of singular use-cases. Finally, many realizations of machine learning-based systems address individual aspects of industry-inspired problems, without combining these solutions into entire end-to-end pipelines. Our work consists of several contributions which aim to address these gaps and their combined curiosity in industry-inspired research. Our first contribution posits the recognition that models must often be reduced before it may be shown that they can be trusted. Therefore, by applying abstraction to neural networks, we showed that inputs that have little effect on outcomes can be formally identified and removed, producing simpler yet bounded models that remain open to analysis. Secondly, we address the demand for an understanding of forecasts' statistical reliability. We introduced conformal prediction for providing coverage guarantees with Shapley values (a statistical concept utilized for approximating the allocation of contribution of individual input components to the overall result [1]) to trace load contributions back to individual tasks. In this way, we turned raw processor traces into safety-relevant insight. The next contribution was the development of a data-driven simulator for cache-behavior. Instead of replacing traditional hardware simulators outright, we reproduced input–output behavior at a fraction of the computational cost. The final contribution was HASCo, a Hybrid AI Simulation Compiler that combined all of the thus attained principles into an end-to-end pipeline. This artifact produces runnable simulation scenarios from natural-language safety reports. In addition to HASCo, we introduced an Effort-to-Usability metric to measure how easily engineers can adapt the results. Seen together, these contributions establish a path toward machine learning that does not remain entirely a black box but becomes a more transparent partner in the industrial workflow.

1 Introduction

In contemporary industrial environments that utilize vast amounts of digital infrastructure, there exists an abundance of structured, semi-structured or perhaps entirely unstructured data generated by the combination of system devices' operations and the engineers operating these system devices [2, 3, 4]. At the same time, industrial environments often suffer from lack of optimization with regards to problem-solving, either due to cyber-physical limitations of the systems being worked on or due to the sheer magnitude of complexity of the processes required to solve these problems [5, 6]. These two simultaneous intricacies, as well as the pervasiveness of machine learning in modern contexts, open up the door to out-of-the-box approaches that may have previously been unthinkable for developers actively tackling these problems. The ability of universal function approximators to provide powerful insights into vast amounts of data, particularly industry-specific measurements and aggregated expert insights, is no longer an avoidable phenomenon [7, 8]. Thus, ML is treated as a pragmatic tool for reasoning from vast quantities of industrial data. We show in this work, that this same data is crucial for understanding both how these systems operate and how the problems arising in industrial environments may be solved in novel ways, while leaving room for potential improvements on the way in the future, as the field of ML progresses. At the same time, there is a vast body of work in the direction of verifying ML approaches and ensuring compatibility with industrial safety standards [9]. We researched applied ML in automation contexts of industrial development, and it is based on this that we postulate this body of work.

The key component of more widespread adoption of ML in industrial contexts is attaining stakeholder trust, with regards to their particular use-case and environment. AI is a broad field of technologies for building machines with abilities of mimicking cognitive functions associated with human intelligence, whereas ML is a subset of AI that enables a system to progressively improve from experience. The disambiguation of AI and ML is a significant step in the right direction towards attaining stakeholder trust with regards to the risks and costs of an ML system. There exist tradeoffs with utilizing an ML system that must be clearly communicated between ML practitioners and stakeholders, such as the complexity of evaluation of ML-produced results. Finally, cost is a significant hurdle, as the introduction of ML pipelines, typically on hardware foreign to the original intent of the industrial environment, must justify the cost to the stakeholders. It is for these reasons that a key aspect of any ML introduction must be a solution to ensure *trustworthiness*, *explainability* and *efficiency* of the approach. By *trustworthiness* here, we are referring to the level of trust that stakeholders may reasonably have in the results of ML-based analysis, stemming from inherent properties of the ML system and all of its core components. By *explainability* in this context, we are referring to the clarity of the reasoning behind the ML-based analysis

and the ability of the ML system to manifest its logic in an understandable way. Lastly, by *efficiency*, we mean the ability of the ML system to demonstrably be a financial net-positive for the stakeholders, i.e., be able to clearly demonstrate its value proposition in terms of cost-benefit analysis. This is a particularly acute point, as the widespread total transition of a large deal of businesses towards AI-based solutions has had mixed to poor investment results [10]. It is therefore pertinent that successful AI adoption may not be achieved by chasing trends, but rather about thoughtful planning, strategic implementation procedures, and a clear view on how to measure the success of the implementation.

Keeping the above in mind, our overarching research goal is designing of ML components to tackle industrial challenges with the express purpose of explainability by design and appropriate guarantees on their outputs, with evaluation of usefulness in corresponding industrial contexts. Our research builds on this goal through a series of studies that gradually connect model abstraction, predictive guarantees, hardware-aware learning, and generative methods for simulation.

Initially, our main concern has been that neural networks (**NNs**), being some of the fundamental components upon which many ML systems are built [11], often become too large due to incorrect outright design, and come with the severe drawback of their black-box nature for the practitioners utilizing them for modeling safety-critical systems [12]. Additionally, ML systems and NNs in particular have come under strict regulator scrutiny, with various ethical concerns [13] and explainability of results, particularly with respect to European GDPR law [14] being the focal points. Significant work has been done in the field of formal NN verification [9], with many aspects of NNs being able to be checked and many tools for this formal checking arising in recent years [15, 16]. Of particular interest to us was the Marabou verification tool [17], which is continuously being updated and actively worked on. This tool can prove a linear bound on the output of an NN, given a constraint on the inputs to the NN, and provided that the NN utilizes linear or piecewise-linear activation functions. Elboher et al. [18] utilized the Marabou tool for formal verification of an abstracted NN, reduced in size by counter-example guided abstraction refinement [19]. In terms of execution time, they prove that through formal methods a reduction of the internal layers of the NN leads to faster verification. In our first contribution [20], we extend this idea for abstraction-based NN analysis, but instead of the internal layers, we shift focus onto formulating an over-under approximating NN abstraction for finding insignificant inputs to the NN. In this work, we show with two concrete algorithms, one for creating over/under approximating NNs and another for creating difference NNs, how this technique may be used to reduce model size and thereby improve inference speed and reduce model operating cost while keeping the variation of the NN's output finitely bounded. This approach not only reduces complexity but also provides formal guarantees on model behavior, offering a step toward neural networks that engineers can analyze even when training data is

unavailable.

Building on this notion of trustworthy simplification, we then turn to a real industrial problem: ensuring stable processor load under changing software configurations. Namely, it is our intent to utilize expertise attained throughout the time spent working on ML model trustworthiness by resolving a specific central processing unit (**CPU**) overloading concern of one of our industrial partners, which gave rise to the research problem of interest. This overloading manifested with some regularity in scenarios where additional tasks would be activated on the device by customers, beyond the manufacturer's original intention. Given that the device in question is a real-time industrial-scale processing unit, adding or modifying tasks running on the device have the potential to overload the device's processing unit and compromise real-time performance [21, 22]. The engineers need to know the forecast of the CPU load, given a set of tasks to be activated on the device, how uncertain this forecast is, and a quantification of each task's specific contribution to this load. For this reason, it is not sufficient to merely build an ML framework for statistical forecasting of future CPU loads based on tabular data, irrespective of the peculiarities of the underlying regression model that would act as the data-driven substitute for the device. Our answer to this is building and testing of an empirically grounded ML framework for forecasting CPU load that would operate on a black-box regression model and deliver uncertainty quantification, as well as feature attributions to this prediction. For model-agnostic uncertainty quantification, we utilized the conformal prediction framework as a tool of quantile regressor calibration, which means that regardless of the underlying predictive model, we can construct statistically valid prediction intervals whose coverage probability (probability of the predicted label being within the interval) is guaranteed under minimal assumptions (exchangeability of data). In practice, this entails taking the raw quantile predictions from a regression model and adjusting them using nonconformity scores on a calibration set (itself a subset of training data), ensuring that the resulting intervals achieve the desired coverage while remaining adaptive to the distribution of residuals [23]. For feature attributions on our model's outputs, we opted for the utilization of Shapley values, which means each input feature's contribution to a prediction is quantified by fairly distributing the prediction difference (relative to an approximated baseline) across all possible feature coalitions [1]. By realizing our contributive work [24] in the form of a GUI tool, the framework demonstrates how predictive ML can become actionable for engineers who can see not only the "what" a data-driven model of their device predicts, but also the "why".

The focus of our inquiry then shifts to the underlying system hardware itself. Industrial processing unit performance is shaped not just by the tasks to be run, but also by low-level interactions between various microarchitectural components. One significant aspect of these components is the processor cache.

Traditional microarchitecture simulators, though in many practical examples accurate or cycle-accurate[1], are often prohibitively slow and demand deep architectural expertise to run [26, 27]. As an effort to remedy this difficulty, we explored how program traces, consisting of sequences of x86 instructions, memory access addresses, instruction type information and cache features could be tokenized and modeled with LSTMs [28] to approximate cache miss behavior throughout the execution. We found our approach [29] to generalize across unseen cache configurations and achieve competitive accuracy while running faster than tools like DynamoRIO [30], to which it was compared. The result was a practical middle ground: fast enough to be usable, yet accurate enough to inform design choices.

Finally, the research broadened into the realm of physical simulation and code co-generation. Research in the field of scenario-based testing and writing safety-critical software is increasingly moving in the direction of utilizing code generation models [31]. Large language models (LLMs) promise flexible natural language interfaces, but their outputs are often unreliable, whether due to hallucinations [32] or merely owing to the fact that these models do not appear to actually reason [33]. In collaboration with Volvo Cars, we contribute to *HASCo*, the Hybrid AI Simulation Compiler, which generates executable traffic scenarios from natural language descriptions. To ensure usability, *HASCo* integrates retrieval, validation, and self-repair mechanisms, and introduces the *Effort-to-Usability* (E2U) metric to quantify the human effort needed to refine generated scenarios. This work connects the earlier focus on guarantees and efficiency with the broader challenge of human-centered trust in ML-driven pipelines. Taken together, these contributions form a narrative arc: from simplifying opaque networks, to making predictions accountable, to learning from hardware traces, and finally to enabling generative simulation workflows. Each step is motivated by the same principle: machine learning can support industrial-scale development only if it combines predictive power with guarantees, explainability, and usability. This thesis thus presents not only research-backed methods but also a vision of how trustworthy ML stacks can be realized in practical contexts.

2 Research Summary and Related Work

This section summarizes the research gaps motivating this thesis, formulates the main goal and research questions, and positions the work against the state of the art.

[1]Cycle-accurate simulators are simulators used in computer science to implement and evaluate the performance of scheduling and mapping techniques in microarchitectures with precise cycle-level accuracy[25]. This enables accurate testing and benchmarking throughout the microprocessor design phase without actually building a physical chip.

2.1 Research Gaps and Industry Needs

Based on our analysis of the state-of-the-art in the field of applied ML, we have found a persistent discord between novel modeling techniques and their industrial application. Many methods remain at the "proof-of-concept" stage without being integrated into real-world systems. Sinha et al. [34] highlight the obstacles in deploying AI in industry. Schmitt [35] describes how deep models often face resistance in business settings, and Shahedi et al. [36] document how technically excellent tools fail to achieve adoption when trust or usability is lacking. On the side of industrial stakeholders, there exists a tendency to elude novel ML ideas, based both on the difficulty of interpreting the data-driven model results, a fundamental aspect of ML techniques and an unfortunate consequence of ML's statistical nature, and potential overestimation of what ML techniques can accomplish for their use case. Though a core component of our thesis work has initially been the exploration of ML techniques for solving various aspects of industrially inspired problems, we have also taken serious steps towards the exploration of techniques for alleviation of at least some of the concerns that precede satisfying and efficient ML adoption. This exploration has so far yielded the following intertwined gaps:

Gap_A: Guarantees (and explainability) are under-addressed in ML for industrial-scale systems. A significant number of studies in the sphere of ML applied to systems/operations emphasizes point accuracy and heuristics, with comparatively fewer works that provide statistical or formal guarantees suitable for operational decision-making, and a pairing of those guarantees with explainability such that engineers can act on them [37]. In a relevant work [23], which is a modern introductory tutorial on conformal prediction, Angelopoulos and Bates show distribution-free guarantees for ML outputs but without integration into system-level workflows, and in their work Zhou et al. [38] note scalability challenges and the rarity of pairing output guarantees with practical industrial ML.

Gap_B: Generalizable, fast low-level performance microarchitectural modeling is sparse; simulators are accurate but slow. Traditional simulators (e.g. PIN [39], Valgrind [40], DynamoRIO [30], Q-EMU [41] and gem5 [42]) are accurate but slow and expertise-heavy; ML substitutes that are both fast and generalize across workloads/hardware, though actively in development by researchers in the field, remain rare and often architecture-tied [43, 44, 45, 46]. In their work, [47] Renda et al. particularly pointed out how CPU simulators are often not unified generalizable mechanisms but rather abstraction of at most several composed or individual processor design concepts.

Gap_C: End-to-end, usable integration (from data to runnable artifacts and decisions) is scarcely treated. Many works address individual components: uncertainty quantification, explainability, performance modeling, or robustness, yet few combine all those pieces in operational pipelines evaluated under the full suite of constraints (latency, resource limits, human interpretability, deployment constraints, drift, etc.) [48, 49, 50]. Even where individual components exist (uncertainty quantification, attribution, or ML-based performance models), integrated pipelines that

- start from the kinds of inputs engineers actually have,
- end in runnable artifacts and human-usable outputs and
- are evaluated as such

are limited [51]. Liang et al. [52] show in a novel approach how LLMs may be utilized for robotics control purposes, however with domain limitations. Huang et al. [53] show that the translation from natural language to actions but without full integration into human workflows. Nouri et al. [54] showcase the potential of simulation-guided LLM code generation for use-case-specific software with expert validation, while Xiao et al. [55] showcase a multi-level generative framework, illustrating modularity but lacking full end-to-end usability.

The aforementioned gaps are well-documented in the SoTA literature and represent generalized aims for the community of researchers in the applied ML field: improving certifiability of ML components, providing actionable safety margins, enabling rapid performance evaluation, and operationalizing ML in human-centered tools.

2.2 Research Goal

Taken together, the gaps highlight that current research provides valuable isolated contributions, yet falls short of offering a coherent methodology. Addressing this requires a unifying research direction that explicitly targets trustworthiness, automation and usability at scale. This brought us to the formulation of the following research goal (RG):

RG: Design and evaluate an *automation-oriented, trustworthy ML approach* for industrial-scale systems, combining explainability, guarantees, performance awareness, and end-to-end integration into usable workflows.

2.3 Research Questions

The overarching research goal of this thesis is to design automation-oriented, trustworthy ML approaches for industrial-scale systems. To realize this aim, we pose five specific research questions, each motivated by industrial and research challenges identified in Gaps A-C, and addressed by the included works [20, 29, 24] and a proposed research paper in progress.

Identifying feature importance in ML models

Engineers often demand clarity on why a model makes its predictions. Most attribution methods, however, are heuristic: they produce plausible explanations (for example, a ranked list of important inputs without stating how reliable those explanations are). For safety- or cost-sensitive applications, this is insufficient. Explanations must be paired with quantified uncertainty, meaning guarantees or bounds on the influence of inputs, so that engineers know whether a highlighted factor is consistently impactful or only appears so by chance. This motivates the following research question:

RQ_1: How can input features be ranked or discarded in a way that provides both interpretability and formal/statistical guarantees on their influence?

RQ_1 is answered primarily by Paper A via abstraction-based reduction for provable input influence, and secondarily by Paper B via conformal prediction with Shapley attribution in forecasting.

Delivering uncertainty guarantees without breaking the dashboard

Forecasting tools in industry must run fast, be lightweight, and fit into existing dashboards. However, most current methods either provide only point predictions or heuristic uncertainty estimates, leaving decision-makers without statistical guarantees. At the same time, classical approaches that do offer guarantees are often too computationally heavy or opaque for real-time use. A promising direction is conformal prediction, which can deliver distribution-free coverage guarantees, but adapting it to constrained industrial environments raises its own challenges. This leads to the question:

RQ_2: How can forecasting methods be designed to provide valid uncertainty guarantees and interpretable explanations, while remaining lightweight enough for practical deployment in real systems?

RQ_2 is answered primarily via Paper B through utilization of conformally calibrated intervals and Shapley feature analysis for CPU load forecasting, secondarily via Paper C through adopting distributional robustness criteria for performance prediction.

Data-driven performance model that travels well across hardware and workloads

Traditional simulators are accurate but slow, and most learned models perform well only on the archi-

tecture they were trained for. Industrial practice demands models that *travel well*, i.e., provide accurate performance insights across new workloads and hardware configurations, leading to the second research question, as follows:

RQ_3: How can data-driven models of system behavior (e.g., cache performance) be designed to generalize across both workloads and microarchitectures?

RQ_3 is answered primarily via Paper C through LSTM-based cache miss modeling across architectures, and secondarily via Paper B which utilizes principles of robustness and uncertainty awareness.

Judging whether a predictive model is good enough for system design

Average accuracy alone is misleading; what matters in practice is whether predictions capture both the distributional level (the shape and variability of behavior across an execution) and the aggregate level (the totals and averages that determine overall system performance). Engineers need evaluation criteria that align with how systems are actually reasoned about, leading to the following question:

RQ_4: How should data-driven performance models be evaluated to ensure fidelity?

RQ_4 is covered primarily by Paper C through dual evaluation: subsequence fidelity and aggregate forecasting error and secondarily by Paper D, through its extended evaluation with Effort-to-Usability for automation pipelines.

Turning natural-language reports into simulations, not engineers into editors

Safety and incident reports are often written in plain text, while simulations demand precise formal inputs. The bottleneck is the *human repair effort* needed to bridge the gap. Industrial adoption depends on automation that minimizes this effort while producing trustworthy artifacts, consequently leading to the next research question:

RQ_5: How can natural-language narratives be transformed into runnable simulation scenarios with quantifiable and minimized expert repair effort?

RQ_5 is covered primarily via Paper D, through the HASCo framework with the Effort-to-Usability metric, and secondarily via Paper A through the notion of reusable abstractions for constraints, and Paper C through its data-driven models integrated into pipelines.

The four papers collectively answer these questions, building from principled methods to end-to-end automation:

Paper A (input abstraction for neural networks) addresses **RQ_1** primarily, by offering provable input reduction methods that combine interpretability with guarantees. It also contributes to **RQ_5**, since abstractions can feed into automated constraint handling in scenario generation.

Paper B (uncertainty-aware CPU forecasting) addresses **RQ_2** primarily, delivering practically usable forecasts that combine coverage validity with interpretability. It also contributes to **RQ_1** and **RQ_3**.
Paper C (cache miss prediction with LSTMs) addresses **RQ_3** and **RQ_4** primarily, by developing generalizable data-driven performance models and introducing evaluation criteria for both subsequences and totals. It also supports **RQ_2** secondarily by adopting uncertainty and robustness principles.
Paper D (HASCo) addresses **RQ_5** primarily, demonstrating how natural-language narratives can be compiled into runnable simulations with minimized repair effort, measured via the Effort-to-Usability metric. It also extends **RQ_4** and integrates insights from Papers A-C into a full industrial pipeline.

Per the above, table 1 summarizes the relationship.

Table 1: Mapping of research papers to research questions (RQ)

Paper	**Title**	$\mathbf{RQ}_1$	$\mathbf{RQ}_2$	$\mathbf{RQ}_3$	$\mathbf{RQ}_4$	$\mathbf{RQ}_5$
A	*Abstraction-Based Reduction of Input Size for Neural Networks*	✓				✓
B	*Conformal Prediction-Based CPU Load Forecasting*	✓	✓	✓		
C	*Machine Learning-Based Cache Miss Prediction*		✓	✓	✓	✓
D	*HASCo: Hybrid AI Simulation Compiler*				✓	✓

2.4 Thesis Contributions

The contributions of this thesis can be understood as a set of independent artifacts, each addressing a different research challenge to trustworthy, automation-oriented machine learning in industrial systems. They are comprised of formal methods for simplifying neural networks, forecasting frameworks and performance models with guarantees, and finally of a compiler that translates natural-language reports into runnable simulation scenarios.

2.4.1 Abstraction-based reduction of neural networks

As a first contribution, we develop a principled mechanism for simplifying neural networks while retaining provable guarantees. We introduce an *abstraction method* [20] that constructs paired over- and under-approximating variants with i-th input removed, NN_i^+ and NN_i^- respectively of a trained neural network model NN with (piecewise) linear activation functions, so that for any input vector $\mathbf{x}$ with its i-th input removed (denoted $\mathbf{x}^{-i}$):

$$NN_i^-(\mathbf{x}^{-i}) \leq NN(\mathbf{x}) \leq NN_i^+(\mathbf{x}^{-i})$$

The over- and under-approximating variants are computed in a symmetrical manner with respect to one another, via the splitting and coloring method described in detail in Paper A. We propose the notion of insignificant inputs, where an i-th input to the NN is deemed insignificant if varying the i-th input maximally within its range does not change the resulting NN output by more than a fixed threshold T, $T \in \mathbb{R}$. With this in mind, we design an algorithm to generate a *difference* neural network for the i-th input denoted NN_i^{diff}, such that

$$NN_i^{diff}(\mathbf{x}) = NN_i^{+}(\mathbf{x}) - NN_i^{-}(\mathbf{x}).$$

The difference network NN_i^{diff} bounds how much the i-th input affects the total output, by taking the difference between the over-approximation and under-approximation for the input vector. We propose and show the framework that generates this network, and subsequently utilizes Marabou [17] (a state-of-the-art SMT-based verifier for neural networks that can formally check properties such as robustness and output bounds) to establish formally if it is the case that the output is lesser than a given limit value. If this is the case, the i-th input is deemed insignificant, and vice-versa.

This makes it possible to safely remove features with negligible influence while preserving system behavior within known bounds. Unlike heuristic feature selection, the approach provides interpretability by pinpointing which inputs matter, and formal assurance by guaranteeing how outputs are affected when inputs are discarded. The outline of this mechanism, delineated in greater detail in Paper A, is shown in fig. 1.

2.4.2 Forecasting with uncertainty and explanations

As the next contribution, we develop an uncertainty-aware CPU load forecasting framework designed to support practical safety assessment in industrial computing platforms. The central idea is that predictions should not only be accurate but also come with quantified uncertainty and an explanation of what drives the results.

At its core, the framework wraps a black-box regression model with conformal prediction, as this provides greater information about the safety of configurations on an industrial platform with respect to CPU load thresholds, as can be seen in fig. 2.

Given a nonconformity score function $s(x, y)$, which measures how "strange" or atypical a candidate label y is for an input x relative to the training data, and a calibration set of size m, prediction intervals are defined as

$$\Gamma^{\alpha}(x) = \{y : s(x, y) \leq q_{1-\alpha}\},$$

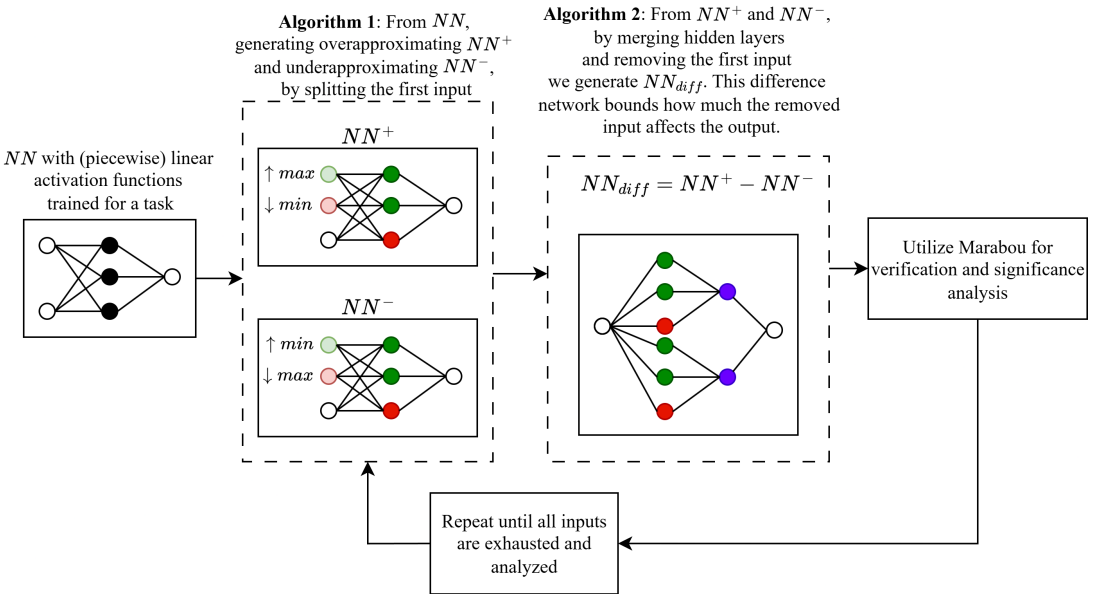

Figure 1: Formal construction of difference networks for abstraction (Paper A).

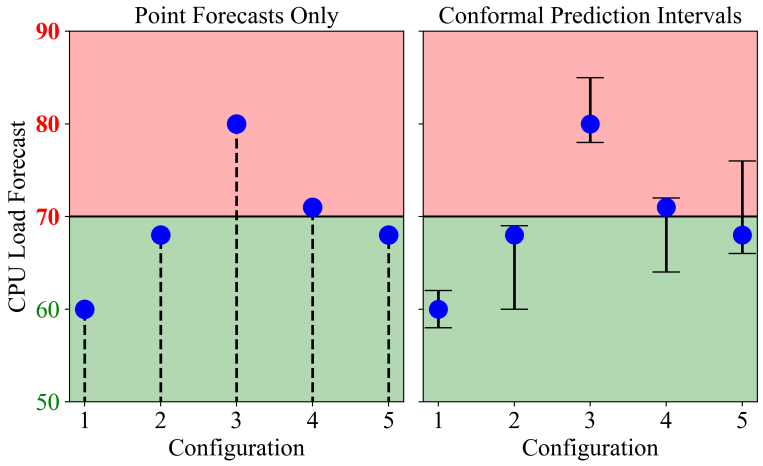

Figure 2: Point forecasting of CPU load versus intervals around predictions generated by conformal prediction. For demonstration, an arbitrary CPU load threshold was set to 70%.

where $q_{1-\alpha}$ is the $(1-\alpha)$ quantile of the calibration scores. This guarantees that, marginally over new samples, the interval covers the true value with probability at least $1-\alpha$, regardless of the distribution of the data. As is widely adopted in contemporary conformal prediction practice, we took the miscoverage rate α to be 0.1, or in other words our goal was 90% coverage. In our implementation, nonconformity

scores were based on a standard choice in regression conformal prediction, the differences between the predicted label values and the nearest quantile value of the calibration samples as per

$$s(x, y) = \max(QR_{lower}(x) - y, y - QR_{upper}(x)),$$

where QR_{lower} estimates the uncalibrated $\alpha/2 = 0.05$ quantile and QR_{upper} estimates the uncalibrated $1 - \alpha/2 = 0.95$ quantile of the residuals. Here, the quantile function returns the threshold value below which a specified proportion of the residuals fall. We then *conformalize* the quantile regressors by expanding both lower and upper *prediction interval* bounds by $\hat{q}$, where $\hat{q} = \text{Quantile}\left(s_1, s_2, ..., s_m; \frac{\lceil (m+1)(1-\alpha) \rceil}{m}\right)$.

To make forecasts actionable, the framework integrates Shapley values, assigning each task a contribution score toward predicted load. If ϕ_i denotes the Shapley value of task i, then the prediction $f(x)$ can be decomposed as

$$f(x) = \phi_0 + \sum_{i=1}^{n} \phi_i,$$

where ϕ_0 is the baseline and ϕ_i represents the marginal effect of including task i. This attribution enables engineers not only to see whether the system is likely to exceed a threshold but also which tasks are most responsible, bridging forecasting with explainability.

We validate the method on real task-trace datasets from industrial computing setups, where scenarios are designed around critical thresholds (e.g., predicting whether load would exceed 80%). Results confirmed that coverage levels matched the nominal rates (e.g., 90% intervals covering in approximately 90% of cases), while average interval widths remain narrow enough for practical use, i.e., within 5% CPU load margin. Importantly, Shapley attributions are consistent across runs, providing stable explanations even in the presence of task interactions. The framework, which is described in greater detail in Paper B, is showcased in fig. 3. To highlight its usability, a prototype GUI was implemented. A screenshot of the GUI may be observed in fig. 4, and a screenshot of the ML output results may be observed at fig. 5.

It displays both the forecast intervals and task-level Shapley scores in a form reminiscent of widely used monitoring tools such as Task Manager or `htop`, but augmented with predictive insight. In this way, the framework demonstrates how statistical guarantees can be made accessible in the daily workflow of engineers.

2.4.3 Data-driven cache miss prediction

Another contribution is the development of data-driven models for cache miss prediction at the microarchitectural level. Traditional cycle-accurate simulators such as gem5 or DynamoRIO cachesim deliver precise measurements of cache behavior, but at the cost of long runtimes and necessity of deep user

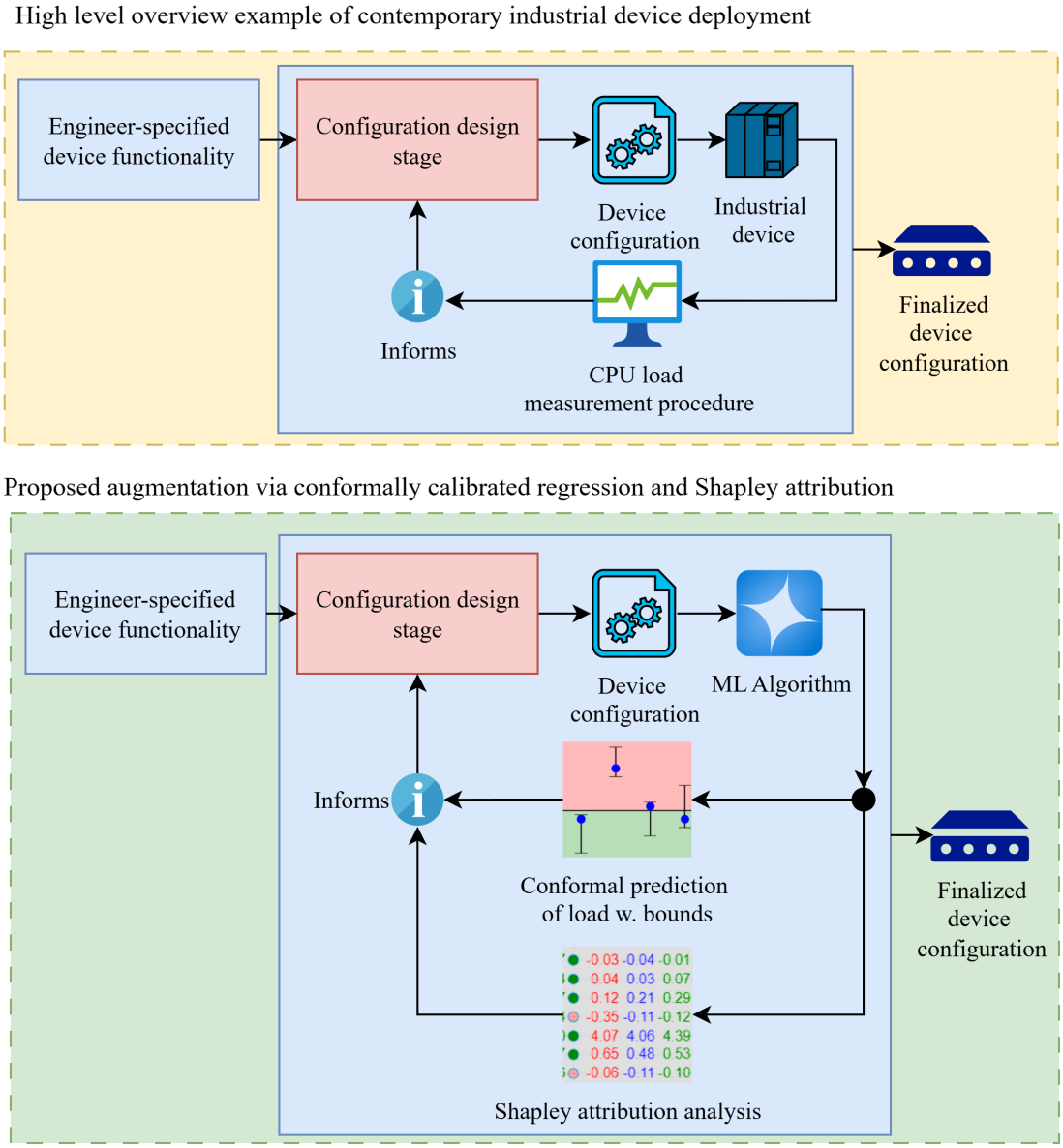

Figure 3: High-level overview contrasting the current state-of-practice (top) with the proposed framework augmentation (bottom). The augmented framework provides not only point forecasts of CPU load but also statistically valid prediction intervals and task-level attributions, enabling engineers to make configuration decisions with both reliability and explainability.

expertise. To provide faster insights, we designed a sequence-learning model that predicts cache miss distributions directly from instruction traces. At a high-level, the distinction of our approach may be

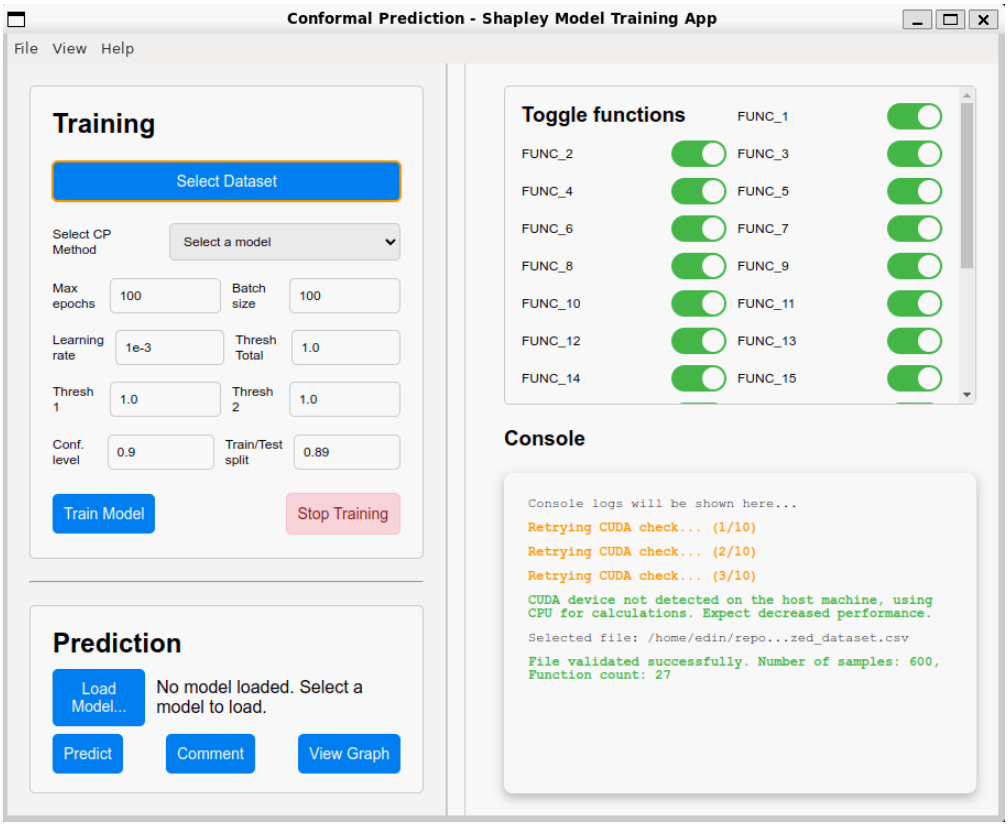

Figure 4: Initial GUI screen - The left hand side is divided into Training and Prediction sections. The Training section uses user input for various general model training parameters, the Prediction section controls the loading of a trained model, annotations and graph window. The right hand side is divided into the Toggle section where tasks are toggled On/Off for prediction and the Console section where various information, warnings and errors are shown to the user.

observed in fig. 6.

Disassembled x86 program traces of the form $(i_1, i_2, \ldots, i_T)$, (T being the length of the trace) are tokenized and mapped to embeddings, which are then processed with stacked LSTM layers. Each line $i_k, k \in [1, T]$ is composed of multiple instruction row features, such as the disassembly string, instruction number, access address delta, program counter address delta and others (more details in Paper C). The network outputs a predicted cache miss distribution tailored to specific cache/core configurations. Here, a cache miss distribution refers to the sequence of miss counts or probabilities observed across program execution, capturing not only the total number of misses but also how they are spread over time or instruction subsequences. This setup is illustrated in fig. 7.

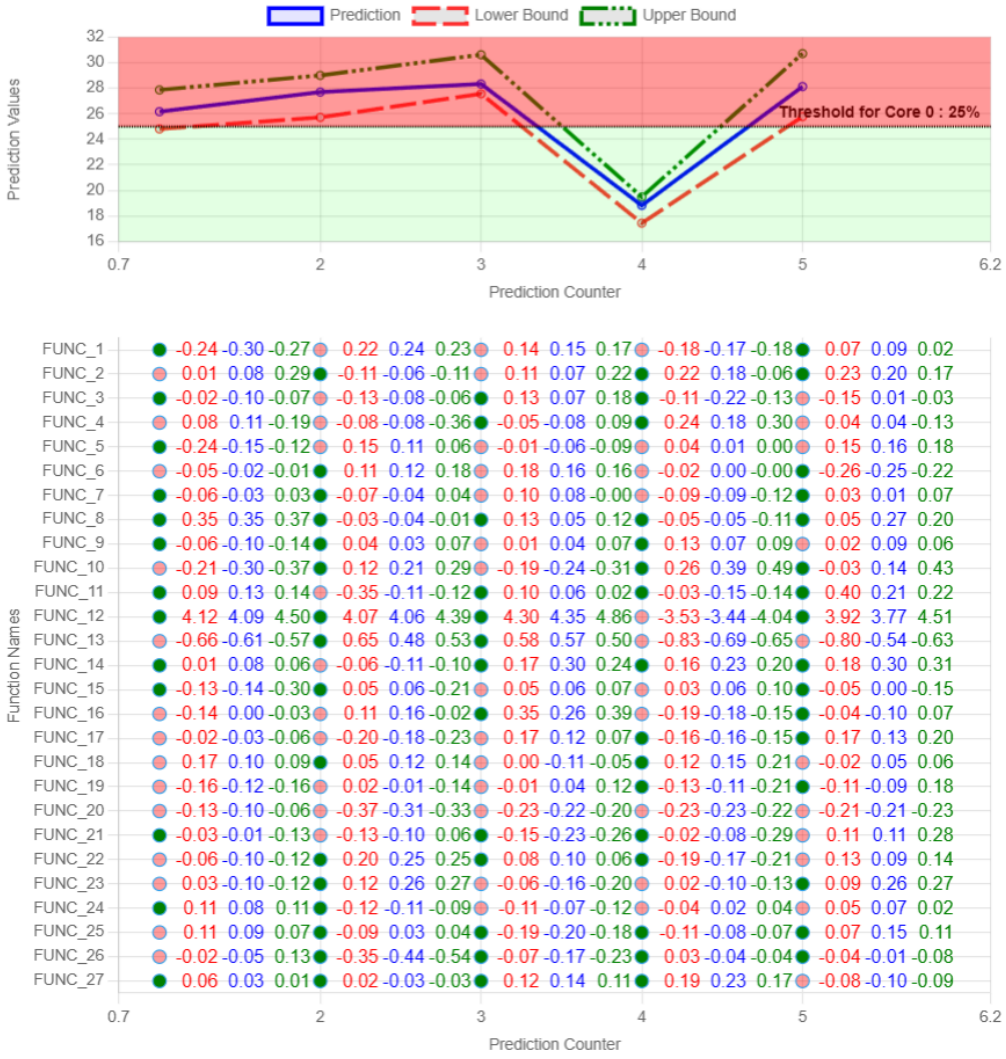

Figure 5: Example screenshot of the CPU load forecaster app with a graph window for the predicted CPU loads for Core 0. Displayed with 5 predictions and an example load threshold of 25%. Active tasks are colored green, while inactive ones are colored red. Individual task contributions are shown as a triplet: contribution to lower bound, contribution to predicted value and contribution to the upper bound.

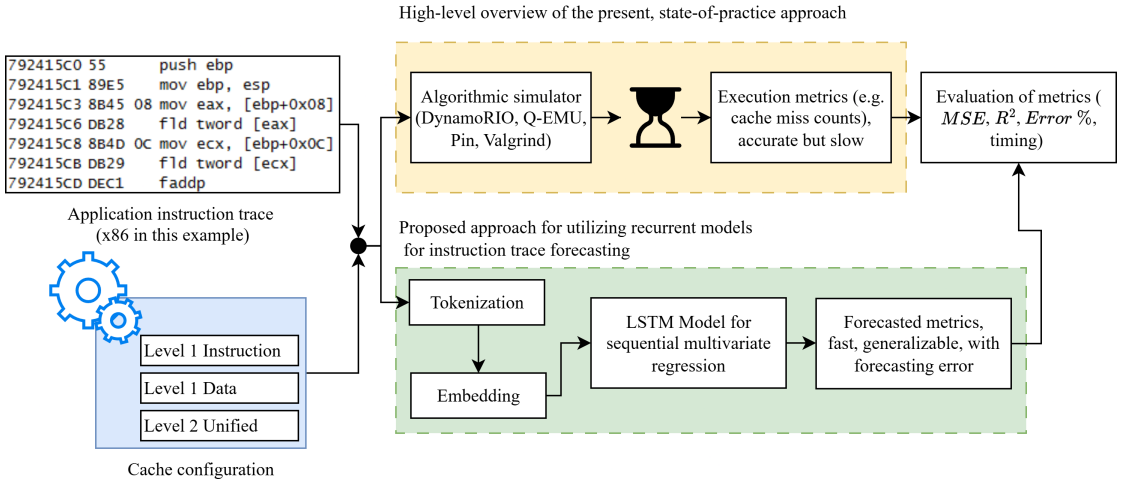

Figure 6: Overview of the formulated approach (Paper C)

To validate our approach in an industrial setting, we propose an evaluation designed around two complementary criteria. First, *distributional fidelity* is assessed using subsequence-level metrics such as MSE (mean squared error), RMSE (root mean squared error, highlighting the average magnitude of deviations), and R^2 (coefficient of determination, indicating how much of the variance in the simulator output is explained by the model). These capture how closely the predicted miss distribution matches simulator outputs over time. Second, we measure *aggregate accuracy*, as the relative error between predicted and simulated total miss counts, reflecting whether the model preserves global system-level behavior. Although in principle perfect distributional fidelity would imply aggregate accuracy, in practice the two can diverge: models may capture the shape of execution traces while missing totals due to systematic bias, or match totals while misrepresenting local dynamics. This dual perspective ensures that models are judged not only on average accuracy but also on whether they faithfully reproduce workload distributions and totals that engineers rely on for design decisions.

We carry out experiments across Sysbench CPU benchmarks and image-processing workloads, validated against cycle-accurate simulators. Results show that the models can reproduce simulator fidelity in many workloads at a fraction of the computational cost, enabling rapid exploration of cache/core configurations. Importantly, the approach also reveals the limits of generalization: while models generalize well across a range of cache sizes and benchmarks, complex workloads such as FileIO expose edge cases where predictions diverge from simulator outputs. These findings emphasize both the promise and the boundaries of applying data-driven performance models in practice.

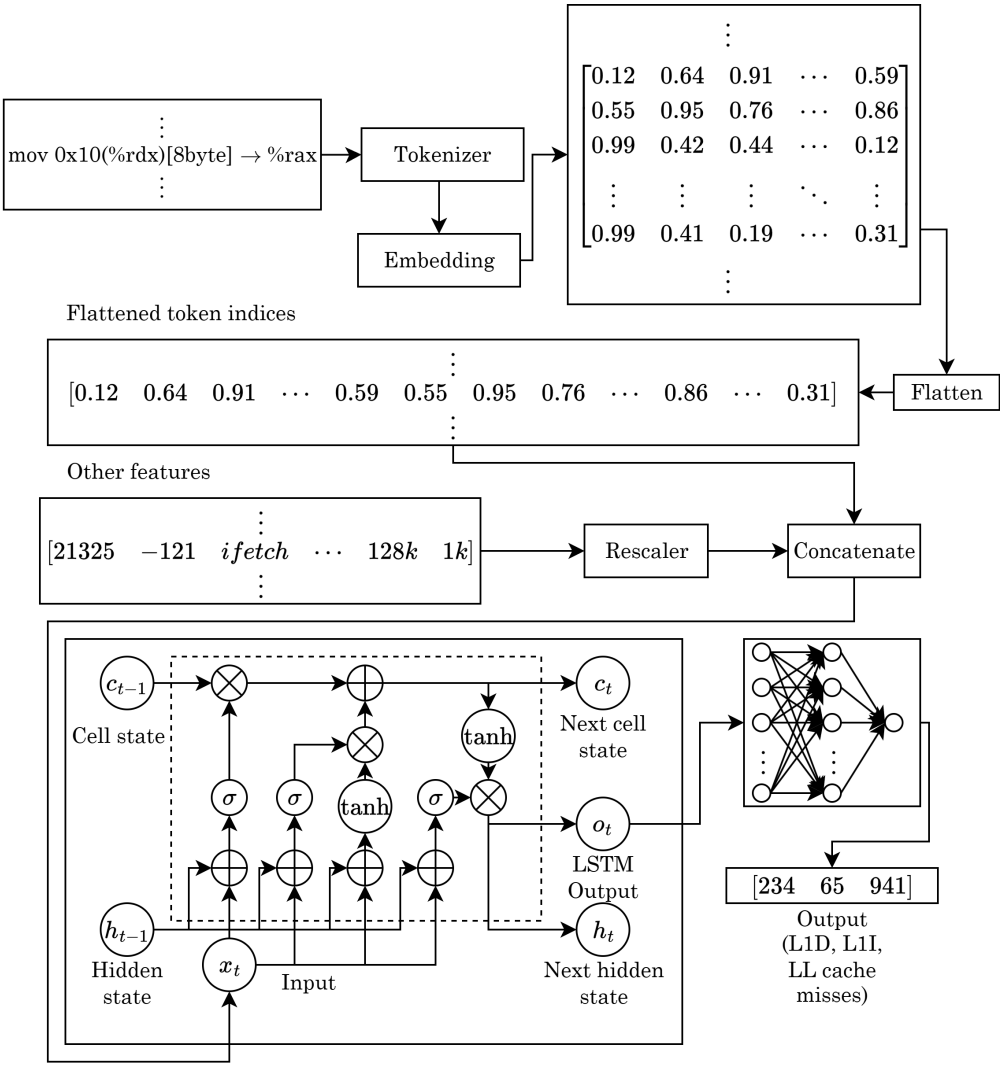

Figure 7: LSTM-based cache miss prediction pipeline (Paper C).

2.4.4 Hybrid AI Simulation Compiler (HASCo)

The final contribution is HASCo, a Hybrid AI Simulation Compiler that transforms natural-language reports into runnable OpenSCENARIO/OpenDRIVE simulations. HASCo integrates retrieval over standards, execution-based validation, and self-repair loops on top of large language models.

To evaluate practical adoption, the *Effort-to-Usability (E2U)* metric is introduced, quantifying the expert repair effort required to make generated scenarios usable. This reframes evaluation from correctness to usability, a decisive factor in safety-critical workflows. A high level overview of the pipeline architec-

ture, of which more details will be found in Paper D, is shown in fig. 8, and the simplified visualization

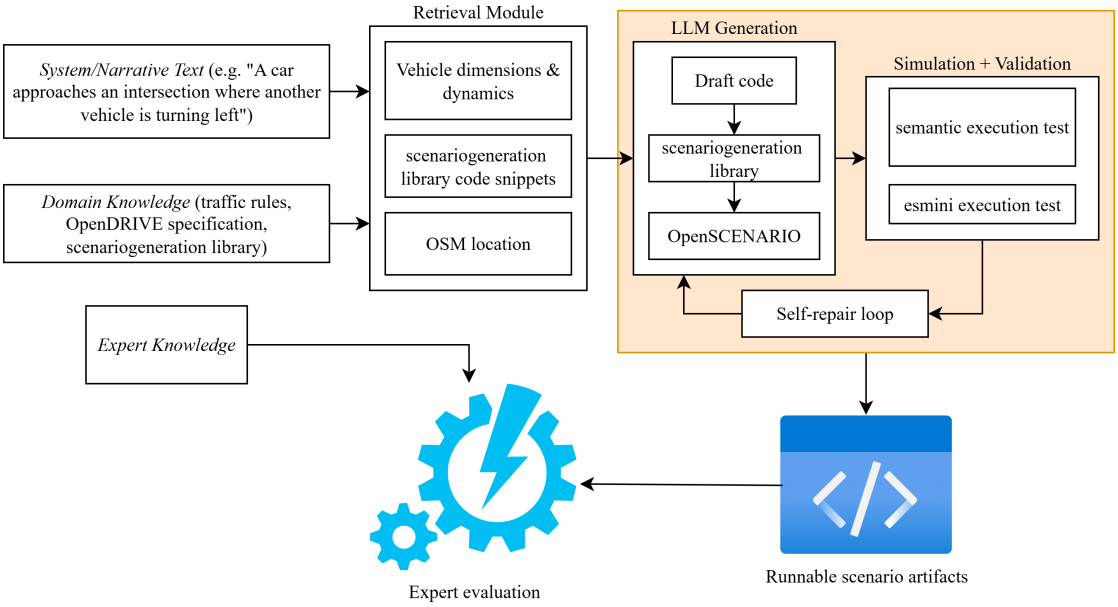

Figure 8: HASCo pipeline architecture (Paper D draft).

used in this thesis is given in fig. 9.

Together, these contributions illustrate four independent but complementary advances: a formal method for neural network abstraction, a forecasting framework with guarantees and explanations, data-driven models for cache prediction with dual evaluation criteria, and a hybrid compiler that automates scenario generation with usability explicitly quantified.

2.5 Related Work

This section reviews relevant literature across areas that form the backdrop of this thesis.

2.5.1 Formal abstraction and dimensionality reduction in neural networks

Work on verification-driven abstraction has developed methods to simplify neural networks while preserving guarantees. For example, Elboher et al. introduce abstraction-refinement for sound verification of deep networks [18], while subsequent tools such as Marabou [17] extend this to bounding outputs under constrained inputs. These efforts differ from purely statistical dimensionality reduction like PCA, which trades guarantees for compactness. Other streams include pruning and quantization in model compression, but abstraction-based methods emphasize property-preserving simplification suited for certification

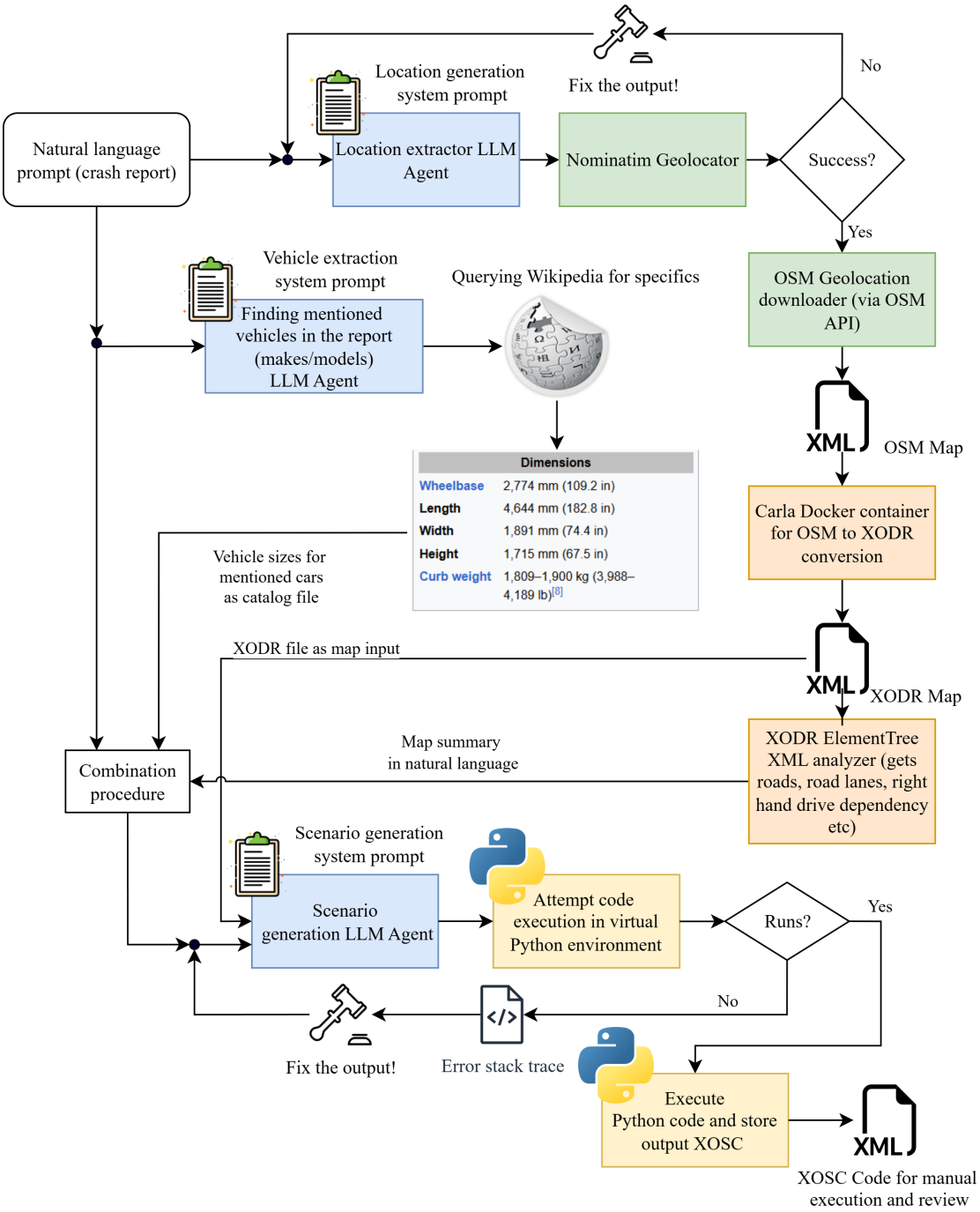

Figure 9: Proposed HASCo workflow

[56]. These efforts show how abstraction can simplify neural networks while preserving guarantees, but most focus on verification rather than performance reduction for downstream ML tasks. Paper A of this thesis extends the principle of property-preserving abstraction to input reduction, introducing difference networks to over- and under-approximate feature importance. This bridges verification concepts with dimensionality reduction needs in certifiable embedded ML.

2.5.2 Uncertainty-Aware Forecasting in Industrial Systems

Forecasting in safety-critical domains requires not just point accuracy but calibrated uncertainty estimates. The framework of conformal prediction (CP) provides rigorous, distribution-free coverage guarantees, originally introduced by Shafer and Vovk [57] and surveyed in modern treatments such as Angelopoulos and Bates [23]. Recent research extends CP with adaptive and conditional variants, e.g., jackknife+ rescaled scores [58] or reweighted nonconformity scores [59], which tighten intervals while retaining coverage. In parallel, industrial load prediction has often relied on queuing-theoretic or regression models (e.g., Hammer et al. for statistical queuing [60]; Ferikoglou et al. for resource-aware scheduling [61]), which provide efficiency but not guarantees. Energy and power systems have recently embraced CP for forecasting tasks such as electricity prices[62, 63], illustrating CP's practical traction in safety-critical resource management. On the interpretability side, Shapley values [64, 65] remain the most widely used attribution mechanism, with recent work exploring their role in explaining predictive uncertainty [66]. Still, integration of uncertainty quantification and interpretability remains underdeveloped in embedded contexts. Some progress has been made in hardware performance prediction with CP-based confidence estimators for DNN accelerators [67], but task-level explainability is rarely addressed. Our work advances this landscape by introducing the first framework that combines conformal prediction with Shapley-based attribution for CPU load forecasting in embedded systems. This approach provides statistically guaranteed coverage while also quantifying the contribution of each task to processor load, bridging the gap between uncertainty calibration and interpretable attribution in safety-critical industrial forecasting.

2.5.3 Machine Learning for Performance Modeling in Computer Architecture

Computer architecture has long relied on cycle-accurate simulators such as gem5 [68] and Q-EMU [41], which offer detailed fidelity but at prohibitive computational cost. To improve scalability, a range of machine-learning-based data-driven alternatives has been proposed. A notable example is Ithemal, which applies LSTM models to predict basic-block throughput directly from x86 assembly, achieving accuracy comparable to Intel's hand-tuned models [45]. Similarly, SimNet uses temporal convolutional networks to approximate microarchitectural behavior and accelerate design-space exploration [44]. More recently,

TAO extends this line of work with transferability across microarchitectures by rethinking latency modeling via learned instruction embeddings [46]. Other strands of work explore cache and memory behavior prediction: Jha et al. [69] demonstrated that neural networks can anticipate cache miss patterns from instruction traces, while Zeng et al. [70] integrated LSTM-based predictors into hardware prefetchers. These works illustrate how sequence learning can complement traditional performance models. Most ML data-driven alternatives either model instruction throughput (e.g., Ithemal) or cache/memory effects tied to a specific microarchitecture (e.g., SimNet, TAO). Paper C extends this landscape by modeling cache miss distributions across architectures, showing that sequence models can generalize beyond single platforms and achieve simulator-level fidelity with significantly improved scalability.

2.5.4 LLMs for Scenario-Based Testing and Code Generation

Scenario-based testing is central in validating autonomous and cyber-physical systems. Zhang et al. [71] and Queiroz et al. [72] outline frameworks for structured scenario generation, while newer works harness LLMs: Zhao et al. propose Chat2Scenario for dataset-derived scenarios [73], Deng et al. present TARGET for LLM-guided translation of traffic rules [74], and ScenicNL [75] enables probabilistic synthesis. Recent frameworks such as TrafficComposer [76], ML-SceGen [55], and ChatSUMO [77] highlight multimodal and controllable scenario generation. Surveys confirm the rapid growth of this space, emphasizing issues such as hallucination, semantic fidelity, and integration with simulators [78, 79]. In parallel, software engineering research has begun to evaluate LLMs in safety-related programming tasks. Nouri et al. [80] catalog common LLM errors and propose sanity-check pipelines, while benchmarks such as SafeBench [81] and SWE-bench [82] provide standardized evaluation. Metrics like CodeBLEU, CodeBERTScore, ICE-Score, and SWE-Judge focus on correctness and syntactic validity, while human-computer interaction studies (e.g., Copilot evaluations [83]) measure repair effort and usability. Together, this body of work underscores both the opportunities and reliability challenges of LLM integration in critical software pipelines. While LLM-based scenario generation frameworks (Chat2Scenario, TARGET, ScenicNL, etc.) demonstrate the promise of text-to-simulation pipelines, they rarely account for the repair cost and trustworthiness of generated outputs. Paper D introduces the concept of Effort-to-Usability (E2U) and implements a hybrid pipeline (HASCo) that measures both automated and human repair effort, directly addressing the usability gap in LLM-assisted scenario generation.

3 Research Methodology

The research presented in this thesis follows a design science methodology [84, 85] complemented by mixed-methods evaluation [86]. The central orientation is toward the creation and validation of artifacts that address both industrially and research motivated problems, following the cycle of *build*, *evaluate*, and *reflect*. This integration of design, evaluation and iterative refinement aligns with what Holz et al. [87] describe as characteristic of computing research methods, whose distinctive trait is blending design, experimentation, and reflection. The procedure spanning the research may be observed in fig. 10. Each

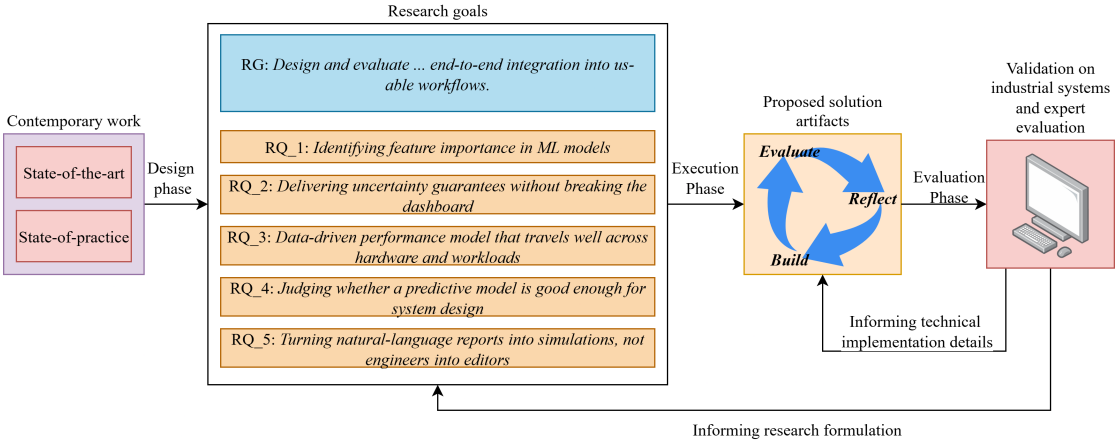

Figure 10: Thesis research process, visualized

artifact was developed either to meet a practical need (e.g., uncertainty-aware forecasting, accelerated performance modeling, or automated scenario generation) or to explore methodological foundations (as in the case of neural network abstraction). All were evaluated against established baselines and refined through iterative feedback. This section presents the methodological arc of the research as it unfolded across the contributions.

3.1 Abstraction to interpretability

The first line of research concerned the simplification of neural networks. The guiding question was whether formal abstraction techniques, long studied in verification, could be applied to input reduction in ways that preserved analyzability. Here, the methodology followed what Wieringa [84] terms *technical action research*: an artifact (the difference network construction) is built, then assessed on benchmark models to test its ability to identify inputs of negligible influence. Evaluation combined formal proof obligations (ensuring $NN_i^-(\mathbf{x}^{-i}) \leq NN(\mathbf{x}) \leq NN_i^+(\mathbf{x}^{-i})$ for all $\mathbf{x}$ etc.) with empirical tests of accu-

racy preservation after reduction. The iterative cycle involved designing abstraction operators, validating them with Marabou (while exploring other similar tools for neural network verification), and refining the reduction procedure to balance tightness of bounds with computational tractability.

3.2 Black-box prediction to actionable forecasts

The second research stream targeted the problem of CPU load forecasting in industrial settings, where operators need predictions that are both reliable and interpretable. Here, the methodological approach combined controlled experimentation with design science. A forecasting pipeline was designed that couples conformal prediction [57, 23] with Shapley-based attribution [64]. Experiments were run on task-trace datasets collected from industrial hardware, with evaluation focusing on empirical coverage, interval width, and threshold exceedance detection. The explanatory component was validated by measuring consistency of Shapley attributions across repeated runs. The iterative process involved adjusting nonconformity functions, calibrating interval levels, and building a prototype GUI to test usability against the familiar idioms of tools such as Task Manager.

3.3 Cycle-accurate simulation to data-driven models

The third strand of work concerned the construction of data-driven models for cache miss prediction. The methodology here reflects what Brown [88] discussed in the form of *design as a search process*: exploring alternative model architectures (LSTMs, embedding strategies, sequence lengths) and evaluating them against ground truth from cycle-accurate simulators. The research was empirical and experimental in nature: models were trained on instruction traces, validated on Sysbench and image-processing workloads, and assessed both for subsequence fidelity (MSE, RMSE, R^2) and aggregate accuracy of total miss counts. Particular attention was given to generalization, by testing models on unseen cache configurations and workloads. The methodological cycle alternated between artifact building (model architecture design), empirical testing, and reflection on failure cases (e.g., FileIO workloads where generalization was limited). This follows Gregor and Hevner's [89] notion of *design evaluation* by means of conducted artificial experiments.

3.4 Component methods to an integrated pipeline

The fourth research stream built on the earlier contributions and asked how they could be operationalized in an end-to-end automation framework. HASCo (Hybrid AI Simulation Compiler) was developed as a novel artifact that integrates large language models, retrieval, validation, and self-repair. The research

approach here aligns with *design science for socio-technical systems* [90], as the pipeline not only generates artifacts (scenarios) but also interfaces with human experts through the Effort-to-Usability (E2U) metric. Evaluation combined objective measures (parse/simulation success, semantic checks) with subjective expert ratings of repair effort, analyzed with inter-rater reliability. The methodology thus blends quantitative metrics with qualitative expert feedback, an example of a *mixed-methods design* [86] tailored to a human-in-the-loop system.

3.5 Cross-cutting methodological principles

The research program combines both methodological exploration and practice-oriented evaluation. Not all contributions were driven by industrial datasets to the same extent: the early work on neural network abstraction (Paper A) was primarily a methodological advance, validated through formal reasoning and benchmark models, while the later contributions (Papers B–D) emphasized industrial traces, workloads, and safety reports. Across this spectrum, four principles consistently guided the research:

- *Iterative refinement:* artifacts were incrementally built and tested, with design adjustments informed by evaluation results.
- *Comparative baselining:* each artifact was evaluated against established alternatives (e.g., heuristic feature selection, non-conformal regressors, cycle-accurate simulators, one-shot LLM generation).
- *Contextual alignment:* where possible, datasets and scenarios were chosen to mirror industrial practice (e.g., processor load traces, instruction-level workloads, natural-language safety reports). For more theoretical contributions such as Paper A, evaluation focused instead on formal guarantees and benchmark networks.
- *Validity management:* threats to validity were addressed through controlled experiments, diversity of workloads, precise metric definitions, and ethical protocols for expert involvement.

In this way, the thesis follows a design science trajectory: constructing artifacts that address identified gaps, validating them through both technical experiments and expert evaluation, and reflecting on their implications for the industrial adoption of ML.

4 Included Papers

All included papers contribute toward the overarching goal of a *trustworthy ML stack for industrial software systems*, with each paper primarily addressing one or more research questions and partially support-

ing the integrative RG.

Personal Contributions: I am the primary author and driver of Papers B-D, and a co-leading contributor to Paper A. Senior co-authors provided ideas, discussions, experimental feedback, and extensive reviews.

4.1 Paper A

Title: Abstraction-Based Reduction of Input Size for Neural Networks.

Authors: Peter Backeman, Edin Jelačić, Tiberiu Seceleanu, Ning Xiong, Cristina Seceleanu.

Status: Published (AISoLA 2023).

Abstract (summary): We extend network abstraction to enable input removal while constructing under/over-approximating surrogates, and show how combined approximations identify low-impact inputs.

Abstract: Machine learning is an increasingly popular method for modeling complex system. A common machine learning model is the neural network. They can be trained to represent complicated functions to a high accuracy. However they often grow large and complex. Recent work is looking in how to abstract networks to yield simpler representation, while retaining some property of the original network, e.g., for every in- put the abstracted networks output is at least as large as the original. In this work, we build on previous ideas and extend it to also consider the input layer. Sometimes, the input vector has a large size, while only a few of the elements are significant in the computation of the output. In this paper, we propose to use a trained neural network model to identify insignificant input elements, i.e, elements which do not contain important information. We show how the presented abstraction method for the input layer can be utilized to achieve this.

4.2 Paper B

Title: A Conformal Prediction-Based Framework for CPU Load Forecasting: A Black-Box Approach.

Authors: Edin Jelačić, Cristina Seceleanu, Peter Backeman, Tiberiu Seceleanu, Axel Jantsch.

Status: Published (IEEE COMPSAC 2025).

Abstract (summary): We propose an uncertainty-aware, threshold-centric CPU load forecasting framework using conformal prediction with Shapley-based explanations to support safety assessment.

Abstract: To address safety concerns in industrial systems, we propose a framework for forecasting CPU load with respect to a predetermined threshold, allowing customers to add tasks from a predefined library. Existing tools, akin to Windows Task Manager, provide limited insights due to their aggregate nature and high computational overhead. Our approach uses conformal prediction for rapid uncertainty-aware

forecasts and Shapley value analysis to quantify individual task contributions to the CPU load. This proof-of-concept framework improves system safety assessment by addressing key research questions in load prediction and validation, paving the way for refined measurement methodologies in industrial applications.

4.3 Paper C

Title: Machine Learning-Based Cache Miss Prediction.

Authors: Edin Jelačić, Cristina Seceleanu, Ning Xiong, Peter Backeman, Sharifeh Yaghoobi, Tiberiu Seceleanu.

Status: Published (STTT, 2025, Special Issue ECBS).

Abstract (summary): We develop an LSTM-based surrogate for cache-miss distributions to approximate algorithmic simulators and analyze feasibility across cache/core configurations.

Abstract: Integrating machine learning into computer architecture simulation offers a new approach to performance analysis, moving away from traditional algorithmic methods. While existing simulators accurately replicate hardware, they often suffer from slow execution, complex documentation, and require deep CPU knowledge, limiting their usability for quick insights. This paper presents a deep learning-based approach for simulating a key CPU component, cache memory. Our model "learns" cache characteristics by observing cache miss distributions, without needing detailed manual modeling. This method accelerates simulations and adapts to different program needs, demonstrating accuracy comparable to traditional simulators. Tested on Sysbench and image processing algorithms, it shows promise for faster, scalable, and hardware-independent simulations.

4.4 Paper D

Title: HASCo: Hybrid AI Simulation Compiler for Trustworthy Scenario Generation.

Authors: Edin Jelačić, Cristina Seceleanu, Peter Backeman, Tiberiu Seceleanu, Rong Gu, Ali Naur, Zhennan Fei, Ammara Asif.

Status: In preparation (target: IEEE Computational Intelligence Magazine (CIM), 2025, SI on Privacy-Preserving Machine and Deep Learning)

Abstract (summary): We present and evaluate an agentic LLM pipeline utilized for generating OpenSCENARIO and OpenDRIVE-based vehicular scenario simulations. The automated pipeline is evaluated via a proposed *Effort-to-Usability* (E2U) rating that captures estimated human repair effort alongside objective checks of the generated scenarios' semantic fidelity.

Abstract: Large language models hold promise for automating code generation, yet their utility in safety-adjacent domains depends less on syntactic correctness and more on whether outputs can be transformed into usable software with minimal human effort. We present HASCo (Hybrid AI Simulation Compiler), an agentic pipeline that generates, validates, and repairs Python code for the scenariogeneration library, a widely used implementation of the OpenSCENARIO/OpenDRIVE standards for road-traffic simulation. HASCo integrates retrieval over specifications and documentation, execution-based validation, and self-repair loops to convert natural-language accident reports into runnable traffic scenarios. To capture practical usability, we introduce Effort-to-Usability (E2U), an expert-rated scale quantifying the human effort needed to repair LLM-generated code into a working simulation, complementing and extending standard metrics such as pass@k, CodeBLEU, or functional test success. We benchmark HASCo across several local and hosted large language models and conduct ablations that isolate the impact of retrieval and self-repair. Results show that HASCo consistently reduces expert-assessed repair effort, with failure modes shifting from syntax and runtime errors toward higher-level semantic fidelity issues such as missing crash triggers or misaligned lane topologies. Hosted models yield lower baseline E2U scores, but local models benefit most from the hybrid pipeline, underscoring the importance of agentic design. By combining code execution feedback with expert usability ratings, our work contributes a reproducible framework for evaluating LLM-generated software not only on correctness but on the human repair effort saved, a dimension essential for trustworthy adoption of agentic AI in software engineering.

5 Progress and Timeline

This section presents an overview of the overall progress with respect to the required courses and publications for the licentiate degree, along with planned activities until the licentiate seminar.

5.1 Included Papers' Status

The status of each paper presented in section 4 is shown in table 2.

5.2 Courses

The required credits of courses for the licentiate degree are 45 ECTS. Table 3 provides an overview of the completed and ongoing courses required for the thesis defense.

Table 2: Status of the included papers

	Status
Paper A	Published in AISoLA 2023
Paper B	Published in COMPSAC 2025
Paper C	Published in STTT in April, 2025.
Paper D	In preparation (IEEE CIM, SI on Privacy- Preserving Machine and Deep Learning, 2025)

Table 3: The summary of completed and ongoing courses

Courses	***Credits***	***Status***
Research Methods in Computer Science	7.5	Completed
Intelligent Decision Support Systems	3	Completed
Artificial Intelligence Systems	3	Completed
Machine Learning with Big Data	7.5	Completed
Catching Bugs by Formal Verification	7.5	Completed
AI-based Modeling and Optimisation for Industrial Systems	7.5	To be finished / Fall Semester 2025
Introduction to Qualitative Research Methodologies	4	To be finished / Fall Semester 2025
Communicating Research	2.5	To be finished / Fall Semester 2025
Academic Writing 3	2.5	To be finished / Fall Semester 2025
Total	**28.5**/45	

5.3 Timeline

Three included papers of this thesis have already been published. However, Paper D is to be written and submitted to the targeted venue, IEEE CIM Special Issue on Privacy-Preserving Machine and Deep Learning. The licentiate thesis is to be completed by the start of January 2026 and is planned to be presented by February 2026. Table 4 provides an overview of the planned activities towards the licentiate thesis defense. It contains the activities, the expected starting and finishing date of each activity, respectively, and the duration in weeks.

Table 4: Time plan for the activities until the licentiate seminar

Activities	***Start Date***	***Finish Date***	***Duration***
Writing licentiate proposal	1-September-2025	22-September-2025	3 Weeks
Updating licentiate proposal based on feedback	22-September-2025	28-September-2025	1 Week
Licentiate proposal presentation	2-October-2025	2-October-2025	1 day
Writing of paper D	22-September-2025	24-November-2025	1 Month
Finalization and Submission of Paper D (CIM 2025 SI on Privacy-Preserving Machine and Deep Learning)	24-November-2025	1-December-2025	1 Week
Completion of req. credit for courses	-	20-December-2025	-
Writing of lic. 1st draft	1-December-2025	1-January-2026	1 Month
Updating Lic. based on feedback	1-January-2026	7-January-2026	1 Weeks
Finalizing and submitting camera-ready version of thesis	-	End of January 2026	1 Week
Licentiate seminar	Beginning of February 2026	Beginning of February 2026	1 day

6 Thesis Outline

This licentiate thesis will be a collection of four research papers, as discussed in section 4. The proposed outline of the thesis is as follows:

Part I: Thesis

1. **Introduction**

   Provides an overview of the industrial motivation, research problem, and scope of the thesis. It summarizes the challenges of applying ML in industrial-scale systems and introduces the main goal, research questions, and overall contributions.

2. **Background**

   Presents fundamental concepts and technical foundations relevant to the thesis:

   2.1 Neural networks and abstraction techniques.

2.2 Forecasting and uncertainty quantification (conformal prediction, explainability).

2.3 Computer architecture and performance modeling.

2.4 Scenario-based testing and large language models.

2.5 Summary of thesis contributions.

3. **Related Work**

   Reviews the state of the art in four areas most connected to the contributions: (i) formal abstraction and dimensionality reduction, (ii) uncertainty-aware forecasting, (iii) ML for performance modeling in computer architecture, and (iv) LLMs for scenario-based testing and code generation.

4. **Research Methodology**

   Describes the design science and mixed-methods approach. Includes method synthesis and artifact design, datasets and baselines, evaluation metrics, and threats to validity.

5. **Contributions and Research Results**

   Summarizes the contributions of Papers A–D with respect to the research questions. Includes visualizations of each contribution and a mapping between papers and RQs.

6. **Conclusions and Future Work**

   Concludes the licentiate thesis, reflecting on lessons learned, open challenges, and future research directions toward a doctoral dissertation.

Part II: Included Papers

1. **Paper A:** Abstraction-Based Reduction of Input Size for Neural Networks.
2. **Paper B:** A Conformal Prediction-Based Framework for CPU Load Forecasting: A Black-Box Approach.
3. **Paper C:** Machine Learning-Based Cache Miss Prediction.
4. **Paper D:** HASCo: Hybrid AI Simulation Compiler for Trustworthy Scenario Generation.

7 Third Cycle Outcome Overview

Table 5: Third Cycle Outcome Overview.

#	THIRD CYCLE OUTCOME	Done (Y/N)
	Knowledge and understanding	**Completed**
1.1	Demonstrate knowledge and understanding of the research area including current specialist knowledge in a defined part of the field, and a deeper knowledge of scientific methods in general and the specific research area in particular.	**YES**
	Motivations ***1.1a Knowledge and understanding of the research area*** • Completed several graduate-level courses, including "ML with Big Data", "Modeling and Simulation", and "Research Methodology for CS&E". • Participated in academic seminars and defenses including: - PhD Defense of Zenepe Satka (12 May 2025) - PhD Defense of Iliar Rabet (14 January 2025) - PhD Defense of Daniel Bujosa (17 January 2025) - Licentiate Defense of Aldin Beriša (14 November 2024) - VeriDevOps and FMAES group meetings (e.g. 5 May, 13 May 2025) - Weekly PerFlex meetings - Thesis presentations and academic project meetings ***1.1b Specialist knowledge in a defined part of the research area*** • Authored/co-authored four research papers on: neural network abstraction, CPU load forecasting with conformal prediction, cache miss simulation with LSTMs and simulation generation via agentic LLMs . • Performed literature reviews on conformal prediction, explainability (SHAP), and industrial-level ML tools. ***1.1c Deep knowledge of research methods*** • Participated in research methodology courses and used methods such as ablation analysis, benchmarking, conformal prediction theory, and formal verification (e.g., Marabou usage on 30 Oct 2024).	

	Skills and abilities	**Completed**
2.1	Demonstrate the ability to critically, independently and creatively and with scientific accuracy identify and formulate issues and to plan and use appropriate methods to undertake a limited research project and other advanced tasks within given time frames and thereby contribute to the development of knowledge and to evaluate this work	**YES**
	Motivations • Designed and executed research plans across multiple domains including neural abstraction (Oct 2024), CPU forecasting (Feb-May 2025), and cache simulation (2023-2024). • Published results in international venues (e.g., Springer journal and COMPSAC conference). • Proposed new paper directions, integrating multiple strands of work (e.g., ICAD + conformal forecasting in Apr 2025). • Maintained continuous planning, self-reflection, and writing, as shown in diary (e.g., weekly thesis meetings, proposal development in May 2025).	
2.2	Orally and in writing present and discuss research results with the national and international scientific community, and the society in general	**YES**
	Motivations • Presented work at project meetings with PerFlex and FMAES groups. • Held internal and external presentations, including for Hitachi (Jan-Feb 2025) and Ericsson (Dec 2024-Jan 2025). • Published and presented papers at COMPSAC and Springer venues. • Wrote and iterated multiple paper submissions as reflected throughout diary (e.g., March-May 2025).	
2.3	Independently conduct research and development	**YES**

	Motivations • Independently implemented full data pipelines (e.g., cache trace collection using DynamoRIO, model training with PyTorch). • Proposed novel extensions and hypotheses (e.g., transformer-based anomaly forecasting, Hamming distance weighting). • Integrated GUI development into ML forecasting apps. • Built summarization app and scheduling systems (e.g., March-April 2025).	

	Judgement and approach	**Completed**
3.1	Demonstrate ability to make ethical assessments in own research	**YES**
	Motivations • Took graduate-level courses covering research ethics and methodology. • Ensured GDPR-compliant handling of log/trace data. • Discussed responsible use of explainable AI in safety-critical predictions in publications.	
3.2	Demonstrate understanding of science's role and use in society, including its possibilities and limitations, and responsibility of its use	**YES**
	Motivations • Participated in collaborative projects with industrial partners (Hitachi, Ericsson, Volvo). • Designed explainable models to enable responsible decision-making in industrial-scale contexts. • Reflected on societal relevance through seminars and conference participation.	
3.3	Identify one's need for further knowledge and take responsibility for one's learning	**YES**
	Motivations • Actively chose and followed advanced courses (e.g., ML with Big Data, Risk courses, Modeling). • Built custom learning paths in PyTorch, time series analysis, conformal prediction. • Kept personal research diary and logs (as included) to reflect on learning needs.	

References

[1] Benedek Rozemberczki et al. “The Shapley Value in Machine Learning”. en. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*. Thirty-First International Joint Conference on Artificial Intelligence IJCAI-22 (Vienna, Austria). California: International Joint Conferences on Artificial Intelligence Organization, July 23, 2022, pp. 5572–5579. (Visited on 09/26/2025).

[2] Shan Ren et al. “A comprehensive review of big data analytics throughout product lifecycle to support sustainable smart manufacturing: A framework, challenges and future research directions”. en. In: *J. Clean. Prod.* 210 (Feb. 2019), pp. 1343–1365.

[3] Junping Wang et al. “Industrial big data analytics: Challenges, methodologies, and applications”. In: *arXiv [cs.DB]* (July 3, 2018).

[4] J Krauß et al. “Application areas, use cases, and data sets for machine learning and artificial intelligence in production”. en. In: *Lecture Notes in Production Engineering*. Cham: Springer International Publishing, 2023, pp. 504–513. (Visited on 09/05/2025).

[5] Eduardo A Hinojosa-Palafox et al. “An analytics environment architecture for industrial cyber-physical systems Big Data solutions”. en. In: *Sensors (Basel)* 21 (13 June 23, 2021), p. 4282.

[6] Mohd Javaid et al. “An integrated outlook of Cyber–Physical Systems for Industry 4.0: Topical practices, architecture, and applications”. en. In: *Green Technologies and Sustainability* 1 (1 Jan. 2023), p. 100001.

[7] Adib Bin Rashid and M D Ashfakul Karim Kausik. “AI revolutionizing industries worldwide: A comprehensive overview of its diverse applications”. en. In: *Hybrid Adv.* 7 (100277 Dec. 1, 2024), p. 100277. (Visited on 09/02/2025).

[8] Elias Dritsas and Maria Trigka. “Exploring the intersection of machine learning and big data: A survey”. en. In: *Mach. Learn. Knowl. Extr.* 7 (1 Feb. 7, 2025), p. 13.

[9] Francesco Leofante et al. “Automated verification of neural networks: Advances, challenges and perspectives”. In: *arXiv [cs.AI]* (May 24, 2018).

[10] Mit Nanda et al. *Preliminary Findings from AI Implementation Research from Project NANDA: State of AI in Business 2025 Report*. Research rep. Accessed: 2025-09-02. Reviewed by Pradyumna Chari, Project NANDA. Based on a multi-method research design including systematic review, structured interviews, and surveys. Project NANDA, July 2025.

[11] Kevin Gurney. *An Introduction to Neural Networks*. London, England: CRC Press, Apr. 21, 2014. 234 pp.

[12] Vanessa Buhrmester, David Münch, and Michael Arens. "Analysis of explainers of black box Deep Neural Networks for Computer Vision: A survey". en. In: *Mach. Learn. Knowl. Extr.* 3 (4 Dec. 8, 2021), pp. 966–989.

[13] Heimo Muller et al. "The ten commandments of ethical medical AI". In: *Computer (Long Beach Calif.)* 54 (7 July 1, 2021), pp. 119–123.

[14] Bryce Goodman and Seth Flaxman. "European Union regulations on algorithmic decision-making and a "right to explanation"". In: *AI Mag.* 38 (3 2017), pp. 50–57.

[15] Changliu Liu et al. "Algorithms for verifying deep neural networks". In: *Found. trends® optim.* 4 (3-4 2021), pp. 244–404.

[16] Ekaterina Stroeva and Aleksey Tonkikh. "Methods for Formal Verification of Artificial Neural Networks: A Review of Existing Approaches". ru. In: *International Journal of Open Information Technologies* 10 (10 Sept. 29, 2022), pp. 21–29. (Visited on 09/18/2025).

[17] Haoze Wu et al. "Marabou 2.0: A versatile formal analyzer of neural networks". In: *arXiv [cs.AI]* (Jan. 25, 2024).

[18] Yizhak Yisrael Elboher, Justin Gottschlich, and Guy Katz. "An abstraction-based framework for neural network verification". en. In: *Computer Aided Verification*. Lecture notes in computer science. Cham: Springer International Publishing, 2020, pp. 43–65. (Visited on 09/05/2025).

[19] Edmund Clarke et al. "Counterexample-guided abstraction refinement". In: *Lecture Notes in Computer Science*. Lecture notes in computer science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 154–169.

[20] Peter Backeman et al. *Abstraction-based Reduction of Input Size for Neural Networks*.

[21] Shiva Nejati et al. "Modeling and analysis of CPU usage in safety-critical embedded systems to support stress testing". In: *Model Driven Engineering Languages and Systems*. Lecture notes in computer science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 759–775.

[22] John Regehr and Usit Duongsaa. "Preventing interrupt overload". en. In: *SIGPLAN Not.* 40 (7 July 12, 2005), pp. 50–58.

[23] Anastasios N Angelopoulos and Stephen Bates. "A gentle introduction to conformal prediction and distribution-free uncertainty quantification". In: *arXiv [cs.LG]* (July 15, 2021).

[24] Edin Jelačić et al. “A conformal prediction-based framework for CPU load forecasting: A black-box approach”. en. In: *2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC)*. 2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC) (Toronto, ON, Canada). IEEE, July 8, 2025, pp. 361–370. (Visited on 08/28/2025).

[25] Jean Luc Béchennec. *Cycle-Accurate Simulator*. [Online; accessed 2025-09-11]. (Visited on 09/18/2025).

[26] Shang Li et al. “Rethinking cycle accurate DRAM simulation”. In: *Proceedings of the International Symposium on Memory Systems*. MEMSYS ’19: The International Symposium on Memory Systems (Washington District of Columbia USA). New York, NY, USA: ACM, Sept. 30, 2019.

[27] Vincent M Weaver and S Mckee. “Are cycle accurate simulations a waste of time?” In: (2008).

[28] Ralf C Staudemeyer and Eric Rothstein Morris. “Understanding LSTM – a tutorial into long Short-Term Memory Recurrent Neural Networks”. In: *arXiv [cs.NE]* (Sept. 12, 2019).

[29] Edin Jelačić et al. “Machine learning-based cache miss prediction”. en. In: *Int. J. Softw. Tools Technol. Transf.* 27 (1 Feb. 22, 2025), pp. 53–80. (Visited on 04/23/2025).

[30] Derek Bruening, Qin Zhao, and Saman Amarasinghe. “Transparent dynamic instrumentation”. In: *Proceedings of the 8th ACM SIGPLAN/SIGOPS conference on Virtual Execution Environments*. VEE ’12: ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (London England, UK). New York, NY, USA: ACM, Mar. 3, 2012.

[31] Mingxing Liu et al. “An empirical study of the code generation of safety-critical software using LLMs”. en. In: *Appl. Sci. (Basel)* 14 (3 Jan. 26, 2024), p. 1046.

[32] Ziwei Ji et al. “Towards mitigating LLM hallucination via self reflection”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Findings of the Association for Computational Linguistics: EMNLP 2023 (Singapore). Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Stroudsburg, PA, USA: Association for Computational Linguistics, 2023, pp. 1827–1843.

[33] Parshin Shojaee et al. “The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity”. In: *arXiv [cs.AI]* (July 18, 2025).

[34] Sudhi Sinha and Young M Lee. “Challenges with developing and deploying AI models and applications in industrial systems”. en. In: *Discov. Artif. Intell.* 4 (1 Aug. 16, 2024), pp. 1–19. (Visited on 09/25/2025).

[35] Marc Schmitt. “Deep learning in business analytics: A clash of expectations and reality”. In: *arXiv [cs.LG]* (May 19, 2022). (Visited on 09/25/2025).

[36] Kaveh Shahedi et al. “From technical excellence to practical adoption: Lessons learned building an ML-enhanced trace analysis tool”. In: *arXiv [cs.SE]* (Aug. 2, 2025). (Visited on 09/25/2025).

[37] Johannes Lederer. “Statistical guarantees for sparse deep learning”. en. In: *Adv. Stat. Anal.* 108 (2 June 2024), pp. 231–258. (Visited on 09/18/2025).

[38] Xiaofan Zhou et al. “Conformal prediction: A data perspective”. en. In: *ACM Comput. Surv.* 58 (2 Jan. 31, 2026), pp. 1–37.

[39] Chi-Keung Luk et al. “Pin: building customized program analysis tools with dynamic instrumentation”. en. In: *SIGPLAN Not.* 40 (6 June 12, 2005), pp. 190–200.

[40] Nicholas Nethercote and Julian Seward. “Valgrind: a framework for heavyweight dynamic binary instrumentation”. en. In: *SIGPLAN Not.* 42 (6 June 10, 2007), pp. 89–100.

[41] Fabrice Bellard. “QEMU, a fast and portable dynamic translator”. In: *USENIX annual technical conference, FREENIX Track* (Apr. 10, 2005), pp. 41–46.

[42] Nathan Binkert et al. “The gem5 simulator”. en. In: *Comput. Archit. News* 39 (2 May 31, 2011), pp. 1–7.

[43] Lingda Li, Thomas Flynn, and Adolfy Hoisie. “Learning generalizable program and architecture representations for performance modeling”. en. In: *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*. SC24: International Conference for High Performance Computing, Networking, Storage and Analysis (Atlanta, GA, USA). IEEE, Nov. 17, 2024, pp. 1–15. (Visited on 09/10/2025).

[44] Lingda Li et al. “SimNet: Accurate and high-performance computer architecture simulation using deep learning”. en. In: *Proc. ACM Meas. Anal. Comput. Syst.* 6 (2 May 26, 2022), pp. 1–24. (Visited on 09/10/2025).

[45] Charith Mendis et al. “Ithemal: Accurate, portable and fast basic block throughput estimation using deep neural networks”. In: *arXiv [cs.DC]* (Aug. 20, 2018).

[46] Santosh Pandey, Amir Yazdanbakhsh, and Hang Liu. “TAO: Re-thinking DL-based microarchitecture simulation”. en. In: *Proc. ACM Meas. Anal. Comput. Syst.* 8 (2 May 21, 2024), pp. 1–25.

[47] Alex Renda et al. “DiffTune: Optimizing CPU simulator parameters with learned differentiable surrogates”. In: *arXiv [cs.LG]* (Oct. 8, 2020). (Visited on 09/10/2025).

[48] Pulei Xiong et al. “Towards a robust and trustworthy Machine Learning system development: An engineering perspective”. In: *arXiv [cs.LG]* (Jan. 8, 2021). (Visited on 09/18/2025).

[49] Jakob Gawlikowski et al. “A survey of uncertainty in deep neural networks”. en. In: *Artif. Intell. Rev.* 56 (S1 Oct. 29, 2023), pp. 1513–1589. (Visited on 09/18/2025).

[50] Rob Ashmore, Radu Calinescu, and Colin Paterson. “Assuring the machine learning lifecycle: Desiderata, methods, and challenges”. In: *arXiv [cs.LG]* (May 10, 2019). (Visited on 09/18/2025).

[51] Nijat Mehdiyev, Maxim Majlatow, and Peter Fettke. “Quantifying and explaining machine learning uncertainty in predictive process monitoring: an operations research perspective”. en. In: *Ann. Oper. Res.* 347 (2 Apr. 2025), pp. 991–1030. (Visited on 09/18/2025).

[52] Jacky Liang et al. “Code as policies: Language model programs for embodied control”. In: *arXiv [cs.RO]* (Sept. 16, 2022). (Visited on 09/10/2025).

[53] Wenlong Huang et al. “Language models as zero-shot planners: Extracting actionable knowledge for embodied agents”. In: *arXiv [cs.LG]* (Jan. 18, 2022). (Visited on 09/10/2025).

[54] Ali Nouri et al. “On simulation-guided LLM-based code generation for safe autonomous driving software”. en. In: *arXiv [cs.SE]* (Apr. 2, 2025). (Visited on 09/10/2025).

[55] Yicheng Xiao, Yangyang Sun, and Yicheng Lin. “ML-SceGen: A Multi-level Scenario Generation Framework”. en. In: *arXiv [cs.AI]* (Jan. 18, 2025). (Visited on 09/10/2025).

[56] Yizhak Yisrael Elboher et al. “Abstraction-based proof production in formal verification of neural networks”. In: *arXiv [cs.LO]* (June 11, 2025). (Visited on 09/17/2025).

[57] Glenn Shafer and Vladimir Vovk. “A tutorial on conformal prediction”. In: *arXiv [cs.LG]* (June 21, 2007).

[58] Rina Foygel Barber et al. “Predictive inference with the jackknife+”. In: *arXiv [stat.ME]* (May 8, 2019).

[59] Salim I Amoukou and Nicolas J B Brunel. “Adaptive Conformal Prediction by reweighting non-conformity score”. In: *arXiv [stat.ML]* (Mar. 22, 2023).

[60] Hugo Lewi Hammer et al. “A queue model for reliable forecasting of future CPU consumption”. en. In: *Mob. Netw. Appl.* 23 (4 Aug. 2018), pp. 840–853. (Visited on 09/17/2025).

[61] Aggelos Ferikoglou et al. *Resource aware GPU scheduling in Kubernetes infrastructure*. Mar. 2, 2021. (Visited on 09/17/2025).

[62] Ciaran O’Connor et al. “Conformal Prediction for electricity price forecasting in the day-ahead and real-time balancing market”. en. In: *Energy and AI* 21 (100571 Sept. 1, 2025), p. 100571. (Visited on 09/18/2025).

[63] Jorge De la Torre et al. “Electricity price forecast in wholesale markets using conformal prediction: Case study in Mexico”. en. In: *Energy Sci. Eng.* 12 (3 Mar. 2024), pp. 524–540.

[64] Scott Lundberg and Su-In Lee. “A unified approach to interpreting model predictions”. In: *arXiv [cs.AI]* (May 22, 2017). Ed. by I Guyon et al., pp. 4765–4774. (Visited on 09/18/2025).

[65] Hugh Chen et al. “Algorithms to estimate Shapley value feature attributions”. en. In: *Nat. Mach. Intell.* 5 (6 May 22, 2023), pp. 590–601.

[66] David S Watson et al. “Explaining predictive uncertainty with information theoretic Shapley values”. In: *arXiv [stat.ML]* (June 9, 2023).

[67] Matthias Wess et al. “Conformal prediction based confidence for latency estimation of DNN accelerators: A black-box approach”. In: *IEEE Access* 12 (2024), pp. 109847–109860.

[68] Jason Lowe-Power et al. “The gem5 Simulator: Version 20.0+”. In: *arXiv [cs.AR]* (July 6, 2020).

[69] Rishikesh Jha et al. “Cache Miss Rate Predictability via Neural Networks”. In: (2018).

[70] Yuan Zeng and Xiaochen Guo. “Long short term memory based hardware prefetcher: a case study”. In: *Proceedings of the International Symposium on Memory Systems*. MEMSYS 2017: The International Symposium on Memory Systems, 2017 (Alexandria Virginia). New York, NY, USA: ACM, Oct. 2, 2017, pp. 305–311.

[71] Xizhe Zhang, Siddartha Khastgir, and Paul Jennings. “Scenario description language for automated driving systems: A two level abstraction approach”. en. In: *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (Toronto, ON, Canada). IEEE, Oct. 11, 2020, pp. 973–980. (Visited on 06/27/2025).

[72] Rodrigo Queiroz, Thorsten Berger, and Krzysztof Czarnecki. “GeoScenario: An open DSL for autonomous driving scenario representation”. en. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. 2019 IEEE Intelligent Vehicles Symposium (IV) (Paris, France). IEEE, June 2019, pp. 287–294. (Visited on 06/27/2025).

[73] Yongqi Zhao et al. “Chat2Scenario: Scenario extraction from dataset through utilization of large language model”. In: *arXiv [cs.RO]* (Apr. 24, 2024). (Visited on 08/27/2025).

[74] Yao Deng et al. “TARGET: Automated scenario generation from traffic rules for testing autonomous vehicles via validated LLM-guided knowledge extraction”. In: *arXiv [cs.SE]* (May 10, 2023). (Visited on 07/25/2025).

[75] Karim Elmaaroufi et al. “ScenicNL: Generating probabilistic scenario programs from natural language”. In: *arXiv [cs.SE]* (May 3, 2024).

[76] Zhi Tu et al. “Multi-modal traffic scenario generation for autonomous driving system testing”. en. In: *Proc. ACM Softw. Eng.* 2 (FSE June 19, 2025), pp. 1733–1756.

[77] Shuyang Li, Talha Azfar, and Ruimin Ke. “ChatSUMO: Large language model for automating traffic scenario generation in simulation of Urban MObility”. In: *arXiv [cs.HC]* (Aug. 28, 2024).

[78] Yongqi Zhao et al. “A survey on the application of Large Language Models in scenario-based testing of Automated Driving Systems”. In: *arXiv [cs.SE]* (May 22, 2025). (Visited on 07/01/2025).

[79] Muhammad Monjurul Karim et al. “Large Language Models and their applications in roadway safety and mobility enhancement: A comprehensive review”. In: *arXiv [cs.AI]* (May 19, 2025).

[80] Ali Nouri et al. “Large Language Models in code co-generation for safe autonomous vehicles”. In: *arXiv [cs.SE]* (May 26, 2025).

[81] Zonghao Ying et al. “SafeBench: A safety evaluation framework for multimodal Large Language Models”. In: *arXiv [cs.CR]* (Oct. 24, 2024).

[82] Carlos E Jimenez et al. “SWE-bench: Can language models resolve real-world GitHub issues?” In: *arXiv [cs.CL]* (Oct. 10, 2023).

[83] Priyan Vaithilingam, Tianyi Zhang, and Elena L Glassman. “Expectation vs. Experience: Evaluating the usability of code generation tools powered by large language models”. In: *CHI Conference on Human Factors in Computing Systems Extended Abstracts*. CHI ’22: CHI Conference on Human Factors in Computing Systems (New Orleans LA USA). New York, NY, USA: ACM, Apr. 27, 2022.

[84] Roel J Wieringa. *Design Science Methodology for Information Systems and Software Engineering*. en. 2014th ed. Berlin, Germany: Springer, Nov. 20, 2014. 332 pp. (Visited on 09/24/2025).

[85] Hevner et al. “Design Science in Information Systems Research”. en. In: *MIS Q* 28 (1 Mar. 1, 2004), p. 75. (Visited on 09/24/2025).

[86] J W Creswell and J D Creswell. “Research design: Qualitative, quantitative, and mixed methods approaches”. In: (2017).

[87] Hilary J Holz et al. “Research methods in computing: what are they, and how should we teach them?” en. In: *SIGCSE Bull.* 38 (4 Dec. 26, 2006), pp. 96–114. (Visited on 09/26/2025).

[88] Ken Brown. “Generation and search methods in design: Discussion”. en. In: *Advances in Formal Design Methods for CAD*. Boston, MA: Springer US, 1996, pp. 97–103. (Visited on 09/24/2025).

[89] S Gregor and A Hevner. “Positioning and presenting design science research for maximum impact”. In: *MIS Q.* 37 (June 1, 2013), pp. 337–355.

[90] Joan E van Aken. “Design science: Valid knowledge for Socio-technical system design”. en. In: *Design Science: Perspectives from Europe*. Communications in computer and information science. Cham: Springer International Publishing, 2013, pp. 1–13. (Visited on 09/24/2025).