Data-driven Adaptive ML-enabled Edge-cloud System Framework for Safe and Efficient Autonomous Systems

Eunho Cho* and In-Young Ko

Korea Advanced Institute of Science and Technology, South Korea
E-mail: ehcho@kaist.ac.kr; iko@kaist.ac.kr
**Corresponding Author*

Received 01 November 2025; Accepted 15 December 2025

Abstract

Machine learning (ML)-enabled systems like autonomous driving systems (ADSs) face challenges meeting safety and performance requirements in diverse environments, especially in resource-constrained, latency-sensitive edge-cloud settings. These challenges often arise from the ML models' limitations, including poor generalization to unseen conditions. Static ML models often struggle to generalize to unseen scenarios, particularly under the latency and resource constraints of edge-cloud infrastructure. Adaptive algorithms using ML system switching have been proposed, but existing approaches frequently lack generalizability, support for common black-box systems, and effective use of distributed edge-cloud resources. This paper presents a novel adaptive ML-enabled edge-cloud system framework to address these shortcomings. Our framework combines cloud-based pre-runtime analysis, which leverages simulation for behavioral understanding and scenario-to-system mapping, with collaborative edge-cloud runtime adaptation featuring dynamic ML model switching. It supports black-box systems and aims to balance safety and efficiency by utilizing appropriate edge and cloud resources situationally. Preliminary CARLA-based evaluation of the edge runtime component suggests our framework can potentially improve the safety-efficiency

Journal of Web Engineering, Vol. 25_3, 299–324.
doi: 10.13052/jwe1540-9589.2531

trade-off compared to single-model ADSs in some scenarios. Moreover, extensive experiments using the MetaDrive simulator with 100,000 randomized driving scenarios demonstrate that the adaptive system improves safety by 2.6% while doubling computational efficiency compared to a single-model baseline. These results validate the framework's scalability and the feasibility of data-driven scenario–system mapping for adaptive ML-enabled autonomous systems operating across edge and cloud environments.

Keywords: ML-enabled systems, autonomous driving systems, edge-cloud computing, adaptive systems, simulation-based testing.

1 Introduction

Autonomous systems, particularly autonomous driving systems (ADSs), are increasingly integrated within the broader edge-cloud computing ecosystem. Modern ADSs generate substantial sensor data at the edge, selectively offload data to the cloud for large-scale analysis and model refinement, and receive updated models back at the edge. This collaborative edge-cloud architecture aims for safer roads, reduced congestion, and more efficient mobility services [17].

ADSs rely on machine learning (ML) components for perception, decision-making, and control tasks. However, these ML components face challenges due to the unpredictable nature of real-world scenarios, including diverse traffic patterns, unexpected behaviors, and varying environmental conditions [20, 21]. Ensuring reliable performance, especially under edge constraints (resource limits, latency requirements), remains an open challenge.

Frameworks like Autoware [13] and Apollo [1] use modular designs with independently optimized components. Despite such architectures, achieving consistent safety and efficiency across diverse driving scenarios with a fixed set of ML models is difficult. ML models risk being too large for practical computational budgets or too small for overfitting specific scenarios [11]. Static ML systems, therefore, often struggle with adaptability, particularly when facing scenarios unseen during training, highlighting the need for dynamic solutions in edge-cloud environments.

Adaptive approaches using runtime ML system switching have been proposed to mitigate these challenges [8,15]. However, most existing approaches assume white-box accessibility, requiring predictable model behaviors and limiting practical applicability. This limits practical applicability because

many state-of-the-art ML models used in ADSs, particularly deep neural networks, function as complex black boxes, making their internal states difficult to predict or analyze directly for adaptation purposes. Moreover, these methods often don't fully leverage both cloud computational strengths and edge real-time capabilities simultaneously. Thus, frameworks that seamlessly integrate edge-cloud collaboration for effectively handling black-box ML systems under varying conditions are critically needed.

To address the limitations of static models in diverse/unseen scenarios and the applicability constraints of existing white-box adaptive approaches, particularly concerning black-box systems and effective edge-cloud collaboration, we present an adaptive ML-enabled edge-cloud system framework designed to overcome these limitations. Our approach combines a cloud-driven pre-runtime phase (leveraging extensive simulations for behavioral analysis and scenario mapping) with a collaborative runtime phase where edge and cloud systems jointly identify scenarios and dynamically select optimal ML systems. This two-phase strategy maximizes cloud resources for exhaustive pre-runtime analyses and edge speed for real-time adaptability.

In our previous work [7], we conducted a preliminary evaluation of the framework's potential by implementing simple rule-based runtime adaptation, such as 'entering an intersection', based on the CARLA simulation platform.

This paper aims to extend this framework at a large scale and validate its overall effectiveness. To this end, we utilized the MetaDrive simulator to generate 100,000 random driving scenarios. Among these, 10,000 scenarios were used as training data for the 'cloud pre-runtime analysis' phase. In this process, we extracted over 400 scenario features and labeled each scenario with the ADS policy that yielded optimal performance. We then constructed a decision tree model as the 'knowledge base' to select the optimal policy based on these scenario features.

Subsequently, the runtime adaptation evaluation, conducted on 90,000 separate test scenarios, demonstrated that the proposed data-driven adaptive ADS achieved the highest driving score compared to the two fixed ADSs. Concurrently, in terms of efficiency, the system achieved an intermediate processing speed, positioned between two fixed ADSs, demonstrating that the framework successfully manages the trade-off between safety and efficiency.

The main contributions of this study are:

- **An edge-cloud adaptive framework**: An edge-cloud adaptive framework supporting black-box ML systems, defining cloud roles

(pre-runtime analysis) and edge-cloud collaboration roles (runtime adaptation).

- **Implementation and instantiation**: We introduce the first concrete implementation of the 'cloud pre-runtime analysis' and 'scenario-system mapping' phases, which were previously proposed only theoretically in the original paper. This instantiation is realized as a data-driven decision tree model that leverages over 400 scenario features.
- **Large-scale experimental validation**: We conducted a large-scale evaluation using 100,000 randomized MetaDrive scenarios. The results quantitatively demonstrate that the proposed framework successfully manages the trade-off between safety and efficiency in diverse and unseen scenarios. This significantly extends the preliminary evaluation from the original paper and validates the robustness of our approach.
- **Insights and future research directions**: We provide insights and future directions for scenario generation, knowledge base maintenance, and deployment in practical edge-cloud environments.

The remainder of this paper is structured as follows: Section 2 reviews relevant literature. Section 3 introduces our proposed adaptive framework. Section 4 summarizes the preliminary validation from the original paper, which utilized CARLA, and presents in detail the core implementation and the results of the main validation of our framework leveraging MetaDrive. Section 5 discusses implications, limitations, and future research. Finally, Section 6 summarizes our findings.

2 Related Work

The adaptive system framework, employing system or model switching, is critical for ensuring ML system adaptability in dynamic environments. This technique dynamically transitions between models/systems to maintain optimal performance, safety, and quality under varying circumstances.

Several studies have explored adaptive frameworks for ML-enabled systems, showcasing diverse strategies to address dynamic challenges. Amir et al. [5] proposed a predictive control strategy with reconfigurable state-based models, focusing on real-time environments. Noguchi et al. [18] introduced a model selection and management approach using transfer reinforcement learning to improve efficiency in Open IoT environments. Cho et al. [8] developed an anomaly-aware adaptation framework for cyber-physical systems, using reinforcement learning to address anomalies.

Kulkarni et al. [15] proposed a QoS-based model switching framework for ML-enabled systems, dynamically managing performance uncertainties.

However, these valuable studies often have limitations. Many frameworks are constrained to specific environments or predefined scenarios, limiting generalizability. Some depend on accessible system behaviors, making them unsuitable for black-box systems. It is very difficult or impossible to determine in which situation the system behavior will react as desired for many high-performance ML models used in practice. Furthermore, existing works frequently focus on specific aspects like efficiency or anomaly detection, often neglecting the complex interplay with other critical factors such as overall system safety.

Recent research explores edge AI optimization considering resource constraints [24] and novel cloud-edge collaborative architectures [12]. Yet, these often do not directly tackle the specific challenge of dynamically adapting black-box ML models for safety-critical autonomous driving systems. Our work aims to bridge this gap by proposing a framework integrating adaptive black-box ML switching within a structured edge-cloud approach tailored for ADS safety and efficiency.

To address these limitations, our proposed framework introduces three key innovations:

- **Diverse scenarios**: Our framework systematically evaluates the performance of multiple ML systems across a wide range of scenarios during pre-runtime testing, enabling effective adaptation even in previously unseen conditions.
- **Black-box systems**: Our approach supports black-box ML systems, expanding the applicability of adaptive frameworks to these systems. By leveraging simulation-based testing, it becomes suitable for a broader range of real-world applications.
- **Safety and efficiency**: The framework ensures consistent performance across diverse driving contexts by dynamically switching ML systems based on scenario-specific requirements, balancing safety and efficiency.

Building on prior work and addressing key challenges, our framework offers a scalable and robust solution for adaptive ML-enabled systems, enabling improved safety, efficiency, and adaptability in complex autonomous environments. By designing a model-switching framework grounded in simulation-based validation within an edge-cloud perspective, the proposed approach extends beyond conventional QoS-driven white-box methods. It specifically addresses adaptive ML-enabled systems in black-box

settings, overcoming challenges such as limited generalizability and static assumptions in prior methods. This innovation significantly broadens the applicability of adaptive systems to a wider range of real-world scenarios, ensuring their effectiveness in dynamic and unpredictable environments.

3 Adaptive ML-enabled Edge-cloud System Framework

Our framework utilizes an edge-cloud architecture comprising two main phases: a pre-runtime phase executed on cloud resources and a runtime phase operating across edge and cloud infrastructure. This section first discusses the overall approach, followed by detailed examinations of the pre-runtime and runtime phases.

Figure 1 illustrates the framework's flow, divided into two phases. The lower section depicts the cloud-based pre-runtime phase, which involves simulation-based testing, behavioral analysis, and scenario-system mapping to prepare the system knowledge base. This phase evaluates candidate ML systems through extensive simulation, analyzes their behavior to identify optimal operating subspaces, and maps the most suitable system to each subspace within the knowledge base, leveraging cloud computational power.

The upper section shows the runtime phase, based on the distributed MAPE-K loop [14], a standard model for self-adaptive systems. The edge monitors the environment (monitor); edge and cloud collaboratively identify the current scenario subspace (analyze); they collaboratively select the

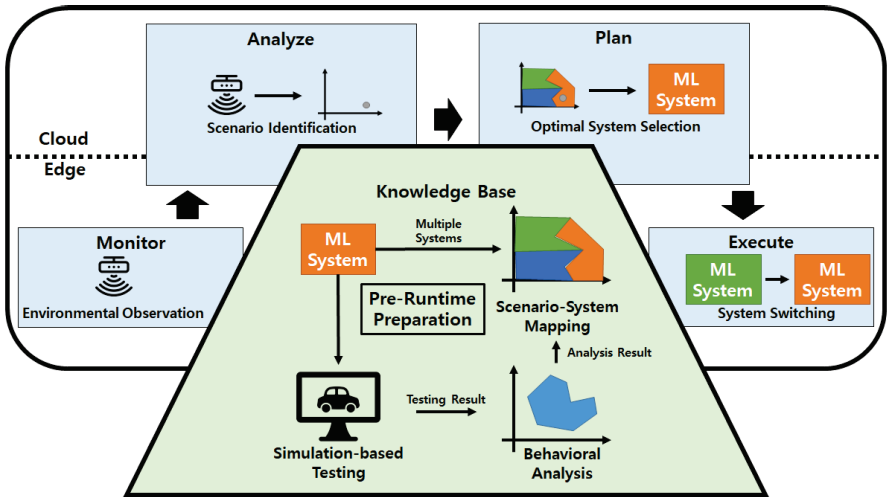

Figure 1 Overall adaptive ML-enabled edge-cloud system framework.

optimal ML system using the knowledge base (plan); and the edge executes the system switch (execute). This combination of cloud pre-runtime preparation and distributed runtime adaptation enables stable and effective adaptation in complex environments.

In this framework, a 'scenario' represents the specific operational context, often captured as a vector encompassing environmental factors such as weather and road type, system goals like waypoints, and current system state variables including speed and position. We assume multiple ML systems exist, each optimized for different scenarios or data distributions. The primary goal of the pre-runtime phase is to analyze system behavior across diverse scenarios via large-scale simulations, identify the scenarios best suited for each system's operation, and systematically compile this information into a knowledge base that informs runtime decision-making.

3.1 Pre-runtime Phase

Executed on the cloud due to computational demands, the pre-runtime phase includes three key steps: simulation-based testing, behavioral analysis, and scenario–system mapping.

3.1.1 Simulation-based testing

First, we evaluate ML system performance using extensive simulations (e.g., using CARLA) across diverse environments. These scenarios, encompassing complexities like varying weather, road networks, and traffic in ADS contexts, enable the collection of performance data for each ML system. Scenario generation prioritizes diversity to cover a broad spectrum of operating conditions, grounding the identification of suitable operating scenarios for each system.

Simulation-based testing is crucial for evaluating ML systems, especially when real-world testing is risky or infeasible. It allows systematic performance evaluation under diverse, controlled conditions. Platforms like CARLA [25] offer realistic physics-based simulations suitable for analyzing safety and efficiency metrics in autonomous systems like autonomous driving systems (ADSs). The scale and computational demands of such simulations are well-suited for cloud execution. Simulation testing characterizes ML system performance. Prior work includes surrogate model-based frameworks [10] and search-based approaches to identify hazard boundaries [23], demonstrating the simulation's ability to define operational limits. In our framework, simulation testing is a key component of the cloud-based pre-runtime phase.

3.1.2 Behavioral analysis

Next, each ML system's behavior is analyzed using the simulation data, focusing on safety and efficiency. For ADSs, safety metrics might include detailed collision types, involving pedestrians or vehicles, for example, off-road excursions, route completion ratio, and compliance with traffic regulations such as traffic signal adherence and stop sign compliance. Efficiency metrics cover computational cost, like average inference time per frame or peak memory usage, and network resource consumption required by each ML system. This analysis assesses whether systems meet safety requirements and satisfy QoS criteria within different scenario subspaces.

3.1.3 Scenario-system mapping

This mapping step is crucial because different ML systems exhibit varying performance trade-offs, notably between safety and efficiency, under different scenarios. Based on the behavioral analysis, this step systematically assigns the best-performing system to each identified scenario subspace according to pre-defined objectives, for instance prioritizing safety over efficiency, creating a reliable decision guide known as the knowledge base for the runtime phase. This involves cloud-based multi-objective optimization, potentially using evolutionary algorithms or rule-based heuristics. The resulting mapping is stored in the knowledge base, possibly as a compact decision tree or a hash map for efficient runtime lookup, and potentially cached at the edge for runtime access, directly enabling the system's adaptive capabilities.

3.2 Runtime Phase

The runtime phase operates using the distributed MAPE-K (monitor, analyze, plan, execute, knowledge) loop across edge and cloud resources. We assume the edge device is equipped with the necessary sensors. Each step leverages edge and cloud strengths to maintain safety and efficiency in complex environments.

3.2.1 Environmental observation

The runtime process begins with observing the environment in real-time using onboard edge sensors (e.g., cameras, LiDAR, radar). This collected data forms the input for the subsequent analysis step.

3.2.2 Scenario identification

Following observation, the current scenario subspace is determined using the observed edge data. A scenario involves initial environment/system states

and operational goals. Observations identify the current state and goals, but unobservable variables may lead to multiple possible scenarios, thus forming a scenario subspace. This analysis is performed collaboratively on edge and cloud. Rapid analysis of immediate sensor data is performed by the edge device for local context identification, potentially using lightweight convolutional neural networks to identify environment, or other cars. The cloud leverages larger datasets or more powerful models for deeper analysis, perhaps employing long short-term memory networks for predicting traffic or accessing aggregated complex mobility data, providing broader context or predictions. The result is an identified scenario subspace incorporating both edge and cloud perspectives.

3.2.3 Optimal system selection

Based on the identified scenario subspace, the most suitable ML system is selected by referencing the knowledge base. This planning step also occurs collaboratively on edge and cloud. A quick selection is made by the edge using the identified local context and the potentially cached knowledge base to address immediate needs. A more sophisticated selection or refinement is performed by the cloud, considering broader goals or complex trade-offs. This decision can potentially update the edge's initial plan; for instance, based on predicted traffic congestion patterns or system-wide energy optimization goals. The final decision on the optimal ML system combines edge and cloud inputs, aiming for optimal safety and efficiency based on both immediate needs and longer-term objectives.

3.2.4 System switching

Finally, the execution step involves switching to the selected ML system on the edge device. This transition must occur in real-time with minimal disruption. The execution requires edge technologies capable of minimizing transition delays and ensuring system stability post-transition. Furthermore, the performance of the newly activated ML system should ideally be monitored, with results potentially reported to the cloud to refine the knowledge base for future decision-making improvement.

4 Investigation

This section validates the effectiveness of the proposed adaptive edge-cloud framework. First, Section 4.1 summarizes the preliminary validation results [7] from our CARLA-based study, which was conducted to ascertain the potential of the runtime adaptation component. This preliminary

evaluation clearly demonstrated the limitations of a simple rule-based approach, providing strong motivation for the main validation presented in Section 4.2. Section 4.2 validates the overall performance and robustness of the framework through large-scale randomized scenarios using the MetaDrive simulator and the implementation of a data-driven 'scenario-system mapping'.

4.1 Evaluation of a Simple Rule-based Approach

This section presents a preliminary evaluation of the runtime adaptation component within our proposed adaptive ML-enabled edge-cloud framework. A simple prototype adaptive ADS assesses practical applicability, focusing on safety and efficiency.

Experiments ran on Ubuntu 22.04 (Intel Xeon 4215R, 3x RTX A5000 GPUs 24GB, 128GB RAM). The replication kit containing the source code used in this study is publicly available at [3].

4.1.1 Experiment design

We implemented a simple adaptive ADS prototype focusing on runtime adaptation at the simulated edge. The prototype uses CARLA [9], our simulated edge environment, and Leaderboard benchmarks [2] to evaluate safety and efficiency across various scenarios. We compared the adaptive mechanism against single ML systems regarding safety and efficiency at the edge.

CARLA [9] is a widely used open-source ADS simulator providing diverse maps and dynamic components. We used CARLA datasets, originally from Transfuser [6] and InterFuser [22] training, for testing. Key test scenarios included 'longest6', '42routes', 'town05_short/long', 'town06_long', and 'town10_short'. The CARLA Leaderboard [2] executes these scenarios under predefined conditions using modules like scenario runner.

Safety metrics from the CARLA Leaderboard included: *Route Score*, representing the percentage of the route completed; *Penalty Score*, reflecting points deducted for incidents where 1 is perfect; and *Composed Score*, providing an overall safety measure. We also introduced *Scenario Dominance Count*, measuring the number of scenarios where an agent achieved the highest composed score. Efficiency was measured by the decision time ratio, calculated as CARLA simulation time divided by real-world time.

The prototype combines Transfuser [6] and the NPC Agent sourced from the CARLA Leaderboard, simulating edge runtime adaptation. Transfuser is a complex DNN agent utilizing camera and LiDAR fusion via self-attention;

it is capable of handling challenging scenarios but computationally heavy for continuous edge execution. The NPC Agent is a simple, computationally efficient rule-based baseline suitable for edge resources but limited in complex situations. The prototype activates Transfuser near intersections, defined as within 50 meters, and uses the NPC Agent otherwise, simulating edge-based scenario identification and system selection. This logic prioritizes safety with Transfuser in higher-risk intersections and efficiency with NPC on simpler road segments, balancing the trade-offs within the edge environment.

The prototype employs a simple rule-based knowledge base, simulating the output of the envisioned cloud pre-runtime analysis. The underlying principle is that Transfuser offers higher safety at greater computational cost, while NPC is efficient but less safe. The mapping reflects this: Transfuser is selected for high-risk intersections prioritizing safety, and NPC for other roads prioritizing efficiency. This allows the prototype to dynamically adapt at the edge, balancing safety and efficiency based on simple scenario detection.

4.1.2 Experiment result

We evaluated the adaptive ADS prototype against the NPC and Transfuser ADS to address safety and efficiency. Transfuser consistently achieved the highest composed scores, while NPC performed poorly, highlighting the inherent trade-off between performance and computational simplicity relevant to edge suitability. Although the adaptive ADS often scored below the pure Transfuser, it demonstrated robustness. Notably, it matched Transfuser's composed score in the 'longest6' scenario and achieved the highest score, indicating dominance, in 21 scenarios overall. This result suggests, concerning RQ1, the potential effectiveness of edge-based adaptation. However, lower scores in specific scenarios, such as 'town06_long,' indicate limitations of the prototype's simple adaptation rule, pointing to the need for refinement through the full edge-cloud framework involving, for example, more sophisticated scenario identification or a richer knowledge base.

Transfuser's consistently low time ratio signifies high computational cost, potentially problematic for resource-constrained edge devices. Conversely, NPC was the most efficient but demonstrated poor safety performance. The adaptive ADS achieved intermediate efficiency. This result indicates that, concerning RQ2, the adaptive approach successfully balanced safety needs with edge resource usage by selectively engaging the computationally heavier Transfuser only when necessary, thus demonstrating a clear efficiency advantage over running Transfuser continuously.

These preliminary results suggest that edge-based adaptation, as demonstrated by the prototype, can offer safety performance comparable to the best-performing single model (Transfuser) in specific scenarios while achieving significantly greater computational efficiency suitable for edge deployment. This indicates the potential of the edge adaptation component within our broader edge-cloud framework to address the core challenge of balancing safety and efficiency. However, the observed limitations underscore the need for future work focusing on the complete framework implementation, including cloud-based pre-runtime analysis and more sophisticated edge-cloud coordination mechanisms to improve the adaptation strategy and achieve robust performance across a wider range of scenarios.

However, this preliminary experiment revealed limitations, emphasizing the importance of cloud-based pre-runtime analysis and careful component selection for edge execution. The NPC Agent's struggles with unexpected events (e.g., pedestrians, obstacles) negatively impacted safety when it was selected by the simple edge adaptation logic in non-intersection scenarios. This issue stems from the NPC's design as a simple rule-based algorithm, highlighting the challenge of relying solely on simple edge models for complex, unforeseen events and underscoring the need for robust cloud-based pre-analysis to identify such weaknesses or provide mechanisms for escalating to more capable models or cloud intervention.

Similarly, Transfuser's occasional safety failures in certain intersection scenarios, despite high overall scores, demonstrate that the adaptive system's performance is ultimately bounded by the capabilities and potential flaws of the underlying models deployed at the edge. While edge switching can optimize system selection based on known characteristics derived from pre-analysis, it inherits the drawbacks of its constituent systems if the cloud pre-runtime analysis fails to adequately characterize these limitations or if the edge cannot reliably detect the critical conditions requiring a specific model.

Consequently, these observed limitations underscore the critical role of the cloud-based pre-runtime phase, involving extensive simulation-based testing and behavioral analysis, in building a comprehensive and accurate knowledge base. This knowledge base is essential for informing effective edge runtime adaptation logic, enabling accurate mapping of scenarios to the most suitable edge-executable systems, and understanding the operational boundaries and limitations of each component. Therefore, while this study validates the potential of the edge-based adaptive approach, it simultaneously highlights the critical need for thorough cloud-based pre-analysis as an indispensable element for optimizing both safety and efficiency in the design of adaptive edge-cloud ADS systems.

In summary, the CARLA-based preliminary evaluation demonstrated that simple heuristics, such as rules for intersections, are insufficient to substitute for the framework's 'cloud pre-runtime analysis' phase. This finding strongly highlights the need for a data-driven approach to identify the actual weaknesses and operational boundaries of each ADS component.

Therefore, to overcome these limitations, this extended journal study conducts the main validation, which leverages large-scale MetaDrive-based scenarios and a decision tree.

4.2 Evaluation on Data-driven Scenario–System Mapping

The preliminary validation, summarized in Section 4.1, demonstrated the potential of runtime adaptation while also clarifying that the quality of the 'knowledge base' used in the 'cloud pre-runtime analysis' phase profoundly impacts overall system performance. Simple heuristic rules proved insufficient for capturing the complex operational boundaries of ADS components across diverse scenarios.

To overcome these limitations and validate the overall effectiveness of the proposed edge-cloud framework, this section conducts the main validation using a data-driven approach. As discussed in the introduction, achieving both safety and efficiency consistently across diverse driving scenarios is a primary challenge for ADSs, especially considering edge resource constraints and the limitations of static or existing adaptive approaches. Our proposed framework aims to improve this balance through adaptive ML switching in an edge-cloud context. Therefore, this preliminary evaluation focuses on quantifying the potential benefits regarding these two critical aspects, guided by the following research questions (RQs):

RQ1 (Safety): What advantages does the adaptive ADS offer in terms of safety compared to conventional ADS, either holistically or in specific scenarios?

RQ2 (Efficiency): What advantages does the adaptive ADS provide in terms of efficiency compared to conventional ADS?

Experiments ran on Ubuntu 22.04 (Intel Xeon 4215R, 3x RTX A5000 GPUs 24GB, 128GB RAM). The replication kit containing the source code used in this study is publicly available at [4].

4.2.1 Experiment Design

For this experiment, we implemented a simple adaptive ADS prototype focusing on runtime adaptation at the simulated edge. The prototype uses the MetaDrive simulator [16], our simulated edge environment. MetaDrive is

lightweight, offers high-speed execution, and excels at procedural scenario generation. In contrast to the CARLA Leaderboard, which provides fixed benchmark scenarios, MetaDrive can generate a nearly infinite number of randomized scenarios based on seed values. This makes it highly suitable for testing the framework's robustness against diverse edge cases and unseen scenarios-a key limitation identified in the original paper.

The MetaDrive scenario generator takes a single seed number to instantiate one scenario. Each scenario is represented by a feature vector of over 400 dimensions, defining a map composed of five tiles, the state of the ego vehicle, and the states of up to 36 other vehicles. For this experiment, we used a total of 100,000 unique scenarios, which were split into the following two disjoint sets:

- **Training set**: Comprises 10,000 scenarios based on seeds 0 to 9999. This set is used in the 'cloud pre-runtime analysis' phase to train the decision tree 'knowledge base'.
- **Test set**: Comprises 90,000 scenarios based on seeds 10,000 to 99,999. This unseen data, not used during training, is utilized to evaluate the 'runtime adaptation' performance and compare the performance of the three system configurations.

The evaluation focused on safety and efficiency. We employed safety metrics in MetaDrive analogous to those used in CARLA. Specifically, we calculated a *composed score* by replicating the route score and penalty score from the CARLA Leaderboard, based on the driving logs from MetaDrive. Similarly, efficiency was measured as the number of simulation steps the ADS policy could process per second (steps/s), where a higher value signifies lower computational resource consumption.

The prototype combines expert policy [19] and the IDM policy sourced from the MetaDrive. Expert policy is a simple neural network-based ADS provided by the developers of MetaDrive. It utilizes lidar and lane detector. The IDM policy is a simple, computationally efficient rule-based baseline. It tries to keep gap between other cars.

We compared the performance of the following three ADS on the 90,000 test scenarios.

- **Expert-only**: A static ADS that exclusively utilizes the expert policy across all scenarios.
- **IDM-Only**: A static ADS configuration that exclusively utilizes the IDM policy across all scenarios.

- **Adaptive ADS**: The adaptive ADS leveraging the decision tree. At runtime, this system analyzes the current scenario features to dynamically select and switch to the policy predicted to yield superior performance.

4.2.2 Implementing the pre-runtime phase

In the preliminary evaluation, the 'cloud pre-runtime analysis' phase was approximated by a simple heuristic rule. This approach demonstrated clear limitations, as it failed to guarantee safety in specific scenarios like a pedestrian outside an intersection.

In this main validation, we adhere to the framework and present a concrete, data-driven implementation of the 'pre-runtime phase'. This phase, which constitutes an offline analysis step requiring massive computational resources, clearly exemplifies the role of the 'cloud' within the edge-cloud architecture. The objective is to leverage the 10,000 training scenarios to generate an efficient and reliable 'knowledge base' that the edge device can utilize at runtime.

This process was conducted by instantiating the three sub-phases as defined in section 3.

Simulation-based testing: The first step is to comprehensively test the behavior of the candidate systems. To this end, we executed the two candidate ADS policies independently on each of the 10,000 training scenarios. Through this large-scale simulation, totaling 20,000 runs, we collected raw performance data. Each scenario result is composed of the following two key elements. One is scenario feature vector. It includes the topology of the 5-tile map, such as road curvature, the initial state of ego vehicle, and initial states of up to 36 other vehicles, such as location. Another is performance results. The two key performance metrics achieved by the policy in that scenario, namely, driving score (safety), and steps/s (efficiency). This phase is a computationally intensive process, feasible due to the cloud environment, which generates data enabling a direct comparison of the two policies' performance under identical scenarios. The simulation for 10,000 scenarios for 2 ADSs took approximately 23 hours.

Behavioral analysis: In the subsequent phase, we transformed the 20,000 raw data points into a supervised learning dataset of 10,000 labeled instances, guided by the 'pre-defined objectives.' The primary objective of ADS is the maximization of safety. Efficiency is a secondary objective, considered only when safety levels are equivalent. Based on this principle, we assigned an 'optimal policy' label to each of the 10,000 scenarios based on the

performance results. If the safety metric of expert policy is greater or equal than IDM policy, it selects expert policy. Otherwise, it selects IDM policy. This procedure effectively reframes the raw performance data as a training dataset of classification problem determining which policy to select for given scenario feature vector.

Scenario-system mapping: Finally, to implement the 'scenario-system mapping' phase, we trained a decision tree classifier using the training dataset generated in the previous step. We chose decision tree because of interpretability. Although the 'intersection' rule in Section 4.1 was simplistic, its decision logic was transparent. When dealing with over 400 complex features, a black-box model like a neural network makes validating the framework harder. In contrast, decision tree generates human-readable rules. It can show the operational boundaries of ADS in scenario space. The fully trained decision tree model constitutes the concrete 'knowledge base' to be utilized during the 'runtime phase'. It replaces the static and simplistic heuristic rules with a dynamic and sophisticated mapping learned from the 10,000 scenario dataset.

To ensure consistency and robustness in model training, we mitigated bias arising from initialization randomness by training 10 independent decision tree models, varying the random state from 42 to 51. Additionally, to prevent overfitting and ensure fast inference, the maximum tree depth was constrained to max depth at 10.

4.2.3 Experiment results

We evaluated the performance of three ADS systems using the 90,000 unseen test scenarios. The results for the adaptive ADS are derived from applying each of the 10 independent decision tree models to the 90,000 test scenarios. The final performance of all systems is summarized in Table 1. This table directly compares the two fixed policies against the individual and average performance of the 10 decision tree models.

(RQ1) Safety. In terms of safety, Table 1 clearly demonstrates that the proposed adaptive ADS achieved superior performance across the 90,000 test scenarios. The adaptive ADS recorded an average driving score of 0.7884 over the 10 runs. This represents safety improvement compared to the static expert policy ADS. The IDM policy ADS exhibited the lowest performance at 0.7080. This result not only shows that the limitations of the a simple rule-based approach evaluation were successfully overcome but also signifies a critical achievement of our framework.

Table 1

Policy	Seed	Safety	Efficiency
Expert	–	0.7685	479.23
IDM	–	0.7080	3,779.93
Adaptive	42	0.7893	978.49
	43	0.7885	941.88
	44	0.7878	1,007.67
	45	0.7882	960.48
	46	0.7882	960.48
	47	0.7873	970.90
	48	0.7885	981.26
	49	0.7889	951.35
	50	0.7886	978.05
	51	0.7885	972.38
	Average	0.7884	970.29

To analyze how the adaptive ADS achieved better performance than the static ADS systems, the average classification performance of the 10 decision tree models on the 90,000 test scenarios is presented in Table 2. Here, the 'expert' class refers to scenarios where expert policy was deemed optimal according to labeling rules in behavioral analysis, while 'IDM' class refers to scenarios where IDM policy was deemed optimal.

The label distribution of the 90,000 test scenarios reveals a significant imbalance: the expert policy was optimal in 68,480 scenarios (approx. 76.1%), while the IDM policy was optimal in 21,520 scenarios (approx.

Table 2

	Expert			IDM		
Seed	Precision	Recall	F1 Score	Precision	Recall	F1 Score
42	0.809	0.914	0.853	0.534	0.312	0.164
43	0.815	0.899	0.851	0.521	0.350	0.180
44	0.810	0.911	0.853	0.530	0.321	0.168
45	0.810	0.911	0.853	0.530	0.321	0.168
46	0.810	0.906	0.852	0.522	0.326	0.170
47	0.812	0.905	0.852	0.524	0.332	0.173
48	0.811	0.913	0.853	0.538	0.323	0.169
49	0.812	0.906	0.852	0.524	0.331	0.172
50	0.811	0.907	0.852	0.527	0.329	0.171
51	0.812	0.903	0.852	0.520	0.334	0.174
Average	0.811	0.907	0.852	0.527	0.328	0.171

23.9%). This distribution explains why the expert policy baseline (0.7685) substantially outperformed the IDM policy baseline (0.7080).

The decision tree model demonstrates a high average recall of 0.898 for the majority 'expert' class. This indicates that the adaptive ADS correctly identifies and executes the 'expert' policy in approximately 90% of the scenarios where it is required. This high recall ensures that the adaptive ADS's performance is fundamentally anchored near the expert policy baseline.

The expert policy ADS, by definition, uses the 'expert' policy for the whole time. Consequently, in the 21,520 scenarios (23.9%) where IDM policy is safer, it inevitably selects a suboptimal policy, incurring a performance penalty. In contrast, Table 2 shows that the adaptive ADS achieves a recall of 0.328 for the 'IDM' class. This signifies that the adaptive system successfully identifies and switches to the 'IDM' policy in approximately 32.8% (average 7055) of the scenarios where 'IDM' is the safer option.

(RQ2) Efficiency. In terms of efficiency, Table 1 demonstrates that the proposed framework successfully manages the trade-off between safety and efficiency. The adaptive-DT system achieved an average processing speed of 970.29 steps/s, attaining approximately 102.5% higher efficiency than the least efficient system, expert policy ADS (479.23 steps/s). Although this is substantially lower than the IDM policy ADS (3779.93 steps/s), it signifies that the computational load on the edge device was halved compared to expert policy, representing a significant resource-saving.

The reason for this efficiency improvement is also explained by the classification report in Table 2. The adaptive ADS executes the highly efficient 'IDM' policy in scenarios where 'IDM' was correctly predicted as optimal (average 7055 scenarios), and scenarios where 'expert' was optimal but was incorrectly predicted as 'IDM' (average 6339 scenarios). By avoiding the execution of the costly expert policy and instead using the IDM policy in these combined scenarios (average 13,393 scenarios), the system nearly doubled its overall efficiency from 479 to 970 steps/s.

Furthermore, the efficiency results from the 10 runs of adaptive ADS exhibit very low variability relative to the mean. This confirms that the system provides predictable and stable resource utilization.

5 Discussion

5.1 Investigation Analysis

This study proposed an edge-cloud adaptive framework in Section 3 to dynamically optimize the safety and efficiency of ML-based autonomous

systems. In Section 4, a two-phase evaluation was conducted to validate this framework. The CARLA-based preliminary investigation in Section 4.1 demonstrated the potential and clear limitations of simple heuristic rules. The MetaDrive-based investigation in Section 4.2 implemented and tested the framework of pre-runtime analysis using 100,000 large-scale scenarios and a data-driven decision tree model.

As summarized in Table 1, the key findings of this study are clear. The adaptive ADS (avg. 0.7884) achieved higher safety than the static ADS, expert policy (0.7685). Furthermore, in terms of efficiency, the adaptive ADS (avg. 970.29 steps/s) was 102.5% more efficient than expert policy ADS (479.23 steps/s). This indicates that the proposed framework successfully and simultaneously optimized the two conflicting objectives of safety and efficiency-an achievement not attained in the preliminary evaluation.

The adaptive ADS successfully identified approximately 32.8% of the 21,520 scenarios where 'IDM' was safer, switching to the 'IDM' policy instead of 'expert'. However, this process incurred a cost by misclassifying some scenarios where 'expert' was safer as 'IDM'. Nonetheless, the benefits of switching to the 'IDM' policy outweighed these losses in terms of both safety and efficiency. That is, the framework successfully achieved a trade-off, doubling efficiency while simultaneously improving overall safety.

The key significance of this experiment is demonstrating that the limitations of simple heuristic scenario–system mapping rule were overcome, and the two conflicting goals optimized using a relatively simple decision tree model as the 'knowledge base', without resorting to deep analysis of the complex scenario space or costly optimization techniques like reinforcement learning. This shows that the 'pre-runtime analysis' is feasible, and if sufficient simulation data can be stored in the cloud, an adaptive ML-enabled system can be implemented by analysis.

Nevertheless, this experiment should still be considered a preliminary validation. As shown in Table 2, the overall classification accuracy of the 'knowledge base' is only around 0.77, and notably, the recall for the 'IDM' policy is very low. This implies that the current 'knowledge base' has not perfectly learned the scenario–system mapping and is still missing approximately 67% of the scenarios where IDM is optimal.

This implies that the 400+ feature vector may not fully capture the scenario complexity, or that the simple decision tree model (max depth $= 10$) lacks the sophistication to capture the complex operational boundaries between the two policies. Consequently, while these 'preliminary' validation results establish the viability of the proposed framework, they also

strongly underscore the necessity of future research toward building a more sophisticated 'knowledge base'.

5.2 Future Directions

To overcome the limitations identified and fully realize the framework's potential, future work should first address the refining the cloud pre-runtime analysis. While our implementation in this research using a decision tree model demonstrated the feasibility of this mapping, it exhibited limitations in capturing the complex operational boundaries between the two policies given features, as analyzed. Indeed, when considering a wider array of candidate ADS and longer feature vectors, more systematic analysis methods are required.

One avenue is the need for robust scenario generation techniques. Simulation scenarios are critical for evaluating the performance and identifying the limitations of candidate ADS models intended for edge deployment. Cloud-based scenario generation methods should systematically explore the operational boundaries of these models by designing diverse scenarios including edge cases, such as unexpected obstacles or highly complex intersections. These methods need to balance criticality (exploring key boundary conditions) and variety (covering diverse environments), while enabling automation to reduce the time and cost associated with building the pre-runtime knowledge base.

Another critical direction is the analysis of the scenario space itself. The scenario space must be decomposed based on the characteristics of the candidate ML-enabled systems. By appropriately partitioning the scenario space-based on analyzing the intrinsic properties of the ML-enabled systems and the results of simulation-based testing-and allocating these partitioned spaces to specific candidate systems, a more effective adaptive framework can be constructed. To achieve this, research focused on decomposing the scenario space based on the learning dynamics and mechanisms of the ML-enabled systems is essential.

Furthermore, effective collaboration mechanisms between the edge and the cloud require in-depth research. Defining clear criteria for when scenario analysis should be handled solely by the edge versus requiring cloud interaction is a key challenge in designing the collaboration logic. This includes designing efficient communication protocols, maintaining data consistency, managing potential conflicts between edge and cloud decisions, and developing strategies for dynamic task allocation and resource management

considering real-time requirements, communication constraints, and computational loads at both edge and cloud.

Future research will prioritize integrating comprehensive cloud-based behavioral analysis with sophisticated self-adaptive techniques executed at the edge, ensuring effective edge-cloud collaboration. Such efforts aim to maximize the safety and efficiency of autonomous driving systems by leveraging the strengths of both edge and cloud resources, while ensuring reliable performance in complex environments. Ultimately, advancing edge-cloud adaptive frameworks is expected to improve the feasibility of autonomous driving technologies for commercialization and play a critical role in developing safe and efficient autonomous systems.

6 Conclusion

We proposed an adaptive ML-enabled edge-cloud framework to enhance autonomous system safety and efficiency. The framework combines cloud pre-runtime analysis with collaborative edge-cloud runtime adaptation (including ML switching) to handle diverse requirements.

The original study [7], through the CARLA-based preliminary evaluation, confirmed both the potential and the limitations of this framework. To overcome these limitations, we conducted a preliminary validation by implementing the cloud-based pre-runtime analysis component of the proposed framework, leveraging 100,000 MetaDrive scenarios and a data-driven decision tree model. As validated on 90,000 test scenarios, the proposed adaptive ADS achieved higher safety while concurrently attaining an adequate level of efficiency, thereby quantitatively demonstrating the framework's viability.

A prototype confirmed the edge adaptation's applicability and highlighted future work: developing robust scenario generation techniques, analysis and decomposition of the scenario space, and refining edge-cloud collaboration. The next step will be to refine and complete the implementation of the framework preliminarily validated in this study.

Acknowledgements

This work was partly supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2025-02218761, 50%), IITP grant funded by the Korea government (MSIT) (RS-2024-00406245, 25%), Samsung Electronics (25%).

References

[1] Baidu apollo team (2017), apollo: Open source autonomous driving. ht tps://github.com/ApolloAuto/apollo. Accessed: 2024-12-09.

[2] Carla autonomous driving leaderboard. https://leaderboard.carla.org/. Accessed: 2024-12-09.

[3] Replication kit. https://github.com/EunhoCho/AdaptiveADS/. Accessed: 2025-04-17.

[4] Replication kit. https://github.com/EunhoCho/MetadriveADS/. Accessed: 2025-10-31.

[5] Maral Amir, Frank Vahid, and Tony Givargis. Switching predictive control using reconfigurable state-based model. *ACM transactions on Design Automation of Electronic systems (tODAEs)*, 24(1):1–21, 2018.

[6] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):12878–12895, 2023.

[7] Eunho Cho and In-Young Ko. Adaptive ml-enabled edge-cloud system framework for safe and efficient autonomous systems. In *5th International Workshop on Big data driven Edge Cloud Services (BECS 2025) Co-located with the 25th International Conference on Web Engineering (ICWE 2025)*, 2025.

[8] Eunho Cho, Gwangoo Yeo, Eunkyoung Jee, and Doo-Hwan Bae. Anomaly-aware adaptation approach for self-adaptive cyber-physical system of systems using reinforcement learning. In *2022 17th Annual System of Systems Engineering Conference (SOSE)*, pages 7–12. IEEE, 2022.

[9] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[10] Fitash Ul Haq, Donghwan Shin, and Lionel Briand. Efficient online testing for dnn-enabled systems using surrogate-assisted and many-objective optimization. In *Proceedings of the 44th international conference on software engineering*, pages 811–822. IEEE, 2022.

[11] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*, 2017.

[12] Xiaofeng Ji, Faming Gong, Nuanlai Wang, Junjie Xu, and Xing Yan. Cloud-edge collaborative service architecture with large-tiny models based on deep reinforcement learning. *IEEE Transactions on Cloud Computing*, 2025.

[13] Shinpei Kato, Shota Tokunaga, Yuya Maruyama, Seiya Maeda, Manato Hirabayashi, Yuki Kitsukawa, Abraham Monrroy, Tomohito Ando, Yusuke Fujii, and Takuya Azumi. Autoware on board: Enabling autonomous vehicles with embedded systems. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, pages 287–296. IEEE, 2018.

[14] Jeffrey O Kephart and David M Chess. The vision of autonomic computing. *Computer*, 36(1):41–50, 2003.

[15] Shubham Kulkarni, Arya Marda, and Karthik Vaidhyanathan. Towards self-adaptive machine learning-enabled systems through qos-aware model switching. In *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 1721–1725. IEEE, 2023.

[16] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[17] Khan Muhammad, Amin Ullah, Jaime Lloret, Javier Del Ser, and Victor Hugo C de Albuquerque. Deep learning for safe autonomous driving: Current challenges and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 22(7):4316–4336, 2020.

[18] Hirofumi Noguchi, Takuma Isoda, and Seisuke Arai. Shared trained models selection and management for transfer reinforcement learning in open iot. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 2170–2176. IEEE, 2021.

[19] Zhenghao Peng, Quanyi Li, Chunxiao Liu, and Bolei Zhou. Safe driving via expert guided policy optimization. In *5th Annual Conference on Robot Learning*, 2021.

[20] Alexandru Constantin Serban. Designing safety critical software systems to manage inherent uncertainty. In *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*, pages 246–249. IEEE, 2019.

[21] Sina Shafaei, Stefan Kugele, Mohd Hafeez Osman, and Alois Knoll. Uncertainty in machine learning: A safety perspective on autonomous driving. In *Computer Safety, Reliability, and Security: SAFECOMP 2018 Workshops, ASSURE, DECSoS, SASSUR, STRIVE, and WAISE,*

Västerås, Sweden, September 18, 2018, Proceedings 37, pages 458–464. Springer, 2018.

[22] Hao Shao, Letian Wang, Ruobing Chen, Hongsheng Li, and Yu Liu. Safety-enhanced autonomous driving using interpretable sensor fusion transformer. In *Proceedings of The 6th Conference on Robot Learning*, volume 205, pages 726–737. PMLR, 2023.

[23] Sepehr Sharifi, Donghwan Shin, Lionel C. Briand, and Nathan Aschbacher. Identifying the hazard boundary of ml-enabled autonomous systems using cooperative coevolutionary search. *IEEE Transactions on Software Engineering*, 49(12):5120–5138, 2023.

[24] Chellammal Surianarayanan, John Jeyasekaran Lawrence, Pethuru Raj Chelliah, Edmond Prakash, and Chaminda Hewage. A survey on optimization techniques for edge artificial intelligence (ai). *Sensors*, 23(3):1279, 2023.

[25] Jie M Zhang, Mark Harman, Lei Ma, and Yang Liu. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering*, 48(1):1–36, 2020.

Biographies

Eunho Cho is a Ph.D. candidate of Web Engineering and Service Computing Lab, School of Computing, Korea Advanced Institute of Science and Technology (KAIST). He received his master's degree from the School of Computing at KAIST. His research interests include AI4SE, simulation-based testing, autonomous driving system testing, self-adaptive system and cyber-physical systems.

In-Young Ko is a professor at the School of Computing at the Korea Advanced Institute of Science and Technology (KAIST) in Daejeon, Korea. He received his Ph.D. in computer science from the University of Southern California (USC) in 2003. His research interests include services computing, web engineering, and software engineering. His recent research focuses on service-oriented software development in large-scale and distributed system environments such as the web, Internet of Things (IoT), and edge cloud environments. He is a member of the IEEE.