SIEGE: A Semantics-Guided Safety Enhancement Framework for AI-enabled Cyber-Physical Systems

Jiayang Song, Xuan Xie, and Lei Ma

Abstract—Cyber-Physical Systems (CPSs) have been widely adopted in various industry domains to support many important tasks that impact our daily lives, such as automotive vehicles, robotics manufacturing, and energy systems. As Artificial Intelligence (AI) has demonstrated its promising abilities in diverse tasks like decision-making, prediction, and optimization, a growing number of CPSs adopt AI components in the loop to further extend their efficiency and performance. However, these modern AI-enabled CPSs have to tackle pivotal problems that the AI-enabled control systems might need to compensate the balance across *multiple operation requirements* and avoid possible defections in advance to safeguard human lives and properties. Modular redundancy and ensemble method are two widely adopted solutions in the traditional CPSs and AI communities to enhance the functionality and flexibility of a system. Nevertheless, there is a lack of deep understanding of the effectiveness of such ensemble design on AI-CPSs across diverse industrial applications. Considering the complexity of AI-CPSs, existing ensemble methods fall short of handling such huge state space and sophisticated system dynamics. Furthermore, an ideal control solution should consider the multiple system specifications in real-time and avoid erroneous behaviors beforehand. Such that, a new specification-oriented ensemble control system is of urgent need for AI-CPSs.

In this paper, we propose SIEGE, a semantics-guided ensemble control framework to initiate an early exploratory study of ensemble methods on AI-CPSs and aim to construct an efficient, robust, and reliable control solution for multi-tasks AI-CPSs. We first utilize a semantic-based abstraction to decompose the large state space, capture the ongoing system status and predict future conditions in terms of the satisfaction of specifications. We propose a series of new semantics-aware ensemble strategies and an end-to-end Deep Reinforcement Learning (DRL) hierarchical ensemble method to improve the flexibility and reliability of the control systems. Our large-scale, comprehensive evaluations over five subject CPSs show that 1) the semantics abstraction can efficiently narrow the large state space and predict the semantics of incoming states, 2) our semantics-guided methods outperform state-of-the-art individual controllers and traditional ensemble methods, and 3) the DRL hierarchical ensemble approach shows promising capabilities to deliver a more robust, efficient, and safety-assured control system. To enable further research along this direction to build better AI-enabled CPS, we made all of the code and experimental results data publicly available at https://sites.google.com/view/ai-cps-siege/home.

Index Terms—Cyber-Physical Systems, Reinforcement Learning, State Abstraction, AI Controllers, Ensemble Methods.

1 Introduction

Cyber-Physical Systems (CPSs) are often complex systems where physical plants collaborate with computer units to perform complex tasks [1]. Due to its cyber-physical integration characteristics, CPS shows a promising capability to enhance and assure the efficiency and reliability of large-scale industrial systems. Considering the advanced interdependencies and collaborative ability between the digital and physical units, CPSs are anticipated to be able to solve complicated and challenging control tasks in various domains, such as autonomous driving, power grids, and medical devices. In terms of the control systems in CPSs, the commonly used traditional controllers are Model Predictive Control (MPC), Proportional-Integral-Derivative (PID) Control, Linear Quadratic Regulator (LQR), etc. We would like to use one of our collected CPSs, Adaptive Cruise Control System (ACC), as a running example to illustrate the characteristics of controllers in CPSs. A model predictive controller is used in ACC, which takes the relative distance between the ego car and the lead car, the user-set cruising velocity, the ego car velocity and the relative velocity to the lead car as inputs and outputs an acceleration control signal to the ego car. MPC generates a control command at each time-step by predicting the movements of the two vehicles in a finite time-horizon. MPC aims to track the user-set cruising velocity while maintaining a safe distance from the lead car.

Artificial Intelligence (AI), a rising trend of industrial adoption in the past decade, shows promising abilities in behavior imitation, decision making and optimization. Among all kinds of AI methods, Deep Reinforcement Learning (DRL) is considered the specialized approach to offer optimized control strategies for non-linear, stochastic systems with high uncertainty [2], [3], [4], [5], [6]. Both industry and academia are motivated to utilize AI as complements or substitutions to empower traditional applications [7], [8]. Likewise, a growing number of CPSs have also adapted AI controllers inside the loops to improve operation safety, efficiency, and security [7], [9]. We refer to this type of AI-embedded CPSs as *AI-enabled CPS (AI-CPS)* [10], [11], [12].

For industrial-scale CPSs, *multiple requirements* are com-

- *Jiayang Song, Xuan Xie and Lei Ma are with University of Alberta, Canada. E-mail: {jiayan13, xxie9}@ualberta.ca, ma.lei@acm.org.*
- *Lei Ma is also with the University of Tokyo, Japan.*

monly needed to be considered during the design and development of the system. However, considering the complex dynamics nature of CPS in general, as shown by Song et al. [13], a single AI controller can fall short and be limited in handling all feasible complex situations due to the trade-off between optimality and generality. As CPSs are widely deployed as safety-critical applications, researchers and industry practitioners often try their best to exert all possible solutions to assure the safety and robustness of the systems against uncertainties and potential defections.

According to the international standards, e.g.,*ISO26262* [14] and *ISO/PAS 21448 (SOTIF)* [15], modular redundancy [16] has been confirmed as a useful technique to tackle such safety problem. Specifically, modular redundancy refers to a type of system design with duplicated components that operate parallel to increase the overall reliability and overcome possible failures [17], [18], [19]. In terms of control systems, duplicated controllers can operate alternately or parallelly under a single task to prevent potential defections and hold the safety property at all costs.

In the AI community, a similar duplicated-model design, *ensemble learning* [20], [21], [22], [23], has also been commonly applied in many tasks such as classification, regression, and transcription [24], [25], [26]. Ensemble learning combines multiple learning algorithms via certain strategies to achieve better performance than could be obtained by any of the constituent sub-algorithms. Each weak learner inside the ensemble pool does not need to be superior compared to other single-learner algorithms, as the ensemble system can generate a collective output to compensate for the insufficiency of weak learners. Specifically, ensemble learning is commonly used for classification, pattern recognition and detection for ML tasks which have limited discrete output space [20]. However, in terms of CPSs, the control signals are generally represented by large continuous action space, which brings significant challenges to existing ensemble methods. Moreover, CPSs often come with critical safety requirements such that a desired ensemble method should have a safety-aware mechanism to prevent any potential safety violations. The majority of existing tree-based ensemble methods rely on heuristics, and inferring an optimal decision tree is known to be NP-complete for many optimalities [27]; these methods do not work well on large complex systems such as CPSs.

As a cross-point between AI and CPS, we believe it could be promising to leverage the domain knowledge from both communities to construct more reliable, flexible and safety-assured AI-CPSs. Inspired by both the modular redundancy design and ensemble learning, instead of relying on a single or individual AI controller, combining a ground of AI controllers with different behavioural strategies is a promising way to obtain better functional capabilities. However, currently, there is still a lack of understanding of adapting the ensemble methods in the context of CPS domains. Ensemble methods are often designed for typical ML tasks that are not easily transferable to controller design. In addition, to develop a CPS-oriented ensemble strategy, some abstraction techniques are needed to: 1) decompose the large state space of the CPS, 2) describe the status of the system and the behavior of the controller, and 3) provide guidance to the ensemble workflow.

Therefore, to bridge this gap, in this early exploratory work, we propose `SIEGE`: a semantics-guided ensemble control framework to systematically explore and understand the capabilities of ensemble methods in the context of AI-CPS.

We first train sub-controllers with diverse behavior as the cornerstone of the ensemble system. To reduce the prohibitively large state space of the CPS, we then leverage the *semantics-based abstraction*, which provides a human-interpretable way to describe the system status. Semantics is a representation of the degree of satisfaction of the system w.r.t. specifications. Furthermore, we incorporate as many as six ensemble strategies, which aim to provide control logic of the system: majority voting, average, Top-1 and Top-k semantics prediction, coordinator, and coordinator (with prediction). The strategies are based on the constructed semantics-based abstract model, which provides semantics information.

To demonstrate the usefulness of our framework, we collect five industry-level CPS environments from different domains. For each of them, we train 9 DRL-based individual AI controllers with different algorithms, structures of reward functions and agent configurations. Finally, a large-scale evaluation is performed to reveal the effectiveness and characteristics of the ensemble framework on a total of 5 representative AI-CPS tasks, 26 system specifications, 45 DRL agents, 6 ensemble methods, and over 62,500 experimental runs.

The evaluation results and our in-depth analysis demonstrate that: 1) the semantics-based abstraction can effectively describe the system status in a succinct and precise manner 2) the traditional ensemble methods can not much outperform the individual DRL controllers; 3) the new proposed semantics-guided ensemble strategies can bring better performance than the classical ensemble methods, showing that the AI-ensemble system is a promising diction to extend the flexibility and reliability of AI-CPSs.

The main contributions (also depicted in Figure 1) of our paper are summarized as follows:

- A novel semantics-based abstraction, specially designed for CPSs by taking the specifications into account, which gives a human-interpretable way to describe system behavior, and provides predictive guidance for ensemble strategies;
- A general-purpose ensemble framework, `SIEGE` enclosed 4 new ensemble methods with the utilization of semantics predictions;
- A large-scale evaluation is performed to reveal characteristics of the ensemble framework on a total of 5 representative domains, 26 system specifications, 45 DRL agents, 6 ensemble methods, and over 62,500 experiment runs. The results confirm that `SIEGE` outperforms individual state-of-the-art DRL controllers and classical ensemble methods. Our proposed semantics-guided ensemble framework is indeed promising to construct more optimized, robust and reliable control systems for AI-CPSs.
- We initiate an exploratory study to investigate the performance of ensemble methods on AI-CPSs. Our work aims to bring more inspiration to researchers and motivate

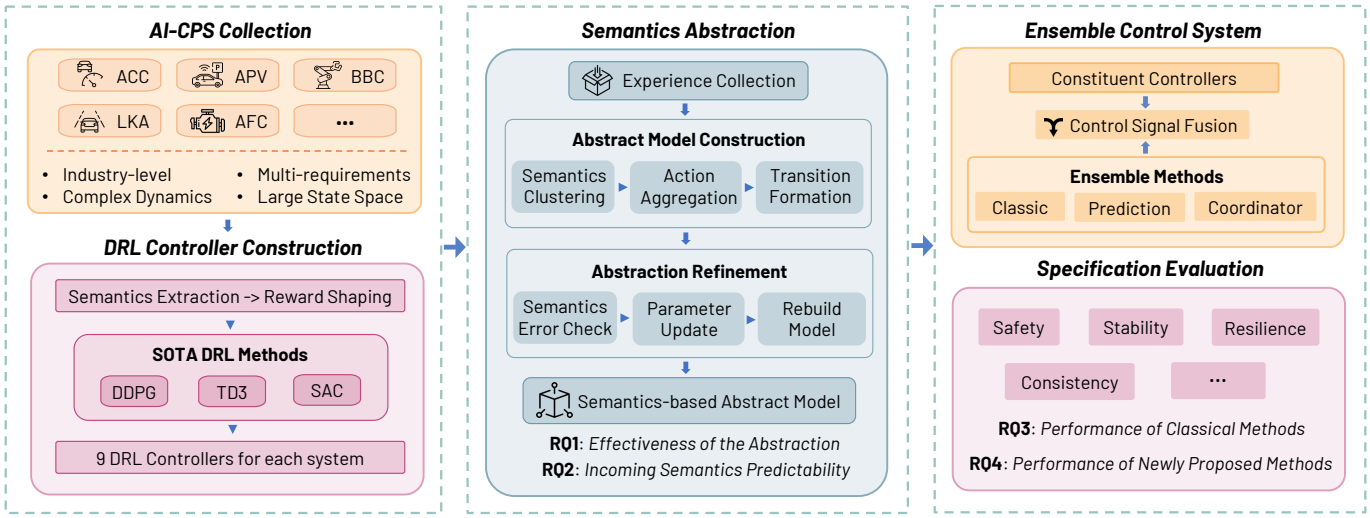


Fig. 1: Workflow summary of SIEGE: A semantics-guided ensemble framework

further improvements in terms of safe and reliable AI-CPSs.

The Contributions to the Software Engineering Field. CPSs are highly integrated systems which include several disciplines such as mechanical, control, electrical and particularly software engineering (SE) [28], [29]. The role of SE plays in CPSs is critical, which coordinates different disciplines throughout the development and deployment so that numerous system components can operate concertedly to provide designed functionalities. In particular, how to improve reliability and provide quality assurance for CPSs is one of the important topics in software engineering [30], [31], [32], [33]. Our work, following the line of the research, aims to enhance the overall safety and reliability of CPSs from various industry domains by leveraging semantics abstraction, modular redundancy design and ensemble learning.

The remainder of the paper is structured as follows. Section 2 introduces the corresponding background. Section 3 describes the proposed abstraction and ensemble methods, and Section 4 presents the experiment setup and evaluation results. Section 5 discusses the potential impact of our outcome. Section 6 analyzes the threats that may affect the validity of our work. Section 7 reviews related work, and Section 8 concludes the paper.

2 Background

In this section, we first give the essential background knowledge on *AI-enabled Cyber-Physical Systems* and *Deep Reinforcement Learning* controllers. We then introduce the *state abstraction* technique and the *ensemble* control method. In addition, we provide a brief description on *Signal Temporal Logic (STL)* [34], a formal language for CPS to describe the expected system behaviour.

2.1 AI-Enabled Cyber-Physical Systems(AI-CPS)

The AI-enabled Cyber-Physical System is illustrated in Figure 2, which mainly contains four components: the AI-based controller, the actuator, the plant, and the sensor. The AI-based controller (*e.g.* Deep Neural Network (DNN), Deep

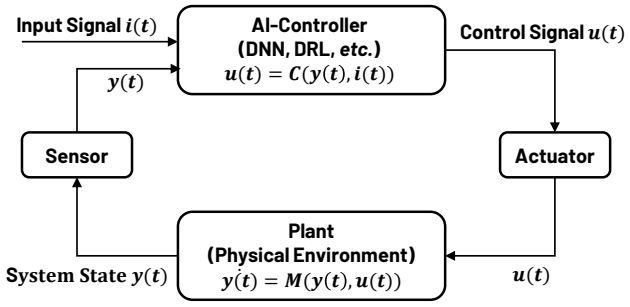


Fig. 2: A cyber-physical system with an AI controller

Reinforcement Learning(DRL)), C, plays a predominant role in the AI-CPS. It takes the system state $y(t)$, from the sensor and the input signal $i(t)$ and outputs a control signal $u(t)$ to control the actuator. In this work, we focus on DRL controllers. The actuator regulates the behavior of the plant toward desired performance by following the control commands $u(t)$ generated by the AI controller. The plant is a physical environment where the next system state $\dot{y}(t)$ is determined by the non-linear continuous dynamics M, the current system state $s(t)$ and the actuator output $u(t)$. The sensor works as a sampler, which sample the continuous system state y and outputs a discrete state $y(t)$ at time step t.

2.2 Deep Reinforcement Learning (DRL)

The behavior of a DRL agent can be described as a *Markov Decision Process (MDP)* $M = (S, A, R, P, \gamma)$, where S is the state space which contains the environment information , A is the action space, $P : S \times A \to [0, 1]$ is the transition function that maps the state-action pair (s, a) to a probability distribution over a successor state s', $R(s, a)$ is a reward function that initiates a reward value to the current state-action pair, and $\gamma \in (0, 1)$ is a discount factor [35]. A discount factor γ determines the importance of between the immediate reward and the future rewards. A large γ will let the agent learn from a long-term reward and vice versa. An MDP defines the environment that an RL agent

will learn and react, and an optimal policy leads the agent to achieve maximum cumulative rewards. The policy of an MDP $\pi : S \rightarrow A$ maps a state to an action that leads to an optimized reward.

Along with the fast advancement and evolution in the RL community, plenty of online, model-free learning algorithms have been proposed to compensate for various demands. There are many well-known, state-of-the-art algorithms such as, *Deep deterministic policy gradient (DDPG)* [36], *Twin-delayed deep deterministic policy gradient (TD3)* [37], *Actor-critic (A2C)* [38], *Proximal policy optimization (PPO)* [39], *Soft actor-critic (SAC)* [40], *etc*. We refer interested readers to [41] for more details.

DRL agents are promising substitutes for conventional control systems, especially in large-scale plants with complicated dynamics [42], [43]. As a combination of reinforcement Learning and Deep Learning, DRL trains an agent in an environment to make sequential decisions towards the final objective [44], [45]. DRL explores the environment and searches for the optimal policy by the trial-and-error paradigm; namely, at each timestep, an action is selected from a set of possible actions to maximize the cumulative reward. In AI-CPS, the DRL agent behaves as a controller that receives environment information from sensors and generates commands to the actuator to control the plant.

2.3 State Abstraction

When analyzing AI-CPS, the dynamics of the system are highly non-linear, and the state space and action space are typically continuous with high dimensionality, which makes the analysis computationally intractable. Consequently, the abstraction technique, which reduces the system complexity, is necessary to shrink the state space and provide an approachable surrogate model for the system behavior.

The state abstraction $\phi(s) : S \rightarrow \bar{S}$ is a mapping from the original *concrete* state space S to a more compact *abstract* space $\bar{S}$, which preserves the intrinsic properties of the MDP but narrows down the size of the state space [46], [47], [48]. In general, there are three different types of abstraction methods for RL [49], [50]:

(i) Policy-irrelevant abstraction: concrete state s_1 and s_2 are considered in the same abstract state $\phi(s_1) = \phi(s_2)$, if they have the identical strategy on action selection $\pi^*(s_1) = \pi^*(s_2)$ following an optimal policy π^*.

(ii) Q value-irrelevant abstraction: $\phi(s_1) = \phi(s_2)$, if $Q^*(s_1, a) = Q^*(s_2, a), \forall a \in A$, where $Q^*(s, a)$ is an optimal state-action value functions that can lead the system to achieve the maximum cumulative reward.

(iii) Model-irrelevant abstraction: Not only focusing on a single reference value, but model-irrelevant abstraction also takes both the reward value and the transition dynamics into account. Namely, $\phi(s_1) = \phi(s_2), \forall a \in A$ if $R(s_1, a) = R(s_2, a)$ and $P(\phi(S)|s_1, a) = P(\phi(S)|s_2, a)$, where $P(\phi(S)|s, a)$ is transition distribution of state s over abstract state space $\phi(S)$.

These aforementioned abstraction approaches work well in the case of low-dimensional and discrete state space. Nevertheless, when encountering high dimensional continuous space, like in the context of AI-CPS, these methods are not applicable. The policy-irrelevant and Q-irrelevant abstraction cannot preserve the reward structure and transition dynamics. For model-irrelevant abstraction, it is hard to get deployed in the context of CPS due to the "curse of dimensionality" [51]. Notice that a poor abstraction can cause a huge approximation error, which means the abstract state space loses its generality and accuracy in representing the original state space. Therefore, to reduce the space of analysis while keeping capturing the characteristics of the AI-CPS, we propose *semantics-based abstraction*, which will be described in Section 3.2.1.

2.4 Ensemble Learning Methods

Ensemble methods have demonstrated their promising capabilities on various tasks, and the feature of modular redundancy design in CPSs naturally provides a playground for ensemble approaches. Therefore, we adopt the combination of two techniques onto AI-CPSs, which target building safety and reliability-enhanced control systems.

As a single machine learning model may fall short of handling complex data environments, ensemble learning methods aim to combine multiple models (sub-learners) to deliver a synergistic performance better than that achievable from any constituent model alone [21]. Typically, an ensemble model mainly consists of two parts: a group of *sub-learners*[1], which takes the same input, and a *strategy*, which decide how to aggregate the output of each sub-learner. The core of the strategy in ensemble methods is to compensate for the errors from a single sub-learner by fusing the output of other sub-learners such that the overall performance is improved.

A good ensemble model should follow several principles [52]. First, sub-learners should keep their independence, i.e., a decision from one learner cannot affect others. Moreover, the sub-learners should have diverse decision characteristics to explore the decision space as much as possible and compensate for the errors from others. For the strategy, it should fuse the output from each learner into a collective decision following a reasonable aggregation method [20].

In practice, ensembles have been considered as the state-of-the-art solution for many ML tasks, and a series of ensemble methods have been proposed, such as *Majority-Voting* [53], *Averaging* [54], *AdaBoost* [55], *Random Forest* [56]. In this paper, we choose Majority-Vote and Averaging as the baseline for the comparison. In this work, we use ensemble learning and ensemble methods interchangeably to represent the overall ensemble pipeline. To avoid misunderstanding, we use ensemble strategies to specifically indicate the aggregation methods.

2.5 Formal Specification

Formal specification is used to describe the desired properties a system should hold, such as safety and efficiency. In the context of CPS, *Signal Temporal Logic (STL)* [38] is a popular specification language that can express general temporal properties which characterize the desired CPS behavior over time.

1. We use "sub-learner/sub-controller" interchangeably.

For STL *syntax*, *atomic propositions* a and *formulas* φ are defined as follows, respectively:

$$\alpha ::\equiv f(\mathbf{v}) > 0$$
$$\varphi ::\equiv \alpha \mid \bot \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \, \mathcal{U}_I \, \varphi \mid \Box_I \varphi \mid \Diamond_I \varphi$$

Here, f is a function that maps a vector $\mathbf{v}$ to a real value, and I is a time interval $[a, b]$, where $a, b \in \mathbb{R}$ and $a < b$. We omit subscripts I for temporal operators if $I = [0, \infty)$. In this definition, $\alpha, \bot$ are atomic formulas. The function f can be any function that takes as input a number of values of variables and maps them to a real number. Usually, the variables are related to the system, and so α describes an atomic requirement that should be satisfied by the system states. As we mentioned, $\mathcal{U}, \Box$ and $\Diamond$ are typical modalities in temporal logic that denote *until*, *always* and *eventually* operators, respectively, and these operators allow STL to express more complicated properties than the atomic ones. For the *semantics* of the language, STL not only equips with the boolean semantics but also with quantitative robust semantics, which can quantify the degree of satisfaction/violation of the system in terms of the STL specification. There is a more comprehensive introduction of STL [38] for the interested reader.

3 Approach

In this section, we introduce our framework, SIEGE, for providing ensemble control analysis for the given AI-CPS.

3.1 Overview of SIEGE

Our framework SIEGE initiates an early exploratory study of ensemble methods on the control systems of AI-CPSs. We leverage a novel semantics-based abstraction to decompose the large state space and transparently inspect the status of the system. Four new ensemble methods are designed with the utilization of semantics information from the abstract model. We evaluate the newly proposed methods with comparisons to individual controllers and two classical ensemble methods by different specifications on each system. Figure 1 depicts the overview of SIEGE. At a high level, SIEGE contains three components: *DRL sub-controller training*, *semantics abstraction*, and *ensemble strategies*.

DRL sub-controller training. As sub-controllers play a foundation in the ensemble system, it is necessary to obtain reliable sub-controllers as constituents. In the context of multi-requirement control, we aim to train sub-controllers with diverse decision logic. To achieve this goal, we concentrate on training DRL agents with different DRL algorithms, as well as the reward function setting. (Section 3.3.1)

Semantics abstraction. We then perform model abstraction to empower the ensemble analysis. We construct the abstract model by semantics-based state abstraction, action abstraction, and transition abstraction. A refinement process is applied to minimize the semantics error between concrete states and abstract states. (Section 3.2)

Ensemble strategies. Finally, we use the ensemble strategies to aggregate the action of the low-level controllers. The ensemble strategies in our study are from three categories: classic strategy, semantics-based strategy and coordinator-based strategy. (Section 3.3.2)

3.2 Semantics-based Abstract Model Construction

In this subsection, we describe the steps of building the *semantics-based abstract model*.

Definition 1 (*Semantics-based abstract model*)**.** A semantics-based abstract model is a tuple $(\bar{S}, \bar{A}, \bar{T}, \bar{s}_0, \eta, \Theta)$, where $\bar{S}$ is a set of abstract states, $\bar{A}$ is a set of abstract actions, $\bar{T}$: $\bar{S} \times \bar{A} \to \bar{S}$ is a set of transition, $\eta : \bar{S} \times \bar{A} \times \bar{S} \to [0, 1]$ is the transition probability function which gives the probability to the transitions, $\bar{s}_0$ is the set of initial state, and Θ is the semantics space.

Semantics space $\Theta = \{\vartheta(s_1, \Phi), \ldots, \vartheta(s_n, \Phi)\}$ is a hyperspace that contains the semantics values that reflects the degree of satisfaction regarding the system specifications $\Phi = \{\varphi_1, \ldots, \varphi_m\}$. Based on the enclosed physical information, each concrete state can be mapped into the semantics space to reveal its satisfaction w.r.t the specifications. We will introduce more details about semantics and the mapping mechanism in the following sections.

To construct the model, we first conduct a large number of simulations of the CPS with respective sub-controllers, which aim to collect enough characteristics of the system. Taking the simulation data and the AI-CPS as input, we then profile the system to extract information, *i.e.*, semantics, actions, and transitions, which can represent the system behavior. The input to the abstraction is a set of traces that records the semantics information, actions, and transitions taken during the simulation. Then, we abstract the semantics, action, and transitions to construct our semantics-based abstract model. In the remainder of the subsection, we describe the detailed procedure of *state abstraction*, *action abstraction*, *transition abstraction*, and the *refinement procedure*.

3.2.1 State Abstraction

As mentioned in Section 2.3, existing commonly used state abstraction methods fall short of handling the high-dimensional and continuous state space of AI-CPS. In particular, the policy-irrelevant and Q-irrelevant abstractions are not precise enough to reflect the unique behavioural characteristics of the controllers, and the model-irrelevant is a desirable solution but is hard to implement in practice. Therefore, we propose a new type of abstraction method, i.e., semantics-based abstraction, to tackle this challenge.

Let us first define what semantics is.

Definition 2 (*Semantics*)**.** The semantics $\vartheta \in \mathbb{R}^J$ of a concrete state $s \in \mathbb{R}^n$ is defined as follows:

$$\vartheta(s, \Phi) = (\theta_s^{\varphi_1}, \ldots, \theta_s^{\varphi_J}),$$

where $\Phi = \{\varphi_1, \ldots, \varphi_J\}$ is a list of specifications and $\theta_s^{\varphi} \in \mathbb{R}$ represents the degree of satisfaction of the system with respect to the specification φ.

Intuitively, the semantics reflects the status of the system with respect to the desired specifications. The advantages of using semantics to augment the concrete states are two-fold. First, it can represent the system behaviors in the case of *multiple requirements*. For industrial-scale CPSs, multiple requirements arise inevitably during the design and development of the complex system. Here, semantics is a suitable measurement for the degree of satisfaction of the

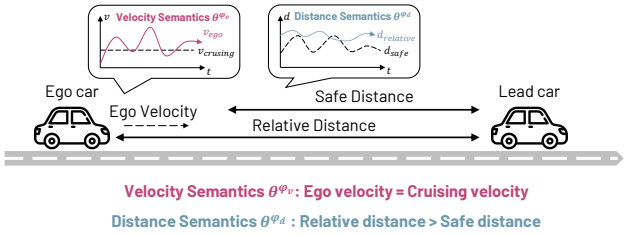


Fig. 3: An example of the semantics of the ACC system

system with respect to multiple requirements. In addition, semantics can be used as an *abstract representation* of the original state space. Typically, the state space of the system is high-dimensional, which is hard to analyze. Semantics, as a low-dimension representation of the system, can effectively catch the system behaviors triggered by the huge number of input signals.

We use Example 1 to illustrate how the semantics is used to describe system behavior.

Example 1. Let us consider an autonomous vehicle with the adaptive cruise control (ACC) system, shown in Figure 3. ACC receives the information from cameras, lidar, and radar sensors and outputs an acceleration or deceleration command to manipulate the vehicle's speed. There are two requirements for the system: (1) *distance requirement* φ_{dist}, which means the vehicle must maintain a safe distance from the leading car; (2) *velocity requirement* φ_{velo}, which says velocity should be close to the user-set cruising speed when the distance requirement is satisfied. The degree of satisfaction of the distance control is obtained by: $\theta_{dist} = \tanh(d_{rel} - d_{safe})$, where $(d_{rel} - d_{safe})$ measures a quantitative value about to what extent the safety distance requirement is satisfied or violated. Namely, a large negative value represents a serious violation of distance specification where the relative distance between two cars is much less than the required safety distance, which also indicates the potential risk of an accident. Likewise, for velocity specification, the degree of satisfaction θ_{vel} is measured by the deviation from the user-set velocity $(v_{ego} - v_{set})$. A positive value for the degree of satisfaction of velocity will be generated if the velocity deviation is within a specific threshold; otherwise, a negative value is given to represent a violation. Suppose, in state s, the degree of satisfaction of distance requirement is 15, and the degree of satisfaction for velocity is -7 (negative number represents the violation of the requirement). The semantics of ACC is $\vartheta(s, \Phi) = (\theta_s^{\varphi_{dist}}, \theta_s^{\varphi_{velo}}) = (15, -7)$.

Remark 1. As the degree of satisfaction aims to reveal the satisfaction/violation of the system behavior w.r.t. the target system specifications, the developers should have clear specifications from the documentation of their applications regarding safety, stability, operation efficiency, etc. Then, to properly and systematically define the degree of satisfaction, we recommend the developers identify the reference values of violation/satisfaction regarding each specification. Namely, the system output values/conditions can indicate whether the system is operating normally or problematically. Next, the developers can use mathematical functions (e.g. Hyperbolic tangent) to normalize the target system outputs within a unified scale such as $[-1, 1]$. Note that, to maintain the sensitivity and preciousness of the degree of satisfaction, we suggest normalizing the values with clipping mechanisms; namely, the boundary values that denote serious satisfaction/violation and any values that exceed such boundaries should be regularized. This approach is vital, especially for the system outputs with large available ranges. Once the regularized values are settled, the developers should carefully check that any positive/negative values should explicitly represent a certain degree of satisfaction/violation with no cross-over areas.

After the semantics is obtained, each concrete state s_i in the semantics space is a vector $(\theta_{s_i}^1, \ldots, \theta_{s_i}^J)$. Such a granular representation is still impractical for further analysis since each dimension of the state is continuous, which means the state space could be prohibitively large. Thus, we perform state abstraction to group the states that are close.

We perform two steps to construct the abstract states. First, since the scale of different requirements might be different, we perform normalization to normalize the data into a unifying scale. Specifically, the degree of satisfaction is normalized to $[-1, 1]$. A positive (negative) number indicates the case of satisfaction (violation), and a higher/lower number represents the requirement that is more satisfied (violated) by the system.

Then, we partition the J-dimensional space into $\Pi_{j=1}^{J} K_j$ segments, such that there are K_j segments on each dimension:

$$d_i^j = \Big[lb_i^j,\ ub_i^j\Big],$$

where d_i^j is the i-th segment on j-th dimension, lb_i^j and ub_i^j are the lower bound and the upper bound of the segment.

The space partition problem is turned into an optimization problem, which is stated as follows.

$$\begin{aligned} \text{maximize} \quad & ub_i^j - lb_i^j \\ \text{subject to} \quad & d_{\text{MIN}}^j \le (ub_i^j - lb_i^j) \le d_{\text{MAX}}^j, \\ & |\hat{s}_i^j| \ge n_{\text{MIN}}^j, \\ & \text{MEAN}\Big\{\theta_{\hat{s}}^j - \mathbb{E}[\theta_{\hat{s}}^j]\Big\} < e_{\text{MEAN}}^j, \\ & \text{MAX}\Big\{\theta_{\hat{s}}^j - \mathbb{E}[\theta_{\hat{s}}^j]\Big\} < e_{\text{MAX}}^j, \end{aligned} \tag{1}$$

where d_{MIN}^j and d_{MAX}^j are the minimum and maximum lengths of a segments on j-th semantics dimension, $\hat{s}_i^j = \{s|\ \theta_s^j \in d_i^j\}$ is the set of concrete states whose semantics value θ_s^j falls in interval d_i^j, n_{MIN}^j is the minimum number of concrete states in the segments on j-th dimension, e_{MEAN}^j and e_{MAX}^j are the predefined average and maximum tolerances of the abstraction errors on j-th dimension. Equation 1 ensures each segment would contain enough concrete states while maintaining a low abstraction error.

In this way, the concrete states with similar values across all semantics dimensions are mapped to the same abstract state:

$$\bar{s} = \Big\{s_i|\ \theta_{s_i}^1 \in d_{\_}^1 \wedge \cdots \wedge \theta_{s_i}^J \in d_{\_}^J, d_{\_}^j \in \{d_1^j, \ldots, d_{K_j}^j\}, \\ j \in \{1 \ldots J\}\Big\}.$$

We denote this abstract semantics state space as $\bar{S}$. Moreover, the semantics of an abstract state is

$$\vartheta(\bar{s}, \Phi) = (\theta^{\varphi_1}_{\bar{s}}, \ldots, \theta^{\varphi_J}_{\bar{s}}), \theta^{\varphi}_{\bar{s}} = \text{MEAN}\Big\{\theta^{\varphi}_{s_i} |\ s_i \in \bar{s}\Big\}.$$

Example 2. After normalization, say, the concrete semantics of ACC in Example 1 is $s = (0.65, -0.38)$. After solving Equation 1, a possible partition is

$$d^1_1 = [-1, -0.4], d^1_2 = [-0.4, 1.0]$$
$$d^2_1 = [-1, 0.3], d^2_2 = [0.3, 1].$$

The abstract states would be

$$\bar{s}_1 = \Big\{s_i |\ \theta^1_{s_i} \in d^1_1 \wedge \theta^2_{s_i} \in d^2_1\Big\}, \bar{s}_2 = \Big\{s_i |\ \theta^1_{s_i} \in d^1_1 \wedge \theta^2_{s_i} \in d^2_2\Big\},$$
$$\bar{s}_3 = \Big\{s_i |\ \theta^1_{s_i} \in d^1_2 \wedge \theta^2_{s_i} \in d^2_1\Big\}, \bar{s}_4 = \Big\{s_i |\ \theta^1_{s_i} \in d^1_2 \wedge \theta^2_{s_i} \in d^2_2\Big\}.$$

As a result, the concrete semantics would belong to $\bar{s}_3$.

Remark 2. A good abstract model should be a *succinct* and *precise* model at the same time. Succinctness means the number of states should be small, while preciseness implies that the semantics error between the abstract states and concrete states is small. Note that there is a trade-off between these two requirements: smaller abstract space, i.e., less abstract states, may lead to greater error in terms of semantics values, and vice versa.

Thus, in Equation 1, we utilize three parameters to modify the preciseness of the abstraction: 1) the minimum distance between two segments d_{MIN}, 2) the minimum number of concrete states contained in an abstract state n_{MIN}, and 3) maximum distance between two segments d_{MAX}. The minimum distance ensures two segments should reflect different semantics levels, and the minimum number of enclosed concrete states prevents the occurrence of redundant segments which only contain a few concrete states. In addition, we notice that some special concrete states indicate critical circumstances, namely, extreme violation or satisfaction with specifications. Merging these concrete states into the neighbour abstract states may cause a significant error in semantics values; thus, we use the third parameter, maximum distance, to split these particular states as individual abstract states. The values of the semantics of an abstract state are computed as the average semantics values from all enclosed concrete states.

Note that the parameters of abstract error tolerances can be modified regarding the precision and the model size requirements; namely, a tighter error threshold can correspondingly bring a more precise model but larger abstract state space. The first tolerance, e^j_{MEAN}, for the average error is to assure the representative and the overall accuracy of the abstraction. Ideally, each abstract state should represent a group of concrete states with similar semantics meanings, so the average error of each semantics should be relatively small to prevent an abstract state from drifting from its actual physical status. Similarly, for e^j_{MAX}, if a large average error appears, the corresponding model may over-compress the state space where excessive concrete states with notably different semantics have been aggregated to an identical abstract state.

3.2.2 Action and Transition Abstraction

For *action abstraction*, we first leverage the uniform partition to split the action space into identical-length intervals. In other words, the k-dimensional action space is split into m^k sub-space where there are m segments with an identical length on each dimension. Then, we transform the concrete action to the interval that the action belongs to, which is the corresponding abstract action. That is to say, for a concrete action $a \in [u, l]$, the abstract action $\bar{a} = [u, l]$. We denote that abstract action space as $\bar{A}$.

With abstract state space $\bar{S}$ and abstract action space $\bar{A}$, we construct the *abstract transition space* as $\bar{T} : \bar{S} \times \bar{A} \to \bar{S}$, which are a set of concrete transitions between concrete states. In particular, if there exists a concrete transition between $s \in \bar{s}$ and $s' \in \bar{s}'$, then an abstract transition is set up accordingly between the abstract states $\bar{s}$ and $\bar{s}'$, where s and s' are concrete states, and $\bar{s}$ and $\bar{s}'$ are abstract states An abstract transition contains all the concrete transitions that share the same starting and destination abstract state. Moreover, we augment the abstract model with a transition probabilities function. Specifically, we use $\eta(\bar{s}, \bar{a}, \bar{s}')$ to denote the conditional probability of visiting $\bar{s}'$ given the current state $\bar{s}$ and the current action $\bar{a}$, and $\sum_{\bar{s}' \in \bar{S}} \eta(\bar{s}, \bar{a}, \bar{s}') = 1$. The transition probability is defined as:

$$\eta(\bar{s}, \bar{a}, \bar{s}') = \frac{|\{(\bar{s}, \bar{a}, \bar{s}') \in \bar{T} \mid \bar{s} \in \bar{S}, \bar{a} \in \bar{A}, \bar{s}' \in \bar{S}\}|}{|\{(\bar{s}, \bar{a}, \_) \in \bar{T} \mid \bar{s} \in \bar{S}, \bar{a} \in \bar{A}\}|}.$$

In other words, the probability is computed as the number of concrete transitions from $\bar{s}$ to $\bar{s}'$ executing action $\bar{a}$ over the overall outgoing transitions from $\bar{s}$ given action $\bar{a}$.

3.2.3 Refinement of State Abstraction

Algorithm 1: Refinement of Semantics-based Abstraction

Input: A list of specifications $\Phi = \{\varphi_1, \ldots, \varphi_J\}$, a set of concrete states S, the maximum and minimum lengths of the segment d_{MAX} and d_{MIN}, the minimum number of concrete states in a segment n_{MIN}, the error thresholds $e_{\texttt{cur}}$ and $e_{\texttt{pred}}$, and the reduction level threshold $r_{\texttt{cur}}$

Output: An abstract state space $\bar{S}$

Let `refined` $\leftarrow$ False, $\bar{S} \leftarrow \emptyset$,
while `refined` $=$ *False* **do**
  for $j \in \{1, \ldots, J\}$ **do**
    $d^j \leftarrow$ SEGMENTPARTITION$(S, \theta^{\varphi_j}, d_{\text{MAX}}, d_{\text{MIN}}, n_{\text{MIN}})$
  $D \leftarrow \{d^1, \ldots, d^J\}$
  $\bar{S} \leftarrow$ STATEMAPPING(D)
  $\langle \varepsilon_{\texttt{cur}}, \varepsilon_{\texttt{pred}} \rangle \leftarrow$ COMPUTEERROR$(\bar{S}, S)$
  if $\langle \varepsilon_{\texttt{cur}}, \varepsilon_{\texttt{pred}} \rangle > \langle e_{\texttt{cur}}, e_{\texttt{pred}} \rangle$ and REDUCTION_LEVEL $< r_{\texttt{cur}}$ **then**
    Update $d_{\text{MAX}}, d_{\text{MIN}}, n_{\text{MIN}}$ **else**
    `refined` $\leftarrow$ True
return $\bar{S}$

We propose an algorithm for refining the state abstraction, which targets at reducing the semantics error between

the abstract states and the corresponding enclosed concrete states. The algorithmic schema of the refinement procedure is depicted in Algorithm 1.

Given a list of specifications Φ, a set of concrete state S, the maximum and minimum lengths of the segment d_{MAX} and d_{MIN}, and the minimum number of concrete states in a segment n_{MIN}, the refinement procedure produces the refined abstract space $\bar{S}$. Essentially, the refinement is a while-loop to adjust the partition parameters (Line 2-12), and the workflow is summarized as follows:

- For each dimension in the semantics space, it conducts partition by solving Equation 1, and generates corresponding segments d^j (Line 3-4). After the partition, a coarse abstract state space $\bar{S}$ is built (Line 5-6).
- Then, it computes the semantics error $\varepsilon_{\texttt{cur}}$ and $\varepsilon_{\texttt{pred}}$ (Line 7), which have the following meaning:
  - The current state semantics error $\varepsilon_{\texttt{cur}}$ is computed as

$$\text{MEAN}\Big\{\theta^{\varphi}_{\hat{s}} - \mathbb{E}[\theta^{\varphi}_{\hat{s}}]\Big\},$$

    where $\hat{s}$ are the group of concrete states inside the abstract state $\bar{s}$. $\varepsilon_{\texttt{cur}}$ describes the semantics status of the current abstract state, i.e., whether the abstract semantics represents the concrete semantics precisely.
  - The prediction semantics error $\varepsilon_{\texttt{pred}}$ is

$$\text{MEAN}\Big\{\theta^{\varphi}_{\hat{s}'_1} - \mathbb{E}[\theta^{\varphi}_{\hat{s}'_1}], \ldots, \theta^{\varphi}_{\hat{s}'_n} - \mathbb{E}[\theta^{\varphi}_{\hat{s}'_n}]\Big\},$$

    where $\hat{s}'$ is the set of the concrete destination states enclosed by the abstract state $\bar{s}'$ following the abstract transition $\bar{s} \times \bar{a} \to \{\bar{s}'_1, \ldots, \bar{s}'_n\}$. We are concern about the semantics error of the incoming state as it is utilized in the guidance of the action selection, which will be described in Section 3.3.2.
- The computed $\varepsilon_{\texttt{cur}}$ and $\varepsilon_{\texttt{pred}}$ are compared with the predefined threshold $e_{\texttt{cur}}$ and $e_{\texttt{pred}}$ (element-wise), and REDUCTION_LEVEL, which is the rate of the number of states after and before abstraction, is compared with the reduction level threshold $r_{\texttt{cur}}$ (Line 8). This leads to two cases:
  - If the errors exceed the threshold, which means the preciseness and succinctness of the abstract model are not satisfied, we adjust the parameter $d_{\text{MIN}}, d_{\text{MAX}}$, and n_{MIN}, to refine the model, based on a pre-generated list of 3-tuple (Line 9). Each tuple is a sequence of the three parameters, and each element is generated as follows: d^j_{MIN} is a set of values from 0.01 to 0.1 with the step size of 0.001, d^j_{MAX} is a set of value from 0.005 to 0.1 with the step size of 0.005, and n^j_{MIN} is a set of value from 0.1% to 10% with the step size of 0.1%. The list is a permutation of the values of three parameters. The automatic update procedure iterates the list and picks an element from it.
  - Otherwise, we set `refined` to True.
- The algorithm returns the refined abstract space $\bar{S}$ (Line 12).

Remark 3. Our semantics-based abstraction method not only narrows the huge state space of CPSs but also provides a human-interpretable way to describe the status of a system in terms of the degree of satisfaction with different specifications. Namely, despite the numerous system information collected by sensors, designers can understand the system characteristics and controller behaviors intuitively and transparently. More attention can be called regarding the safety and functional requirements during the construction of control systems in AI-CPSs.

Remark 4. Algorithm 1 is guaranteed to converge even if the error parameters $e_{\texttt{cur}}$ and $e_{\texttt{pred}}$ are set to values very close to zero. Because in the worst case, the algorithm returns the original concrete model, where the errors are zero.

3.3 Ensemble Control

In this subsection, we introduce two parts of the ensemble framework: *sub-controllers*, which provide the fundamental control logic of the system, and the *ensemble strategy*, which decides how to combine the output of the sub-controllers. The portrait of the ensemble control is depicted in Figure 4.

3.3.1 Sub-controller

For sub-controllers, in order to choose the optimal decision in different scenarios, we aim to train them towards diverse decision logic and cover different requirements. Towards this goal, we mainly focus on the *reward function* setting and diverse *DRL algorithm* selection.

- Unlike traditional control systems, the decision-making strategy of DRL controllers is predominantly determined by their *reward functions* [57]. The reward function decides whether an action taken by the agent in a state is good or not. In the context of multi-requirement control, we set the reward function as the weighted sum of the degree of satisfaction to the specification, which leads the sub-controllers to satisfy all the requirements at the same time. Specifically, the reward $\mathcal{R}$ for the sub-controller is defined as:

$$\mathcal{R} = \sum_{i=1}^{j} w_i \theta^{\varphi_i}_{s},$$

  where $w_i \in [0, 1]$ is the weight parameter and $\theta^{\varphi_i}_{s}$ is the degree of satisfaction w.r.t. the specification φ_i. By tuning the weights w_i, we can adjust the behavioural preference of the DRL agent on the specifications, especially when dealing with multiple control objectives. For instance, in Example 1, the ACC system has two operation requirements φ_{dist} and φ_{velo}. Assuming the distance requirement is more important, w_{dist} could be set as 0.9, and w_{velo} could be set as 0.1, in order to train the sub-controller towards the compliance of distance requirement.
- Besides the reward functions, we choose different *DRL algorithms*, e.g., DDPG, TD3, and SAC, to expand the diversity of the policies since even with identical reward functions, different DRL algorithms are with distinct control strategies. For instance, DDPG and TD3 agents use a deterministic policy actor to generate actions that maximize the long-term cumulative reward. In contrast, a SAC agent produces actions from a stochastic Gaussian actor according to a probability distribution. An agent with a stochastic algorithm is expected to present better robustness and stability under environments with high randomness [58].

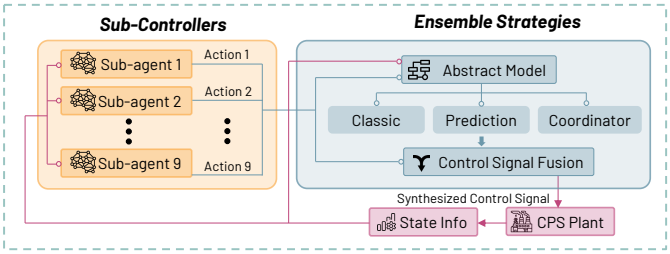


Fig. 4: Ensemble Control System

To conclude, we construct multiple DRL controllers with diverse behaviour characteristics so that the combination of the controllers can balance out their individual weaknesses while taking their own advantages. The diversity of the constituent controllers builds the foundation of the ensemble framework.

3.3.2 Ensemble Strategy

As the cornerstones of the ensemble framework, constituent DRL sub-controllers output a set of candidate actions following their own policies. The next challenge is how to merge these actions in order to deliver a synergistic impact on the required task. In SIEGE, *Ensemble strategy*, which is based on the constructed semantics-based abstract model, aims to provide the control action fusion so that the ensemble controller can outperform the individual controllers. In particular, we propose six ensemble strategies to explore the potential capabilities of ensemble control systems in AI-enabled CPS, and they are of three categories:

- Classic strategy: *Majority Voting* and *Average*.
- Semantics-based strategy: *Top-1 Semantics Prediction* and *Top-k Semantics Prediction*
- Coordinator-based strategy: *Coordinator* and *Coordinator (with prediction)*

In the following, we denote the set of actions at time t from l sub-controllers (l is the number of sub-controllers) as $\{a_1^t, \ldots, a_l^t\}$, and the output of the ensemble strategy at time t as act^t. We omit the superscript t for simplicity.

Classic Strategy. The classic strategy includes Majority Voting and Average.

- For majority voting, the final control output is the mean value of all concrete actions output from the sub-controllers that belong to the abstract action with maximum votes:

$$\mathsf{act} = \text{MEAN}\Big\{a |\ a \in \bar{a}_{vote}\Big\},$$

where $\bar{a}_{vote}$ is the abstract action with maximum votes, a is the concrete actions that belongs to $\bar{a}_{vote}$ and output from the sub-controllers.

- Average strategy takes the average value among actions from all sub-controllers as the final output:

$$\mathsf{act} = \text{MEAN}\Big\{a_1, \ldots, a_l\Big\},$$

where l is the number of constituent controllers enclosed by the ensemble framework.

Semantics-based Strategy. To tackle the multi-requirement control challenge, we leverage the semantics information on the semantics-based abstract model.

- *Top-1 Semantics Prediction.* We rank the action by the sum of the semantics of incoming abstract states and take the mean of concrete action from the sub-controllers in the top-1 abstract action as the output.

$$\mathsf{act} = \text{MEAN}\Bigg\{a | a \in \bar{a}_{top1}, \bar{a}_{top1} = \underset{\bar{a}\in\{\bar{a}_1,\ldots,\bar{a}_l\}}{\text{argmax}} \Big\{\sum_{i=1}^{n} \|\vartheta(\bar{s}_i', \Phi)\| \ | \ \bar{s} \times \bar{a} \to \{\bar{s}_1', \ldots, \bar{s}_n'\}\Big\}\Bigg\},$$

where a is the concrete actions from the sub-controller that belong to $\bar{a}_{top1}$, $\bar{a}_{top1}$ is top-1 abstract action, $\bar{s}$ is the current abstract state, $\bar{a}$ is the abstract action, $\{\bar{s}_1', \ldots, \bar{s}_n'\}$ is the set of incoming states after taking action $\bar{a}$ in current state $\bar{s}$ (there is typically a set of incoming state after taking action), $\{\bar{a}_1, \ldots, \bar{a}_l\}$ is the abstract actions of l sub-controllers, $\bar{s}'$ is the incoming abstract state after taking action $\bar{a}$, and $\vartheta(\bar{s}', \Phi)$ is the semantics of the incoming state $\bar{s}'$. An action is neglected in the ranking if a negative number exists in the predicted semantics, as a negative value indicates a violation of a specification.

- *Top-k Semantics Prediction.* Similar to the above strategy, we also rank the action. However, we take the mean of concrete actions that belong to the top-k abstraction actions as the output action.

Coordinator strategy. Besides the deterministic strategies introduced above, we design stochastic strategies, which are able to explore other potential optimal actions and perform an adaptive selection. A hierarchical architecture in reinforcement learning has demonstrated its promising abilities to decompose a long-term decision-marking task into simpler sub-tasks [59]. Therefore, besides the single-layer ensemble methods introduced in classic strategies and semantics-based strategies, we further propose a two-layer hierarchical ensemble structure to extensively explore the capability of hierarchical ensemble control in the context of AI-CPSs. Namely, instead of using rule-based deterministic strategies to aggregate the control outputs from constituent DRL sub-controllers, we design stochastic strategies, which are able to explore other potential optimal actions and perform an adaptive selection. Particularly, inspired by the exploration and exploitation principle of reinforcement learning, we utilize a "high-level" DRL agent, i.e., coordinator, for the action selection.

In contrast to the constituent DRL controllers in the first layer, a new DRL agent, the coordinator, is trained to select the proper action from the first layer controllers. Namely, the DRL coordinator does not output concrete control signals for the system actuator but selects the output from one of the DRL sub-controllers as the final output. The reward function for the coordinator is slightly different from the sub-controllers. All specifications have equal weights in the reward function, as we consider these safety-related specifications to be equivalently important during the practical operation. In addition, we set a huge penalty for the violation of any specifications to make the coordinator select the optimal action to prevent any violations. Note that we have tried this reward structure on the DRL sub-controllers but failed to obtain an optimal controller which can tackle

the multi-objective optimization regarding different system specifications. This is one of the motivations for leveraging ensemble methods in AI-CPSs, as we have mentioned in Section 1.

Particularly, inspired by the exploration and exploitation principle of reinforcement learning, we utilize a "high-level" DRL agent, i.e., *coordinator*, for the action selection.

- *Coordinator.* This strategy contains a DRL agent to perform an assessment and pick the optimal action for the task. Given the system state as input, the task of the coordinator is to select an optimal action from the sub-controllers.
- *Coordinator (with prediction).* The second coordinator strategy takes additional information on the predicted semantics. The predicted semantics, from the semantics-based abstract model, aims to enhance the collaboration ability of the higher-level DRL coordinator.

4 Evaluation

This section presents the experimental evaluation of our ensemble framework, including research questions, subject systems, experiment setup, and evaluation results.

4.1 Research Questions

4.1.1 RQ1: How is the semantics abstraction in terms of succinctness and preciseness?

When dealing with industry-level CPS with large state and action space, a *succinct* and *precise* abstract model helps reduce the complexity and enhance the explainability of a system for further process. However, the abstraction procedure may lead to the loss of preciseness and may not produce a succinct model.

In this RQ, we aim to investigate the succinctness and preciseness of our abstraction method in terms of system semantics. We check the succinctness by comparing the reduction rate of the number of states before and after the abstraction. To assess the preciseness, we measure the mean and maximum errors between the semantics of the concrete and abstract models.

4.1.2 RQ2: To what extent can the abstract model be helpful for predicting the incoming semantics?

One of our new ensemble methods is leveraging the prediction of the semantics of the very next states. We want to investigate whether the predicted semantics is accurate enough for further analysis. To answer this RQ, we sample new traces and compare the predicted semantics on the abstract model with the ground truth semantics on the concrete traces.

4.1.3 RQ3: Can the ensemble-based controllers be competitive to and outperform the standalone controller?

One of the main goals of constructing an ensemble-based control system is to compensate for the deficiencies of the single controller under multiple requirements. Therefore, we intend to compare the performance of the single controller and two ensemble-based controllers with traditional methods. In particular, the performance is measured by multiple specifications, which will be detailedly introduced in Section 4.4.3.

4.1.4 RQ4: Do our newly proposed ensemble strategies perform better than traditional ensemble methods?

As the classic ensemble methods lack guidance while selecting the candidate actions, we aim to investigate if the semantics prediction can indeed enhance the ensemble strategy. Besides the deterministic methods, we also target to study the capability of DRL-heretical ensemble methods. In this RQ, we assess the performance of the newly proposed ensemble methods, *i.e.*, the prediction approaches and the DRL-coordinator approaches. We compare them with the traditional ensemble methods and the individual controllers, where the same metrics in RQ3 are used for evaluation.

4.2 Studied Systems and Specifications

We next introduce the five CPS systems that we studied for our framework and the specifications.

As we intend to investigate the effectiveness of our approach in practical applications, the systems for experiments must reflect the industry-level complexity, namely, large continuous state space and action space with multiple operational requirements. In addition, a candidate system should be released by official industry or research institutes with detailed documentation of system goals and specifications. Therefore, we can semantically abstract the system, evaluate the controller performance and construct additional new RL controllers with various behaviours. The five representative systems we select for experiments cover multiple industrial domains such as autonomous driving, powertrain and robotics. Three out of five systems come with traditional built-in controllers, like Model Predictive Control (MPC) and PID, and the other two systems are released with RL controllers.

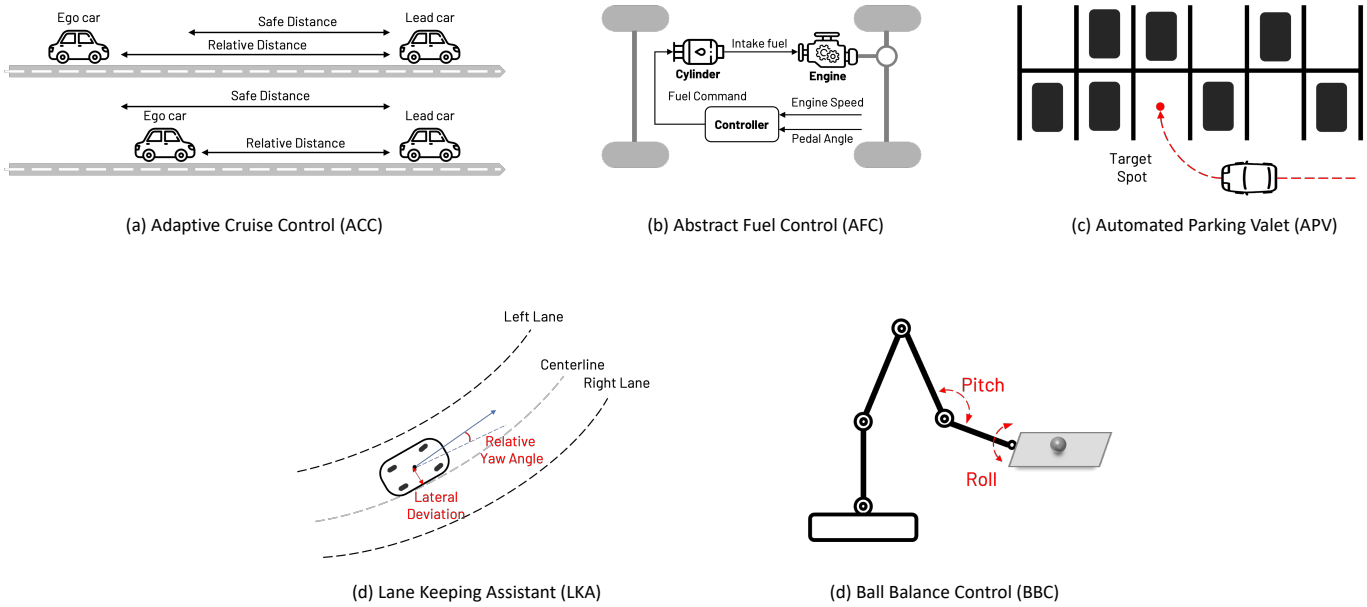
4.2.1 Systems

We collect five subject CPSs to conduct our experiments. These systems cover different industry domains with various specifications and control requirements. We consider the experimental results of these systems can bring a comprehensive understanding of our framework on diverse AI-CPS. Table 1 briefly introduces the tasks and industrial domains of the five CPS systems that studies. Specifically, Column # Blocks shows the number of blocks of each system in the Simulink Model, which can represent the relative complexity of a CPS system. These systems have been used in the research community [13], [60], [61], [62], [63], [64], [65], [66]. Some of the systems, such as AFC are developed by the collaboration between industry companies and MathWorks. We believe these systems can reflect the challenges and characteristics of industry-level CPSs and are good benchmarks for academic research and development. The instruction of functionalities and specifications of each system is presented in the following sub-sections.

Adaptive Cruise Control (ACC). ACC is originally released by MathWorks [67]. As we briefly introduce in Section 3.2.1, this system is programmed to keep the relative distance d_{rel} between two vehicles greater than a safe distance d_{safe} by controlling the acceleration of the ego car a_{ego}. When the distance is secured, the ego car should maintain a user-set cruising velocity v_{set}. The movement of the lead car

TABLE 1: The collected CPS with target domains, introductions, and number of blocks.

Subject CPS	Domain	Description	#Blocks
Adaptive Cruise System (ACC)	Driving assistant	Maintain a safe distance and operate within the target speed	297
Abstract Fuel Control (AFC)	Powertrain	Maintain a reference air-to-fuel ratio inside the cylinder	281
Lane Keeping Assistant (LKA)	Driving assistant	Keep a vehicle traveling along the centerline of the lanes	421
Automated Parking Valet (APV)	Parking assistant	Park a vehicle on a target spot without collision	1497
Ball Balance Control (BBC)	Robotics	Balance a ball at the center point of a plate	1225



(a) Adaptive Cruise Control (ACC)

(b) Abstract Fuel Control (AFC)

(c) Automated Parking Valet (APV)

(d) Lane Keeping Assistant (LKA)

(d) Ball Balance Control (BBC)

Fig. 5: Subject CPS illustration

is controlled by the acceleration of the lead car a_{lead}. The safe distance is dynamically changed based on the relative velocity between two vehicles.

Abstract Fuel Control (AFC). AFC is an air-fuel control system released by Toyota [61]. As illustrated in Figure 5 (b), this system is designed to maintain the air-to-fuel ratio AF from a reference value AF_{ref} inside a powertrain cylinder. The external inputs are pedal angle θ_{in} and engine speed ω_S, where θ is a pulse train signal which defines a steady condition and an edge condition to the system. For edge conditions (either rising or falling), the normalized error of $\mu = \frac{|AF - AF_{ref}|}{AF_{ref}}$ should be less than 0.05; and for the steady condition between two edge events, μ has a tighter constraint of 0.02. These two conditions come out cyclically; the controller should regulate the error value μ under assigned thresholds, respectively.

Lane Keeping Assistant (LKA). This system is collected from MathWorks [68]. LKA measures the lateral deviation d_{lat} and relative yaw angle θ_{yaw} between the vehicle and the centerline of the road and keeps the vehicle travelling along the centerline by adjusting the front steering angle θ_{steer}. The goal of the system is to drive both lateral deviation and yaw angle close to zero.

Automated Parking Valet (APV). We collect APV from MathWorks [69]. Under a parking lot environment, APV parks the vehicle in a target empty spot from a random initial position. The car moves at a constant speed, and the output signal from the controller is the steering angle θ_{steer}. Successful parking is counted when the position error d_{car} and the orientation error θ_{head} are lower than specific thresholds with no collision $(d_{lidar} > 0)$ occurred during the parking maneuver.

Ball Balance Control (BBC). This system is constructed based on a seven degree-of-freedom (DOF) Kinova Gen3 robot [70], [71]. The manipulator is tasked to balance a small ball at the center point of a flat plate. The system has certain thresholds for the position error of the ball d_{ball} from the plate center and the orientation of the plate itself θ_{plate}; and the controller outputs the torque commands τ_1, τ_2 of two joints to manipulate the plate from pitch and roll axes.

4.2.2 Specifications

Table 2 summarizes the . Specifically, we classify specifications into two categories: **Major** and **Minor** according to their importance.

- **Major**. Such as distance and velocity in ACC, deviation and angle in LKA; each system has unique specifications regarding safety and efficiency. We mark these specifications with a star* Table 2, 7, 8 to denote them as major ones.
- **Minor**. Unlike the unique major specifications, all five systems share three minor specifications (marked with a circle$^\circ$) to represent additional properties:
  - ***Stability***: stability measures how stable the system is during the entire simulation; particularly, a system is

TABLE 2: The STL specifications of the benchmark CPS. A star⋆ denotes an essential specification; and a circle◦ denotes a minor specification; these notations also apply to Table 7, 8.

Spec.	Distance⋆	Velocity⋆	Stability◦	Resilience◦	Consistency◦
ACC	$I=[0,30]$ $\varphi_1 \equiv \mathtt{d_{rel}} \geq \mathtt{d_{safe}}$ $\varphi \equiv \square_I(\varphi_1)$	$I=[0,30]$ $\varphi_2 \equiv \varphi_1 \rightarrow (\mathtt{v_{ego}} = \mathtt{v_{set}})$ $\varphi \equiv \square_I(\varphi_2)$	$I=[0,30]$ $\varphi_3 \equiv \mathtt{d_{rel}} \geq (\mathtt{d_{safe}} + 0.2)$ $\varphi \equiv \square_I(\varphi_3)$	$I=[0,30], I'=[0,1]$ $\varphi_{41} \equiv \neg\varphi_3, \varphi_{42} \equiv \varphi_3$ $\varphi \equiv \square_I(\varphi_{41} \rightarrow \diamond_{I'}\varphi_{42})$	$I=[0,30]$ $\varphi_5 \equiv \lvert a^{\Delta}_{ego}\rvert \leq 1$ $\varphi \equiv \square_I(\varphi_5)$
Spec.	Steady⋆	Edge⋆	Stability◦	Resilience◦	Consistency◦
AFC	$I=[0,30]$ $\varphi_{11} \equiv (\theta = 8.8) \wedge \diamond_{(0,\epsilon)}(\theta = \alpha)$ $\varphi_{12} \equiv (\theta = \alpha) \wedge \diamond_{(0,\epsilon)}(\theta = 8.8)$ $\varphi_1 \equiv \varphi_{11} \vee \varphi_{12} \rightarrow (\mu < 0.02)$ $\varphi \equiv \square_I(\varphi_1)$	$I=[0,30]$ $\varphi_{21} \equiv \varphi_{11} \vee \varphi_{12}$ $\varphi_2 \equiv \neg\varphi_{21} \rightarrow (\mu < 0.05)$ $\varphi \equiv \square_I(\varphi_2)$	$I=[0,30]$ $\varphi_{31} \equiv \neg\varphi_{21} \rightarrow \mu < 0.018$ $\varphi_{32} \equiv \varphi_{21} \rightarrow \mu < 0.045$ $\varphi_3 \equiv \varphi_{31} \wedge \varphi_{32}$ $\varphi \equiv \square_I(\varphi_3)$	$I=[0,30], I'=[0,1]$ $\varphi_{41} \equiv \neg\varphi_3$ $\varphi_{42} \equiv \varphi_3$ $\varphi \equiv \square_I(\varphi_{41} \rightarrow \diamond_{I'}\varphi_{42})$	$I=[0,30]$ $\varphi_5 \equiv \lvert \dot{m}^{\Delta}_c\rvert \leq 0.3$ $\varphi \equiv \square_I(\varphi_5)$
Spec.	Lateral Deviation⋆	Relative Yaw Angle⋆	Stability◦	Resilience◦	Consistency◦
LKA	$I=[0,15]$ $\varphi_1 \equiv \lvert d_{lat}\rvert \leq 0.75$ $\varphi \equiv \square_I(\varphi_1)$	$I=[0,15]$ $\varphi_2 \equiv \theta_{yaw} \leq 0.26$ $\varphi \equiv \square_I(\varphi_2)$	$I=[0,15]$ $\varphi_3 \equiv \lvert d_{lat}\rvert \leq 0.6$ $\varphi \equiv \square_I(\varphi_3)$	$I=[0,15], I'=[0,1]$ $\varphi_{41} \equiv \neg\varphi_3, \varphi_{42} \equiv \varphi_3$ $\varphi \equiv \square_I(\varphi_{41} \rightarrow \diamond_{I'}\varphi_{42})$	$I=[0,15]$ $\varphi_5 \equiv \lvert \theta^{\Delta}_{steer}\rvert \leq 0.75$ $\varphi \equiv \square_I(\varphi_5)$
Spec.	Position⋆	Orientation⋆	Stability◦	Resilience◦	Consistency◦
APV	$I=[0,25]$ $\varphi_{11} \equiv d_{lidar} > 0$ $\varphi_{12} \equiv \lvert d_{car}\rvert < 0.75$ $\varphi \equiv \square_I(\varphi_{11}) \wedge \diamond_I(\varphi_{12})$	$I=[0,25]$ $\varphi_{22} \equiv \lvert \theta_{head}\rvert < 0.17$ $\varphi \equiv \square_I(\varphi_{11}) \wedge \diamond_I(\varphi_{21})$	$I=[0,25]$ $\varphi_3 \equiv d_{lidar} \geq 0.25$ $\varphi \equiv \square_I(\varphi_3)$	$I=[0,25], I'=[0,1]$ $\varphi_{41} \equiv \neg\varphi_3, \varphi_{42} \equiv \varphi_3$ $\varphi \equiv \square_I(\varphi_{41} \rightarrow \diamond_{I'}\varphi_{42})$	$I=[0,25]$ $\varphi_5 \equiv \lvert \theta^{\Delta}_{steer}\rvert \leq 0.52$ $\varphi \equiv \square_I(\varphi_5)$
Spec.	Ball Position⋆	Plate Angle⋆	Stability◦	Resilience◦	Consistency◦
BBC	$I=[0,20]$ $\varphi_1 \equiv \lvert d_{ball}\rvert < 0.125$ $\varphi \equiv \square_I(\varphi_1)$	$I=[0,20]$ $\varphi_2 \equiv \lvert \theta_{plate}\rvert < 0.35$ $\varphi \equiv \square_I(\varphi_2)$	$I=[0,20]$ $\varphi_3 \equiv \lvert d_{ball}\rvert < 0.11$ $\varphi \equiv \square_I(\varphi_3)$	$I=[0,20], I'=[0,0.3]$ $\varphi_{41} \equiv \neg\varphi_3, \varphi_{42} \equiv \varphi_3$ $\varphi \equiv \square_I(\varphi_{41} \rightarrow \diamond_{I'}\varphi_{42})$	$I=[0,20]$ $\varphi_{51} \equiv \lvert \tau^{\Delta}_1\rvert \leq 0.425$ $\varphi_{52} \equiv \lvert \tau^{\Delta}_2\rvert \leq 0.425$ $\varphi \equiv \square_I(\varphi_{51} \wedge \varphi_{52})$

TABLE 3: Hyperparameters of abstraction

	ACC		AFC		LKA		APV		BBC	
Semantics	Distance	Velocity	Steady	Edge	Deviation	Angle	Position	Orientation	Ball	Plate
d_{MIN}	0.001	0.001	0.001	0.001	0.010	0.001	0.005	0.005	0.001	0.001
d_{MAX}	0.005	0.005	0.100	0.100	0.050	0.005	0.050	0.050	0.020	0.020
n_{MIN}	1.00%	1.00%	0.10%	0.10%	1.00%	1.00%	0.50%	0.50%	0.10%	0.10%

considered stable if it is not close to the edge condition of major property violation.

- ***Resilience***: Resilience inspects the recoverability of a system against fluctuations. Namely, if a system runs into an edge condition, we would like to know if it can bounce back to steady conditions within a fixed time interval.
- ***Consistency***: This specification evaluates the oscillation of the control output; namely, if a controller generates a fluctuating signal, it may bring uncomfortableness to crews, mechanical staining, and energy waste. An unstable and steady control output is preferred from a control unit.

4.2.3 Abstract Model Details

Hyperparameters. Table 3 summarizes the hyperparameters of the abstraction. The semantics values in each dimension are normalized from -1 to 1.

We set the same mean and maximum values for error thresholds $\varepsilon_{\texttt{cur}}$ and $\varepsilon_{\texttt{pred}}$. The average semantics error e_{MEAN} to 0.005, which is 0.25% of the entire value range and the semantics maximum error e_{MAX} is set to 0.2 which is 10% over the value bound. The reduction level threshold is set to 0.5%.

Sub-controller setting. As we mentioned in Section 3.3.1, each subject system contains nine DRL sub-controllers from three state-of-the-art algorithms (DDPG [36], SAC [40], and TD3 [37]) with different reward configurations. The sub-controllers are trained with biased weights w_i for each degree of satisfaction regarding the specifications φ_i which is normalized to $[-1, 1]$. More specifically, for the three sub-controllers under each algorithm, two of them have weights of $[0.7, 0.2, 0.1]$ where 0.7 is assigned for one of the major specifications, 0.2 is for the other major specification, and 0.1 is distributed uniformly to the other minor specifications. For the third type of sub-controller, it is with weights configuration of $[0.4, 0.4, 0.2]$, where two major specifications have the same weight 0.4, and the rest is assigned to minor specifications. In addition, we incorporate the default controller from the collected benchmark, *e.g.*, proportional integral derivative controller and model predictive controller. In total, there are 10 sub-controllers for each AI-CPS.

Semantics.

For the degree of satisfaction, we manually and carefully design the formula that can reflect the extent of the satisfaction/violation for the specification of the system. The formulae are in the form of Simulink blocks, which is hard to display in the paper. Thus, we put the detailed formulas

TABLE 4: RQ1 – Comparison of state, action and transition space between concrete and abstract models. (*Total Con.*: total concrete samples. *Uni. Con*: Unique concrete samples. Abstract: number of abstract state. Reduction: level of abstraction.)

System	State				Action			Transition		
	Total Con.	Uni. Con.	Abstract	Reduction	Uni. Con.	Abstract	Reduction	Uni. Con.	Abstract	Reduction
ACC	7,442,000	6,014,268	891	0.0148%	5,287,459	11	0.0002%	6,106,552	6,526	0.1068%
AFC	2,820,000	2,820,000	225	0.0079%	1,449,622	13	0.0009%	2,714,989	2,070	0.0762%
LKA	830,500	226,211	403	0.1781%	226,904	11	0.0048%	227,715	1,814	0.7966%
APV	368,817	368,817	843	0.2285%	333,370	12	0.0036%	351,586	6,496	1.8476%
BBC	1,655,500	1,655,500	362	0.0218%	1,581,955	104	0.0066%	1,520,471	13,336	0.8770%

TABLE 5: RQ1 – Abstraction error of semantics in states

Error	ACC		AFC		LKA		APV		BBC	
	Distance	Velocity	Steady	Edge	Deviation	Angle	Position	Orientation	Ball	Plate
State Mean	3.77E-03	3.85E-03	1.42E-03	1.54E-03	6.83E-04	9.06E-04	2.10E-03	1.58E-03	3.78E-03	5.02E-04
State Max	1.11E-01	1.21E-01	3.69E-02	7.58E-02	1.29E-01	1.08E-01	6.84E-02	2.48E-02	1.98E-02	1.42E-02

and computation of the degree of satisfaction for each specification on our supplement website. In this work, we choose two major specifications from each system, shown in Table 2, to extract the semantics.

4.3 Experiment Setup

RQ1. To answer this RQ, we first simulate 1,000 times with each sub-controller to collect the experiences from 10,000 episodes. The collected experience is then split into a modelling set and a validation set with a ratio of $8:2$. The former is used to build the abstract model, and the latter is to validate the semantics errors between the abstract and concrete models. We use mean and max state semantics errors, defined in Section 3.2.1, to measure the preciseness.

RQ2. We perform the experiment on the five CPS subject systems. Similar to RQ1, we use 80% of the traces to construct the abstract model and use the remaining 20% of the traces to check the prediction error. Moreover, we compute the mean error and maximum error of the predicted semantics (defined in Section 3.2.3). Here, we take more concerns on the *maximum predicted semantics error* since it heavily affects the ensemble methods we proposed. If the predicted semantics significantly deviates from the true value, the ensemble methods may mistakenly drive the system toward crashes.

RQ3. In terms of the evaluation, we simulate each type of controller with 200 random generated episodes and compare the performance between individual and semantics-based controllers.

The specifications illustrated in Table 2 are all extracted from the original documentation of each system. These specifications represent the important operation requirements, such that any violations can possibly result in serious accidents with human life and property losses. As these two metrics are all specification-oriented, which intrinsically reveals the behavior of the system w.r.t safety.

For each system, to produce a comprehensive evaluation, we use multiple specifications as references (shown in Table 2) which contain major specifications like safety, efficiency, and minor specifications related to comfortableness, resilience, etc. Considering the safety-critical features of CPSs, we put more concerns on major specifications and take other minor metrics as supplements to enrich the diversity of the evaluation.

To deliver an in-depth evaluation, we propose two metrics for each specification:

- **Satisfaction Rate (SATE)**: SATE computes the percentage of satisfying simulations in which a specification has never been violated. SATE = $\frac{n_s}{n_T}$, where n_s is the number of satisfying trails without any target specification violations, and n_T is the number of total trails.
- **Mean Absolute Error (MAE)**: As a complement to the SATE metric, MAE reports the mean absolute error. Different controllers may have close SATE but different MAE values; we consider the one with a smaller MAE as the better. MAE = $\frac{1}{n_T}\sum_{i=1}^{n_T} \text{MEAN}|y-\hat{y}|$, where y is the system actual output, $\hat{y}$ is the reference output, and $\text{MEAN}|y-\hat{y}|$ measures the average value of the deviation from the references in one trail.

SATE provides an overview satisfaction ratio regarding the specifications, and MAE reveals a concrete average value of the deviation from the references. Therefore, a safe and robust controller should have a high SATE and a low MAE.

RQ4. Similar to RQ3, we use SATE and MAE evaluation metrics to compare the performance between the newly proposed ensemble strategies and the traditional ones. We also pick the single controller with the best performance in RQ3 in the comparison.

The *Top-1 Semantics Prediction* method takes action with the highest predicted semantics values. We set $k = 3$ for the second *Top-k Semantics Prediction* method.

In terms of the "high-level" agents in the DRL coordinator strategy, we set these agents to put more attention on the specification violations. For the coordinator, we choose stochastic agent A2C, aiming to provide indeterministic action selection. The reward functions of coordinator agents have balanced weights of $[0.4, 0.4, 0.1]$, where 2 major specifications have the same weights 0.4, and the minor specifications share the weight 0.1. Note that different from the balanced sub-controller, for the coordinator agent, any violation of specifications will cause an enormous penalty (-1) to the final reward value. Such that any unsatisfaction will not be overshadowed by other sub-rewards.

Software Dependencies. The ACC model requires the *Model Predictive Control* and *Control System* `MATLAB` tool-

TABLE 6: RQ2 – Abstraction error of semantics in predictions

	ACC		AFC		LKA		APV		BBC	
Error	Distance	Velocity	Steady	Edge	Deviation	Angle	Position	Orientation	Ball	Plate
Prediction Mean	3.61E-03	3.53E-03	1.41E-03	1.46E-03	3.58E-04	6.91E-04	2.07E-03	1.44E-03	3.74E-03	4.89E-04
Prediction Max	1.06E-01	1.46E-01	2.18E-02	4.41E-02	1.29E-01	1.19E-01	7.48E-02	2.48E-02	1.96E-02	1.62E-02

boxes; the APV model requires the *Automated Driving* and *Robotics System* `MATLAB` toolbox; the BBC model requires the *Simscape Multibody* `MATLAB` toolbox. The training for DRL agents in all models requires the *Reinforcement Learning* `MATLAB` toolbox.

Hardware Platform. For DRL training and large-scale evaluation, we use four Lambda Tensorbooks, each of which has an Intel(R) Core(TM) i7-10870H CPU @ 2.20GHz Processor with 8 CPUs and 64G RAM, and an NVIDIA RTX 3080 Max-Q GPU with 16 GB VRAM. The overall computation time of our evaluation takes more than 550 hours.

4.4 Evaluation and Results

4.4.1 RQ1: How is the semantics abstraction in terms of succinctness and preciseness?

We want to examine the effectiveness of the semantics abstraction, *i.e.*, whether the abstract models can shrink the concrete state space while preserving the corresponding semantics characteristics. Table 4 and Table 5 summarize the abstraction details and the semantics errors of the abstraction results among five systems.

The reduction ratio, in Table 4, shows that abstract procedure can effectively reduce the state, action and transition space 99.91%, 99.99%, 99.26%, respectively, on average for all systems. Especially in ACC, AFC and BBC, the size of abstract models only contains hundreds of unique abstract states in contrast to the millions of concrete states in original models.

In terms of the abstraction errors, from Table 5, the mean error of state semantics is on average 0.0020. The maximum semantics error for all systems is on average 0.0708. The results indicate that our abstraction method can succinctly and precisely describe the semantics characteristics in five systems. The abstract model provides a simplified and intrinsic way to understand the system status.

Answer to RQ1: The semantics abstraction can effectively reduce the complexity of the systems and precisely capture the semantics characteristics.

4.4.2 RQ2: Can we rely on abstract models to predict the semantics of the incoming state?

We aim to investigate the effectiveness of our abstraction method in terms of semantics prediction. If the abstract model can precisely estimate the semantics of a state-action pair, then the predicted semantics is expected to guide the ensemble strategy of sub-controllers. Namely, the potential failures can be noticed in advance, and the proper action can be selected to keep the system operating on the right track.

Table 6 shows the error measurements of the semantics between the predicted concrete states and the predicted abstract states. We apply the same thresholds to examine the efficacy of prediction regarding average and maximum semantics errors, respectively. We notice that, regarding the semantics prediction, the abstract model has a mean error of an average 0.0018; and the maximum error of predicted semantics is on average 0.0701. None of the errors in any systems exceed the mean and maximum thresholds (0.005 and 0.2) from the abstraction refinement. These results show that the refined abstract model is capable of accurately predicting the incoming semantics based on the current state-action pair.

Answer to RQ2: The semantics of the incoming states can be precisely predicted based on the abstract model. The semantics predictability of the abstract model is reliable to provide meaningful guidance for new ensemble strategies.

4.4.3 RQ3: Can the classical ensemble-based controllers outperform the standalone controllers?

Table 7 shows the detailed evaluation results for controllers with classical ensemble methods and a single controller in each system. Figure 6 gives an intrinsic illustration for better comparison among various controllers.

- ACC. The major metrics in ACC are distance and velocity, which indicate the safety and efficiency properties, respectively. From Table 7, all ensemble controllers with classical methods have a good performance on the distance metric. Particularly, the distance SATE of Average methods is above 97%, the MAE is less than 0.01, and the Majority-Voting method never violates the distance specification in all testing episodes with 100% SATE.
  From the velocity aspect, ensemble methods also give competitive results. Noticeably, these controllers demonstrate great capabilities in balancing the safety-related distance metric and efficiency-related velocity metric. Namely, the two classical ensemble methods achieve over 74% SATE on velocity while maintaining outstanding results on distance. Most standalone controllers (except for DDPG-3 and TD3-3) can only focus on one metric but overlook the other.
- AFC. Neither individual controllers nor classical ensemble controllers in AFC can perfectly maintain the reference air-fuel-ratio under both steady and edge conditions. From Table 7, the ensemble controllers can tackle steady and edge metrics with about 95% SATEs and show over 80% SATES on stability. Nevertheless, controllers like SAC-2 and TD3-1 also present similar performance. Therefore, we do not observe the obvious advantages of classical ensemble methods in AFC.
- LKA. The Majority-Voting and the Average methods show an intermediate performance between DDPG controllers and TD3 controllers. Namely, the DDPG controllers have 100% SATE on yaw angle but about 84% SATE on lateral

TABLE 7: RQ3 – Performance evaluation of individual controllers and classic ensemble methods on 5 systems. The ensemble methods are marked in gray, and the best result is highlighted in cyan.

ACC	Distance⋆		Velocity⋆		Stability◦		Resilience◦		Consistency◦	
	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE
MPC	59.80%	0.0725	81.20%	0.8681	53.40%	0.2380	2.20%	0.3812	57.20%	0.0980
DDPG-1	59.40%	0.4109	96.40%	0.6716	56.04%	0.2095	0.80%	0.6890	63.20%	0.1003
DDPG-2	57.80%	0.0934	84.60%	0.8162	54.80%	0.2185	4.00%	0.3862	58.40%	0.1121
DDPG-3	100.00%	0.0000	71.00%	1.3808	100.00%	0.0000	-	-	100.00%	0.0147
SAC-1	57.60%	0.1342	90.80%	0.7621	55.40%	0.2196	1.20%	0.5443	56.20%	0.2033
SAC-2	66.40%	0.0592	79.20%	1.0127	59.40%	0.1760	9.00%	0.1259	61.80%	0.2468
SAC-3	65.00%	0.1206	76.20%	1.1686	62.40%	0.1437	14.00%	0.1378	62.00%	0.1964
TD3-1	56.80%	1.6749	95.20%	0.9534	54.80%	0.2125	0.00%	0.1733	96.40%	0.0271
TD3-2	69.00%	0.1693	81.00%	1.0724	59.40%	0.1731	4.00%	0.3891	74.00%	0.0315
TD3-3	98.40%	0.0276	75.20%	1.3358	73.40%	0.0670	0.00%	0.7793	100.00%	.0173
Vote	100.00%	0.0000	74.20%	1.2107	80.40%	0.0080	0.50%	0.2795	16.00%	0.4133
Average	97.80%	0.0083	75.20%	1.1897	73.40%	0.0347	0.80%	0.2501	80.80%	0.0215

AFC	Steady⋆		Edge⋆		Stability◦		Resilience◦		Consistency◦	
	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE
PID	92.80%	0.0011	98.60%	0.0033	92.00%	0.0019	77.50%	0.1261	100.00%	0.0025
DDPG-1	80.00%	0.0065	80.00%	0.0048	25.00%	0.0143	100.00%	0.0404	100.00%	0.0042
DDPG-2	82.00%	0.0083	100.00%	0.0055	72.00%	0.0066	26.00%	0.0556	100.00%	0.0039
DDPG-3	81.50%	0.0129	99.50%	0.0064	4.50%	0.2070	100.00%	0.0000	100.00%	0.0038
SAC-1	94.00%	0.0047	99.00%	0.0038	37.00%	0.0020	100.00%	0.0000	100.00%	0.0061
SAC-2	97.50%	0.0026	97.50%	0.0035	86.50%	0.0009	100.00%	0.0000	100.00%	0.0047
SAC-3	51.50%	0.0119	98.00%	0.0062	26.00%	0.0373	57.50%	0.1241	100.00%	0.0044
TD3-1	100.00%	0.0031	95.50%	0.0034	74.00%	0.0018	100.00%	0.0000	100.00%	0.0044
TD3-2	94.00%	0.0040	94.00%	0.0038	28.50%	0.0016	45.50%	0.0136	100.00%	0.0069
TD3-3	99.00%	0.0029	87.50%	0.0040	89.50%	0.0004	100.00%	0.0000	100.00%	0.0045
Vote	94.50%	0.0029	96.00%	0.0035	81.00%	0.0011	100.00%	0.0000	100.00%	0.0059
Average	95.00%	0.0044	100.00%	0.0037	86.00%	0.0005	100.00%	0.0000	100.00%	0.0081

LKA	Lateral Deviation⋆		Yaw Angle⋆		Stability◦		Resilience◦		Consistency◦	
	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE
MPC	82.50%	0.4634	89.50%	0.1116	63.00%	0.2474	17.05%	0.4872	55.00%	0.5423
DDPG-1	80.00%	0.1064	100.00%	0.0129	69.50%	0.0131	36.07%	0.2242	12.50%	0.2275
DDPG-2	85.50%	0.0543	100.00%	0.0133	76.50%	0.0078	53.19%	0.1185	39.00%	0.2028
DDPG-3	91.50%	0.0617	100.00%	0.0147	81.50%	0.0047	75.68%	0.0405	0.50%	0.5710
SAC-1	91.00%	0.0558	93.00%	0.0198	81.00%	0.0034	94.74%	0.0088	83.50%	0.1698
SAC-2	87.50%	1.8354	81.00%	0.0921	80.00%	0.0747	60.00%	0.3074	5.00%	0.3262
SAC-3	91.00%	0.0522	86.00%	0.0240	81.00%	0.0087	97.37%	0.0100	52.00%	0.2465
TD3-1	93.00%	0.0419	96.50%	0.0148	86.50%	0.0028	96.30%	0.0062	4.00%	0.4270
TD3-2	93.00%	0.0561	96.50%	0.0183	85.00%	0.0030	92.86%	0.0119	0.50%	0.2864
TD3-3	93.00%	0.0386	96.00%	0.0118	86.50%	0.0028	96.30%	0.0062	26.50%	0.2852
Vote	91.00%	0.0501	98.50%	0.0143	80.00%	0.0044	92.50%	0.0125	7.00%	0.3410
Average	88.50%	0.0436	100.00%	0.0119	79.00%	0.0054	78.57%	0.0414	89.00%	0.0945

APV	Position⋆	Orientation⋆	Lidar⋆	Park Success⋆	Stability◦		Resilience◦		Consistency◦	
	MAE	MAE	MAE	SATE	SATE	MAE	SATE	MAE	SATE	MAE
RL-origin	1.2385	0.2539	1.7669	87.50%	92.50%	0.0135	1.00%	0.8667	47.00%	0.2105
DDPG-1	0.9855	0.2546	1.7986	90.50%	96.00%	0.0021	1.00%	0.7500	27.00%	0.2498
DDPG-2	1.3251	0.2559	1.6863	86.50%	89.00%	0.0088	0.50%	0.9545	20.50%	0.2619
DDPG-3	1.3335	0.2956	1.7307	88.50%	91.00%	0.0058	1.50%	0.8333	13.50%	0.2443
SAC-1	1.1124	0.2088	1.8519	90.50%	97.00%	0.0031	0.00%	1.0000	4.00%	0.3852
SAC-2	1.2364	0.1440	1.8116	88.00%	93.00%	0.0081	0.00%	1.0000	5.50%	0.4147
SAC-3	1.1510	0.1889	1.7784	89.50%	92.00%	0.0045	2.00%	0.7500	2.50%	0.4042
TD3-1	1.2063	0.2685	1.8321	91.00%	94.00%	0.0034	1.00%	0.8333	8.00%	0.2170
TD3-2	1.2768	0.2217	1.8593	85.00%	89.50%	0.0149	1.00%	0.9048	46.00%	0.2700
TD3-3	1.3043	0.2451	1.8765	89.50%	92.00%	0.0105	2.00%	0.7500	70.00%	0.3000
Vote	1.2396	0.2121	1.8747	90.00%	94.50%	0.0028	0.00%	1.0000	3.50%	0.4229
Average	1.3702	0.2374	1.9293	89.00%	98.50%	0.0014	0.00%	1.0000	82.50%	0.1270

BBC	Ball Position⋆		Plate Angle⋆		Stability◦		Resilience◦		Consistency◦	
	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE
RL-origin	92.00%	0.0844	89.00%	0.0809	86.50%	0.0447	40.74%	0.3324	26.33%	0.0506
DDPG-1	92.50%	0.2766	88.50%	0.0618	82.50%	0.0445	54.29%	0.2964	47.67%	0.0468
DDPG-2	86.50%	0.4917	82.00%	0.1905	83.00%	0.0830	38.24%	0.4756	63.33%	0.0486
DDPG-3	90.50%	0.5421	81.00%	0.1449	78.00%	0.0754	50.00%	0.3534	62.33%	0.0349
SAC-1	91.50%	0.3192	89.00%	0.0849	85.00%	0.0441	33.33%	0.4079	41.00%	0.0560
SAC-2	87.00%	0.6206	88.00%	0.1340	76.00%	0.0955	45.83%	0.4019	74.33%	0.0454
SAC-3	89.00%	0.0686	89.00%	0.1872	84.50%	0.0713	25.81%	0.3504	64.67%	0.0509
TD3-1	93.00%	0.1219	92.00%	0.0542	84.50%	0.0414	51.61%	0.2784	77.50%	0.0039
TD3-2	93.00%	0.3685	90.50%	0.1166	85.50%	0.0564	44.83%	0.3735	19.67%	0.1562
TD3-3	92.50%	0.3323	83.00%	0.0987	82.00%	0.0555	52.78%	.3027	57.00%	0.0168
Vote	91.00%	0.3624	91.00%	0.0981	82.00%	0.0599	47.22%	0.3371	48.33%	0.0587
Average	89.50%	0.4329	82.50%	0.1103	80.50%	0.0656	46.15%	0.3896	95.00%	0.0062

deviation, while TD3 controllers have 97.67% SATE and 92.33% SATE of the two major specifications respectively. On the other hand, the Majority-Voting method has 91% and 98.5% SATEs, and the Average method obtains 88.5% and 100% SATEs. In addition, we find that the Average method has better consistency results than the Majority-Voting one.

- APV. As mentioned in 4.2.1, successful parking is counted if the vehicle precisely stopped at the target spot without any collision during the maneuver. In Table 7, the park success metric reports the overall successful parking rate over the 200 simulations. The DDPG-1 agent and SAC-2 agent obtain the lowest position MAE and orientation MAE respectively, but the highest parking success rate is achieved by TD3-1 agent. Although the Average method has the best stability and consistency SATE result, the traditional ensemble methods do not show major advantages on major specifications.
- BBC. TD3-1 agent outperforms other controllers on both the position and the angle specifications, where the SATE reaches 93% and 92%, respectively. In terms of the traditional ensemble methods, the Majority-Voting achieves 91% SATE on both position and angle specifications, while the average approach has 89.5% and 82.5% SATEs, respectively. The Majority-Voting method has better performance on major specifications than the Average method, but the Average one has the best consistency result than any others. In general, the traditional ensemble controllers do not perform obviously better than the individual ones.

To summarize, in many cases, the performance of single controllers is better than traditional ensemble strategies. The traditional ensemble methods cannot adaptively select and combine the optimal controllers under different conditions. Nevertheless, the Average method has exceptional results in terms of the consistency metrics, we think the averaged outputs compensate the oscillation phenomenon when switching between different controllers.

Answer to RQ3: The traditional ensemble methods do not obviously outperform the individual controllers. New ensemble strategies are needed to dynamically combine the strengths of the constituent controllers.

4.4.4 RQ4: Are the newly proposed ensemble strategies better than traditional ensemble methods?

In this RQ, we would like to investigate the performance of our newly proposed ensemble. Table 8 and Figure 6 show the evaluation results.

- ACC. The semantics-based methods, Top-1 and Top-k, have distance SATE over 92% and velocity SATE over 80%, which show a better compensating ability between two major metrics. The resilience and consistency of these two controllers have similar performance compared with individual controllers but are better than the classical ensemble methods.
  The Coordinator with prediction method achieves 100% and 85.1% SATE on distance and velocity metrics, respectively, which is the highest comprehensive scores among all types of controllers. The Coordinator method without prediction also obtains 100% STAE on the distance specification but a relatively lower score, 71.2%, on the velocity metric. Furthermore, both methods have 100% SATE on the stability metric, which outperforms any other ensemble controllers.
- AFC. The semantics-based ensemble controllers in AFC have slightly higher SATE values on steady and edge metrics but much smaller MAE than traditional ensemble controllers. It indicates that, for major specifications, the semantics-based methods outperform the classic ones with smaller deviations from reference values. Meanwhile, all ensemble controllers achieve 100% STAE on both resilience and consistency metrics.
  We also observe that the Coordinator(prediction) method obtains 100% SATE over steady, edge, resilience and consistency specifications, which obviously outperforms any other controllers. Namely, this Coordinator method generates a consistent and resilient synergistic control output that never violates the safety requirements during entire testing runs. For the Coordinator(no-prediction) method, it has close performance like the Top-1 semantics method but provides better stability; therefore, we consider it the second-best controller in AFC.
- LKA. We find that the traditional methods behave better than the semantics-based methods for the lateral deviation and yaw angle specifications in LKA. Although the two semantics-based methods obtain 100% SATE on the yaw angle metric, the SATE and MAE of deviation metric are slightly worse than the obtainable from classic methods. The Coordinator(prediction) method shows the best overall results on major metrics, namely, 92% and 100% SATEs on deviation and angle, accordingly.
- APV. We do not find a notable difference between semantics-based and traditional ensemble controllers regarding the major specifications. However, the Majority-Voting and Top-1 semantics methods have a higher success parking rate than the averaging methods (Average, Top-k semantics).
  The top 2 highest successful parking rates are achieved by the two coordinator methods, and the MAEs of position and orientation metrics are fairly low as well. The performance of these two controllers in terms of consistency is not good.
  For the parking environment in APV, as the original model released by MathWorks [69], the controller only outputs the steering angle to adjust the moving direction while the vehicle is running at a constant speed at 2 m/s. And the resilience specification in APV measures the distance from the target vehicle to other parked vehicles. Therefore, when the target vehicle is moving too close to other vehicles, the controller cannot make the vehicle move backwards but change the steering direction to avoid any collision. It is not really efficient, as the vehicle takes a relatively long time to get back to a safe distance from other cars by only controlling the steering angles while the vehicle keeps moving at a constant speed. This situation happens, especially when the initial position of the target vehicle is relatively close to other parked cars. We consider this to be one of our future works; namely, we propose to enable the speed control and the steering angle control at the same time to build a more sophisticated parking system.

TABLE 8: RQ4 – Performance evaluation of new-proposed ensemble methods on 5 systems.

ACC	Distance⋆		Velocity⋆		Stability◦		Resilience◦		Consistency◦	
	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE
DDPG-3	100.00%	0.0000	71.00%	1.3808	100.00%	0.0000	-	-	100.00%	0.0147
Vote	100.00%	0.0000	74.20%	1.2107	80.40%	0.0080	0.50%	0.2795	16.00%	0.4133
Top-1 Pred	92.00%	0.0335	81.00%	0.9843	51.80%	0.2299	5.40%	0.3335	52.50%	0.2294
Top-k Pred	94.20%	0.0472	80.40%	0.9785	53.60%	0.2201	5.00%	0.2356	43.00%	0.3410
Coordinator	100.00%	0.0000	71.20%	1.3799	100.00%	0.0000	-	-	52.40%	0.2743
Coordinator(pred)	100.00%	0.0000	85.10%	0.8025	100.00%	0.0000	-	-	57.00%	0.2164
AFC	Steady⋆		Edge⋆		Stability◦		Resilience◦		Consistency◦	
	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE
TD3-1	100.00%	0.0031	95.50%	0.0034	74.00%	0.0018	100.00%	0.0000	100.00%	0.0044
Average	95.00%	0.0044	100.00%	0.0037	86.00%	0.0005	100.00%	0.0000	100.00%	0.0081
Top-1 Pred	97.50%	0.0075	99.50%	0.0028	12.50%	0.0362	100.00%	0.0000	100.00%	0.0039
Top-k Pred	99.50%	0.0023	94.00%	0.0033	75.50%	0.0011	100.00%	0.0000	100.00%	0.0039
Coordinator	97.00%	0.0043	97.00%	0.0038	81.50%	0.0012	100.00%	0.0000	100.00%	0.0039
Coordinator(pred)	100.00%	0.0096	100.00%	0.0054	86.50%	0.0009	100.00%	0.0000	100.00%	0.0043
LKA	Lateral Deviation⋆		Yaw Angle⋆		Stability◦		Resilience◦		Consistency◦	
	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE
TD3-3	93.00%	0.0386	96.00%	0.0118	86.50%	0.0028	96.30%	0.0062	26.50%	0.2852
Average	88.50%	0.0436	100.00%	0.0119	79.00%	0.0054	78.57%	0.0414	89.00%	0.0945
Top-1 Pred	83.00%	0.0824	100.00%	0.0243	70.50%	0.0144	42.37%	0.2052	0.50%	0.3000
Top-k Pred	85.00%	0.0537	100.00%	0.0120	73.50%	0.0091	50.94%	0.1447	36.00%	0.1997
Coordinator	87.00%	0.0742	99.50%	0.0173	79.00%	0.0060	73.81%	0.0666	0.10%	0.3451
Coordinator(pred)	92.00%	0.0171	100.00%	0.0117	85.50%	0.0041	89.66%	0.0153	27.00%	0.2716
APV	Position⋆	Orientation⋆	Lidar⋆	Park Success⋆	Stability◦		Resilience◦		Consistency◦	
	MAE	MAE	MAE	SATE	SATE	MAE	SATE	MAE	SATE	MAE
TD3-1	1.2063	0.2685	1.8321	91.00%	94.00%	0.0034	1.00%	0.8333	8.00%	0.2170
Vote	1.2396	0.2121	1.8747	90.00%	94.50%	0.0028	0.00%	1.0000	3.50%	0.4229
Top-1 Pred	1.2835	0.2217	1.9195	90.00%	97.50%	0.0013	0.00%	1.0000	85.50%	0.1265
Top-k Pred	1.2655	0.2127	1.8808	88.50%	95.50%	0.0047	0.00%	0.9556	78.00%	0.1325
Coordinator	1.1676	0.1716	1.8922	91.50%	95.50%	0.0021	0.00%	1.0000	2.50%	0.3545
Coordinator(pred)	1.0631	0.1487	1.7616	92.50%	96.00%	0.0068	0.00%	1.0000	3.50%	0.4128
BBC	Ball Position⋆		Plate Angle⋆		Stability◦		Resilience◦		Consistency◦	
	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE	SATE	MAE
TD3-3	92.50%	0.3323	83.00%	0.0987	82.00%	0.0555	52.78%	0.3027	57.00%	0.0168
Vote	91.00%	0.3624	91.00%	0.0981	82.00%	0.0599	47.22%	0.3371	48.33%	0.0587
Top-1 Pred	93.50%	0.0664	91.00%	0.0379	85.00%	0.0313	46.67%	0.2697	19.33%	0.1695
Top-k Pred	90.00%	0.2500	90.50%	0.0669	81.00%	0.0575	39.47%	0.3717	53.00%	0.0459
Coordinator	92.00%	0.1739	91.50%	0.1139	83.00%	0.0511	47.06%	0.2632	44.00%	0.0493
Coordinator(pred)	93.00%	0.0805	95.00%	0.0437	85.50%	0.0275	44.83%	0.2988	53.00%	0.0452

- BBC. The Top-1 semantics method has the highest SATE and lowest MAE on ball position and stability metrics and equivalently good results on plate angle and resilience metrics. Particularly, the position MAE of the Top-1 semantics method is 0.0664, and the angle MAE is 0.0279, which are about 80% and 70% less than the MAE of traditional methods, respectively. In terms of consistency, the Average method has the highest MAE 95%, and the Top-1 semantics method gets the lowest 19.33%.
  Not surprisingly, the Coordinator(prediction) methods give the best performance on major metrics with 93% and 95% SATEs on the position and the angle specifications. Moreover, the stability of the Coordinator(prediction) methods is slightly higher than other ensemble methods.

To conclude, the semantics-based methods have better or similar performance compared with traditional methods on four systems. Namely, the semantics predictions can reinforce the controller selection logic and drive the system toward safe and efficient operations. Note that a well-refined abstract model is required to provide precise predictions of incoming semantics; otherwise, the semantics-based controllers can be misguided to unexpected states. For Coordinator methods, the evaluation results indicate the Coordinator(prediction) approach can outperform other controllers obviously. The Coordinator methods are capable of controlling more dynamically and strategically which brings better control flexibility than deterministic methods.

Answer to RQ4: The semantics-based ensemble strategy can deliver similar or better performance than both traditional ensemble methods and individual control methods in all systems except LKA. The Coordinator ensemble strategies are helpful in enhancing the overall capability and reliability of the control system regarding multiple operation objectives. Among the five experimental CPSs, the Coordinator with prediction method outperforms others.

5 Discussion

State space reduction. High-dimensional continuous state space is one of the major challenges in reinforcement learning, as it makes the search for optimal policy computationally intractable. In addition, unlike the commonly used reinforcement learning benchmarks with discrete state space (e.g. CartPole [36], Mountain Car [72]), CPSs usually come with complicated dynamics and multiple operation requirements which require more efforts to construct reliable control systems.

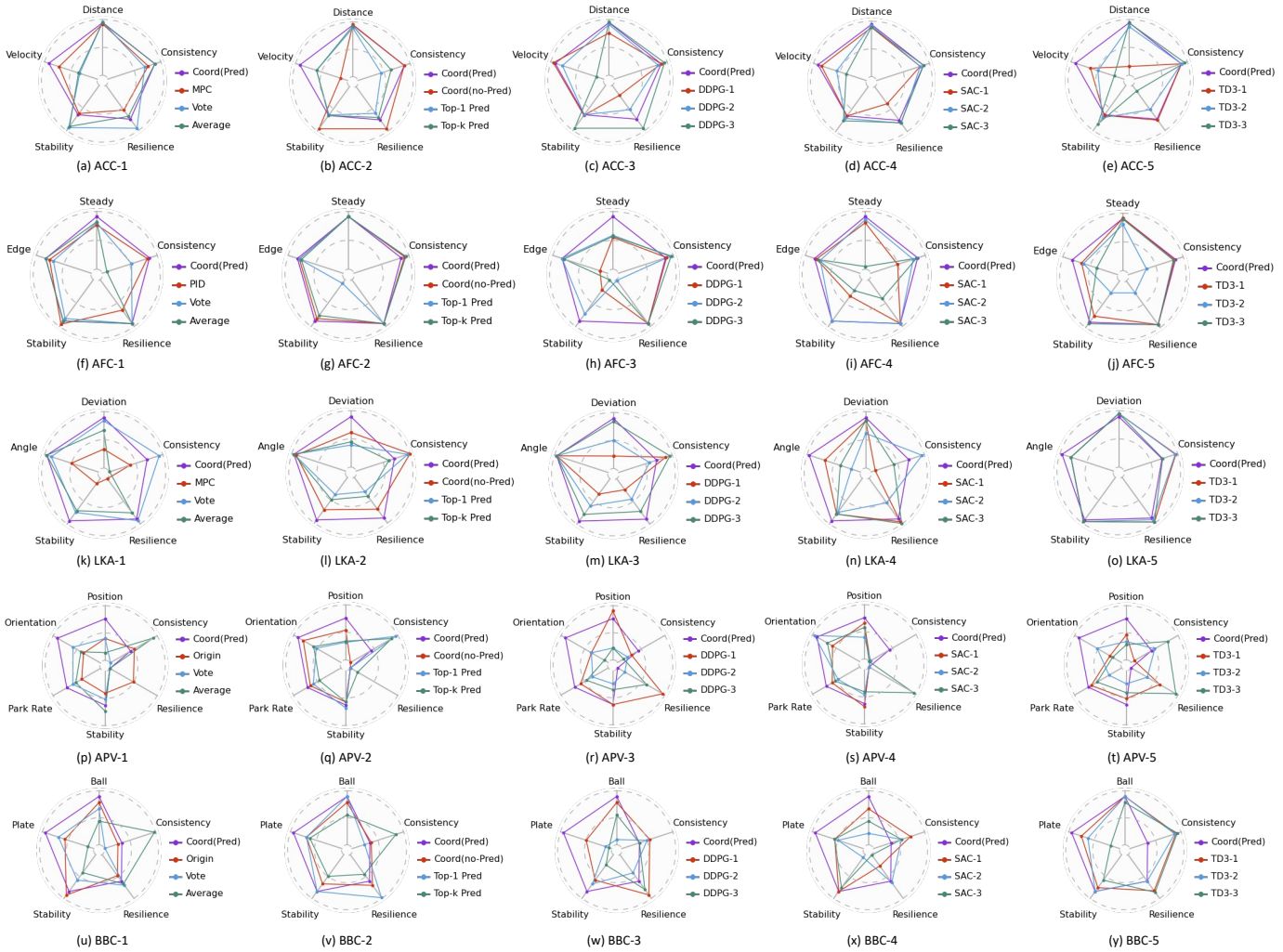


Fig. 6: RQ4 – Performance comparison on standalone and ensemble controllers. For each row, radar 1 is for classic ensemble method with built-in. radar 2 is newly proposed methods. radar 3,4,5 are for Coordinator(pred) with individual.

Our abstraction can decompose the concrete state space and transfer to a more compact abstract state space. Each abstract state should represent a group of concrete states that share similar physical information regarding the degree of satisfaction with different specifications. Therefore, by observing the abstract states enclosed by each simulation trail, we can intuitively understand the satisfaction/violation w.r.t to specifications. A brief overview of the behavior characteristics of the controller can be derived so that we can interpret the strengths and weaknesses of the current controller regarding different system requirements.

Classic ensemble methods such as majority-vote cannot be directly applied to the CPS scenario due to the continuous action space. Certain abstraction techniques are required to narrow the action space to limited options; therefore, further various ensemble strategies become adaptable. Moreover, it is also challenging to conduct semantics prediction on concrete state space. Since when encountering unseen concrete states or actions during simulations, concrete state-based semantics predictions are unavailable as no reference exists in collected experience. Our semantics abstraction maps the concrete states from different regions onto specific abstract states accordingly. Namely, we are able to transfer unseen concrete states to proper abstract states and predict the incoming semantics, which can guide our newly proposed ensemble strategies. Overall, semantics abstraction is necessary as it not only reduces the large state space with better transparency but initiates the predictability of the semantics of the incoming states, which is vital to our ensemble strategies.

Based on the results from five experiment systems, our abstraction method can significantly reduce the size of the state, action and transition space over 99%, and restrict the semantics errors within an acceptable range. We notice that the reduction level of a system is not determined by the dimensions of the state space but by the diversity of its behaviour in terms of the specifications. For instance, BBC has a much larger state space than LKA, but its abstract state space is conversely smaller. This finding fits the practical situation that many subsets in the state space can have diverse and distinct values but represent a similar level of satisfaction regarding the system specifications. Therefore, abstracting the system based on concrete state values can be expensive and intricate; and the corresponding abstracted

model may not effectively scale down the system and accurately conclude the system dynamics. Thus, our semantics-based abstraction tackles this problem from the semantics aspect in order to effectively and precisely decompose complex CPSs.

Guidance from semantics predictions. We investigate whether the semantics can provide guidance to empower the controller fusion strategies within the ensemble framework. To do that, we utilize the abstract model to predict the incoming semantics and monitor the possible outcomes of choosing different transitions. In this way, the ensemble method can avoid dangers in advance and drive the system toward optimal performance. Table 5 demonstrates the ability of the semantics prediction that the abstract models have low prediction errors on the validation dataset.

Enhanced ensemble strategy. In Section 4.4.3 and 4.4.4, we propose two comparison sets between: 1) individual controllers and traditional ensemble methods, and 2) newly proposed ensemble methods and the classical ones. Also, we apply multiple major and minor specifications to obtain a detailed and comprehensive evaluation for each controller. From the enriched evaluation results, we discover that the ensemble controllers can deliver better performance than individual controllers, the semantics prediction can reinforce the ensemble strategies, and finally, the Coordinator methods can outperform any others in each system in terms of major specifications. We consider that the semantics predictions from the abstract model may have a large error when encountering certain corner state-action pairs. As illustrated in Table 6, the maximum error of the predicted semantics is much higher than the average error. The rule-based semantics-guided ensemble strategies, namely, Top-1 and Top-k, may produce a sub-optimal control signal. To mitigate such drawbacks, we can either set tighter thresholds on abstract models to maximally reduce the number of these corner samples or craft advanced semantics-guided ensemble strategies with extra attention on special system conditions. However, we notice that Coordinator (hierarchical) methods can aware of such special system behavior during training. We consider the coordinator method can compensate some weak spots from the abstract model and produce the ensemble output more adaptively.

Besides the Majority-Voting and Averaging methods, some advanced decision fusion strategies have been proposed to enhance the overall ensemble performance, such as Bayes Optimal Classifier [52], Stacked Generalization [73], Super Learner [74], Consensus [75], and Query-By-Committee [76]. We do not apply these methods in our study since 1) unfeasibility in a large dataset and complex environments, and 2) unavailability for tasks in CPSs. Particularly, Bayes Optimal Classifier, Stacked Generalization, and Super Learner are not applicable to systems with multiple sub-learners and large datasets. Methods like Consensus, and Query-By-Committee, are specialized for unsupervised learning and active learning, respectively, that do not fit the context of CPSs.

In this paper, we take the first step towards a semantics-guided safety enhancement framework for AI-CPS. The empirical evaluations show that adopting semantics guidance and DRL hierarchical control into ensemble methods is a promising solution to improve the safety, efficiency, robustness and reliability of AI-CPSs. The semantics abstraction can help developers 1) understand the system status, 2) simplify the system for better manipulability, and 3) support the ensemble strategies regarding the specifications. Moreover, a DRL-coordinator can dynamically combine multiple controllers to drive the system toward an optimal status under various circumstances.

6 Threats To Validity

In this section, we discuss the potential threats to our study and the actions that we have taken to mitigate them.

Internal threats. The abstraction level could be an internal factor that impacts the analysis results. In our study, we design our semantics and abstraction that can reflect the satisfaction of the system at different granularities, w.r.t. requirements. Ideally, a suitable semantics-oriented abstraction can succinctly and precisely reduce the size of state space while preserving the semantics properties.

In particular, we propose a set of parameters to balance the trade-off between the level of abstraction and abstraction errors. The final abstract model has been iteratively refined to succinctly and effectively narrow the state space and maintain sufficient accuracy in semantics description and prediction.

Conclusion threats. The validity of our results can be affected by randomness factors. The randomized external inputs and initial conditions in simulations generate different testing episodes. To mitigate such threats, and further counteract the effects brought by randomness, we repeat the evaluation experiments several runs and report the Satisfaction Rate and the Mean Absolute Error to obtain fair and comprehensive results.

Construct threats. It is possible that the evaluation metrics we proposed might not fully characterize all the performance aspects of the system and reflect the characteristics of controllers. To mitigate this threat and assess the controller from a comprehensive aspect, we apply multiple specifications on each system from multiple angles (trying to be as comprehensive as possible) and categorize them into major and minor cases to validate the effectiveness of the ensemble approaches we proposed. Moreover, we also assign two metrics, SATE and MAE, for each specification to obtain a detailed understanding of the general satisfaction rates and the concrete error values. Thus, we believe our evaluation metrics are appropriate, and the results are convincing, to our best extent.

External threats. Generality to diverse CPSs beyond the studied subject CPSs could always be an external threat, due to the diversity of tasks, specifications, state dimensionalities and controller outputs, making our results might not always be generalized to other CPSs. To mitigate this threat, we select a diverse set of candidate CPSs from various industrial domains that cover a group of widely studied and representative tasks. In addition, we also trained multiple types of RL controllers with different learning algorithms, reward functions and agent configurations to extensively explore the behaviour space and the capabilities of ensemble methods.

7 Related Work

Many algorithms and approaches have been proposed by researchers to study the state abstraction and ensemble methods in diverse aspects such as supervised learning, active learning, transfer learning. However, limited literature considers adopting these methods to CPSs with AI controllers.

Ensemble methods in CPS. Song *et al*. [13] build a publicly available AI-CPS benchmark as a playground for the community, and we take two systems, ACC and AFC, from their work. They took an early step in controller hybridization, but they only used naive methods to combine different controllers to demonstrate the possibility of hybrid controllers.

There are other existing literature [77], [78], [79], [80], [81], [82], [83], [84], [85], [86] have reviewed the capabilities of the hybridization between ensemble methods and real-world applications. For instance, Liu *et al*. [80] proposed a hybrid ensemble DRL model for wind speed forecasting. They demonstrate that the new DRL ensemble approach can predict accurate results in all wind cases and outperform traditional optimization-based ensemble methods. Moreover, Ghosh *et al*. [81] also proposed a multi-agent DRL ensemble framework to perform an optimized air traffic control. They show that the new approach can dynamically suggest adjustments of aircraft speeds in real-time, and their proposed method has the best evaluation results compared with the other three state-of-the-art benchmark approaches.

The works above differ from our approach as we leverage the specification-oriented semantics to guide an end-to-end DRL ensemble framework. Rather than traditional deterministic ensemble strategies, we utilize a high-level DRL agent as a coordinator to integrate actions in real-time. Moreover, we focus on the systems with multiple operation requirements in parallel to demonstrate the capabilities of balancing multi-requirements in safety-critical environments.

State abstraction. State abstraction has been widely studied by researchers to reduce the size of the state space. Du *et al*. [87], [88] used Markov model-based abstraction to analyze the robustness of stateful deep learning systems. However, they focused on the state space from recurrent neural networks. Unlike CPSs, the states in RNN do not have physical meanings related to system requirements. In addition, they use principle component analysis (PCA) to reduce the state dimensionality and uniformly split state space into regular grids. Their abstraction method does not fit in our case, as each state in CPSs contains physical information from sensors, and the uniform partition cannot preserve the semantics distribution.

Multi-objective control. The multi-objective control tasks contain conflicting operation goals where an equilibrium solution is needed to satisfy each requirement as much as possible [89]. Many traditional control methods and AI approaches have been proposed by researchers to overcome such multi-requirements applications in various domains. Ji *et al*. [90] studied a new multi-objective control strategy for inverter-interfaced distributed generation, which utilizes scenario classification and reference determination to distribute the control authority among different controllers. They showed that by dynamically switching the controllers based on different occasions, the new control strategy could maximize the positive sequence voltage to mitigate the risk of sudden power loss; and suppress the oscillation in active power to extend the long-term life of the capacitor. Their work demonstrates a feasible solution for multi-objective control by strategically activating controllers with different operation objectives.

Chen *et al*. [91] proposed a deep Q-network (DQN)-based agent to achieve multi-objective control for energy management in hybrid electric vehicles. The DQN agent is in charge of controlling the motor speed, the CVT gear ratio, and the engine power to maintain the optimal slip ratio, the engine speed and the fuel consumption, respectively. The authors reported that the single DRL-based strategy is facing the problems like computation cost and high complexity. They pointed out that one possible solution is constructing a distributed parameterized control strategy.

Other works [92], [93], [94], [95] have reported several methods to improve the performance of control systems under different multi-goals applications. We notice that two types of approaches are promising: 1) a multi-stage hybrid structure can help to tackle the priority issues from multi-requirements, and 2) a distributed multi-controller framework can reduce the complexity of control strategy design in large-scale systems.

Ensemble learning in ML. Lee *et al*. [96] presented SUNRISE, a unified ensemble method for off-policy RL algorithms from a Q-ensemble aspect. They integrated various RL agents following two key components: weighted Bellman backups, which re-weight Q-values according to uncertain estimation, and an inference method to select actions based on upper bounds of confidence. Although we have close objectives of unifying a group of RL agents to achieve better control performance, their work greatly differs from ours since the collaborated agents in our approach have diverse mechanics in terms of Q-value generation, that they cannot be ensembles based on re-weighted Q-values. Also, the action selection criteria we proposed are powered by semantics predictions and an additional high-level.

Chen *et al*. [97] introduced an architecture for DRL which is designed to reduce the instability issue in deep Q learning. Their approach is designed to stabilize the Q-value's shake by reducing the variance of target approximation error during the training process. Their results show that this architecture can statistically improve the stability performance on several classical control tasks. Notably, their work focused on improving the training process for a specific type of DRL algorithm; in contrast, our method considers the action synergizing techniques. Namely, with an existing set of DRL controllers, how to efficiently and dynamically manage the control authority in safety-critical tasks.

Besides the applications in control tasks, some literature adopt ensemble methods in prediction [98], optimization [99], detection [100], *etc*. Albahli *et al*. [98] studied the ensemble-based defect prediction method for software products. They fused the outputs from three individual classifiers to build an improved prediction model; then, a DRL agent enclosed in this model is used to capture false alarms in real-time predictions. Although their works targeted software environments that do not contain any physical plants and dynamics like CPS, they demonstrated

that DRL agents, ensemble methods, and predicted information can cooperate to deliver superior performance. Xu *et al*. [101] proposed an adaptive subspace optimization ensemble method to handle the high-dimensional data imbalance problem. They show that a more robust and diverse ensemble system can effectively rebalance the real-world high-dimensional dataset and outperform other state-of-the-art imbalance learning methods. Xiao *et al*. [100] studied a new ensemble learning approach to detect traffic incidents, where individual SVM and KNN models have been combined with a strategy to improve the accuracy of indecent detection. Their study shows that ensemble methods are a promising approach to enhance the robustness and reliability of safety-related systems.

8 Conclusion

We present SIEGE, a semantics-guided ensemble control framework for AI-CPS, which aims to construct an efficient, robust, and reliable control system for multi-objectives, and complex CPSs. A semantics-based abstraction is used to decompose and describe the system status in real-time and predicts the incoming semantics regarding the specification satisfactions. We propose a series of ensemble methods that leverage the semantics predictions to optimally generate synergistic control signals from multiple DRL controllers. Further, we adopt an additional DRL agent as a coordinator to form an end-to-end DRL hierarchical control framework to perform a more flexible ensemble strategy. We performed comprehensive evaluations to investigate the performance of different individual controllers and ensemble methods on five industry-level, complex CPSs. The results demonstrated that SIEGE outperforms the state-of-the-art individual controllers and traditional ensemble methods. Our framework is also helpful in delivering a more robust, efficient, and safety-assured control system. To facilitate further research along this direction, we made our source code and experimental details publicly available on our project website.

With the increasing trend of CPSs adopting AI components in the loop, we hope our early exploratory work in this paper could inspire further extensive research along this direction towards providing better quality, safety and reliability assurance techniques for the upcoming era of AI-enabled cyber-physical systems.

Acknowledgment

This work was supported in part by Canada First Research Excellence Fund as part of the University of Alberta's Future Energy Systems research initiative, Canada CIFAR AI Chairs Program, Amii RAP program, the Natural Sciences and Engineering Research Council of Canada (NSERC No.RGPIN-2021-02549, No.RGPAS-2021-00034, No.DGECR-2021-00019), as well as JSPS KAKENHI Grant No.JP20H04168, No.JP21H04877, JST-Mirai Program Grant No.JPMJMI20B8.

References

[1] R. Baheti and H. Gill, "Cyber-physical systems," *The impact of control technology*, vol. 12, no. 1, pp. 161–166, 2011.

[2] L. Brunke, M. Greeff, A. W. Hall, Z. Yuan, S. Zhou, J. Panerati, and A. P. Schoellig, "Safe learning in robotics: From learning-based control to safe reinforcement learning," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, pp. 411–444, 2022.

[3] L. Buşoniu, T. de Bruin, D. Tolić, J. Kober, and I. Palunko, "Reinforcement learning for control: Performance, stability, and deep approximators," *Annual Reviews in Control*, vol. 46, pp. 8–28, 2018.

[4] S. Levine, "Reinforcement learning and control as probabilistic inference: Tutorial and review," *arXiv preprint arXiv:1805.00909*, 2018.

[5] Q. Huang, R. Huang, W. Hao, J. Tan, R. Fan, and Z. Huang, "Adaptive power system emergency control using deep reinforcement learning," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1171–1182, 2019.

[6] Z. Xu, J. Tang, C. Yin, Y. Wang, and G. Xue, "Experience-driven congestion control: When multi-path tcp meets deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1325–1336, 2019.

[7] W. Liu, N. Mehdipour, and C. Belta, "Recurrent neural network controllers for signal temporal logic specifications subject to safety constraints," *IEEE Control Systems Letters*, vol. 6, pp. 91–96, 2021.

[8] S. A. Nivison and P. Khargonekar, "A sparse neural network approach to model reference adaptive control with hypersonic flight applications," in *2018 AIAA Guidance, Navigation, and Control Conference*, 2018, p. 0842.

[9] S. A. Nivison and P. P. Khargonekar, "Development of a robust deep recurrent neural network controller for flight applications," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 5336–5342.

[10] P. Radanliev, D. De Roure, M. Van Kleek, O. Santos, and U. Ani, "Artificial intelligence in cyber physical systems," *AI & society*, vol. 36, no. 3, pp. 783–796, 2021.

[11] C. S. Wickramasinghe, D. L. Marino, K. Amarasinghe, and M. Manic, "Generalization of deep learning for cyber-physical system security: A survey," in *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*. IEEE, 2018, pp. 745–751.

[12] P. Pathak, P. R. Pal, M. Shrivastava, and P. Ora, "Fifth revolution: Applied ai & human intelligence with cyber physical systems," *International Journal of Engineering and Advanced Technology*, vol. 8, no. 3, pp. 23–27, 2019.

[13] J. Song, D. Lyu, Z. Zhang, Z. Wang, T. Zhang, and L. Ma, "When cyber-physical systems meet ai: A benchmark, an evaluation, and a way forward," in *2022 IEEE/ACM 44th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2022, pp. 343–352.

[14] I. O. for Standardization, "ISO 26262:Road vehicles - Functional safety," 2011.

[15] ——, "ISO/PAS 21448: Road vehicles - Safety of the intended functionality," 2019.

[16] S. Hudson, R. Shyama Sundar, and S. Koppu, "Fault control using triple modular redundancy (tmr)," in *Progress in Computing, Analytics and Networking*. Springer, 2018, pp. 471–480.

[17] C. Engelmann, H. H. Ong, and S. L. Scott, "The case for modular redundancy in large-scale high performance computing systems," in *Proceedings of the 8th IASTED international conference on parallel and distributed computing and networks (PDCN)*, 2009, pp. 189–194.

[18] R. E. Lyons and W. Vanderkulk, "The use of triple-modular redundancy to improve computer reliability," *IBM journal of research and development*, vol. 6, no. 2, pp. 200–209, 1962.

[19] A. H. El-Maleh and F. C. Oughali, "A generalized modular redundancy scheme for enhancing fault tolerance of combinational circuits," *Microelectronics Reliability*, vol. 54, no. 1, pp. 316–326, 2014.

[20] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1249, 2018.

[21] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, vol. 14, no. 2, pp. 241–258, 2020.

[22] Z.-H. Zhou, "Ensemble learning," in *Machine learning*. Springer, 2021, pp. 181–210.

[23] R. Polikar, "Ensemble learning," in *Ensemble machine learning*. Springer, 2012, pp. 1–34.

[24] H. M. Gomes, J. P. Barddal, F. Enembreck, and A. Bifet, "A survey on ensemble learning for data stream classification," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–36, 2017.

[25] H.-I. Suk, S.-W. Lee, D. Shen, A. D. N. Initiative *et al.*, "Deep ensemble learning of sparse regression models for brain disease diagnosis," *Medical image analysis*, vol. 37, pp. 101–113, 2017.

[26] A. Chakravarty, J. M. Carlson, R. S. Khetani, and R. H. Gross, "A novel ensemble learning method for de novo computational identification of dna binding sites," *BMC bioinformatics*, vol. 8, no. 1, pp. 1–15, 2007.

[27] F. Avellaneda, "Learning optimal decision trees from large datasets," *arXiv preprint arXiv:1904.06314*, 2019.

[28] J. Holtmann, R. Bernijazov, M. Meyer, D. Schmelter, and C. Tschirner, "Integrated and iterative systems engineering and software requirements engineering for technical systems," *Journal of Software: Evolution and Process*, vol. 28, no. 9, pp. 722–743, 2016.

[29] K. Wan, D. Hughes, K. L. Man, and T. Krilavičius, "Composition challenges and approaches for cyber physical systems," in *2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications*. IEEE, 2010, pp. 1–7.

[30] T. Bures, D. Weyns, B. Schmer, E. Tovar, E. Boden, T. Gabor, I. Gerostathopoulos, P. Gupta, E. Kang, A. Knauss *et al.*, "Software engineering for smart cyber-physical systems: Challenges and promising solutions," *ACM SIGSOFT Software Engineering Notes*, vol. 42, no. 2, pp. 19–24, 2017.

[31] J. Shi, J. Wan, H. Yan, and H. Suo, "A survey of cyber-physical systems," in *2011 international conference on wireless communications and signal processing (WCSP)*. IEEE, 2011, pp. 1–6.

[32] J. Al-Jaroodi, N. Mohamed, I. Jawhar, and S. Lazarova-Molnar, "Software engineering issues for cyber-physical systems," in *2016 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE, 2016, pp. 1–6.

[33] J. O. Ringert, B. Rumpe, C. Schulze, and A. Wortmann, "Teaching agile model-driven engineering for cyber-physical systems," in *2017 IEEE/ACM 39th International Conference on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET)*. IEEE, 2017, pp. 127–136.

[34] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *Proceedings of the 8th International Conference on Formal Modeling and Analysis of Timed Systems*, ser. FORMATS'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 92–106.

[35] R. Bellman, "A markovian decision process," *Journal of mathematics and mechanics*, pp. 679–684, 1957.

[36] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[37] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[38] A. Donzé and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, 2010, pp. 92–106.

[39] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[40] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *ICML*, 2018, pp. 1861–1870.

[41] T. T. Johnson, D. Manzanas Lopez, P. Musau, H.-D. Tran, E. Botoeva, F. Leofante, A. Maleki, C. Sidrane, J. Fan, and C. Huang, "ARCH-COMP20 category report: Artificial intelligence and neural network control systems (AINNCS) for continuous and hybrid systems plants," in *ARCH20. 7th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH20)*, ser. EPiC Series in Computing, G. Frehse and M. Althoff, Eds., vol. 74. EasyChair, 2020, pp. 107–139.

[42] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[43] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[44] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, J. Pineau *et al.*, "An introduction to deep reinforcement learning," *Foundations and Trends® in Machine Learning*, vol. 11, no. 3-4, pp. 219–354, 2018.

[45] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[46] G. Konidaris, "On the necessity of abstraction," *Current opinion in behavioral sciences*, vol. 29, pp. 1–7, 2019.

[47] O. Biza and R. Platt, "Online abstraction with mdp homomorphisms for deep learning," *arXiv preprint arXiv:1811.12929*, 2018.

[48] M. Shanahan and M. Mitchell, "Abstraction for deep reinforcement learning," *arXiv preprint arXiv:2202.05839*, 2022.

[49] D. Abel, D. Arumugam, L. Lehnert, and M. Littman, "State abstractions for lifelong reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 10–19.

[50] T. M. Moerland, J. Broekens, and C. M. Jonker, "Model-based reinforcement learning: A survey," *arXiv preprint arXiv:2006.16712*, 2020.

[51] M. Verleysen and D. François, "The curse of dimensionality in data mining and time series prediction," in *International work-conference on artificial neural networks*. Springer, 2005, pp. 758–770.

[52] M. A. Ganaie, M. Hu *et al.*, "Ensemble deep learning: A review," *arXiv preprint arXiv:2104.02395*, 2021.

[53] R. Atallah and A. Al-Mousa, "Heart disease detection using machine learning majority voting ensemble method," in *2019 2nd international conference on new trends in computing sciences (ictcs)*. IEEE, 2019, pp. 1–6.

[54] G. Elkiran, V. Nourani, and S. Abba, "Multi-step ahead modelling of river water quality parameters using ensemble artificial intelligence-based approach," *Journal of Hydrology*, vol. 577, p. 123962, 2019.

[55] T. Hastie, S. Rosset, J. Zhu, and H. Zou, "Multi-class adaboost," *Statistics and its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[56] C. Zhang and Y. Ma, *Ensemble machine learning: methods and applications*. Springer, 2012.

[57] L. Matignon, G. J. Laurent, and N. L. Fort-Piat, "Reward function and initial values: Better choices for accelerated goal-directed reinforcement learning," in *International Conference on Artificial Neural Networks*. Springer, 2006, pp. 840–849.

[58] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[59] S. Pateria, B. Subagdja, A.-h. Tan, and C. Quek, "Hierarchical reinforcement learning: A comprehensive survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 5, pp. 1–35, 2021.

[60] Z. Zhang, D. Lyu, P. Arcaini, L. Ma, I. Hasuo, and J. Zhao, "Falsifai: Falsification of ai-enabled hybrid control systems guided by time-aware coverage criteria," *IEEE Transactions on Software Engineering*, pp. 1–17, 2022.

[61] X. Jin, J. V. Deshmukh, J. Kapinski, K. Ueda, and K. Butts, "Powertrain control verification benchmark," in *HSCC*, 2014, pp. 253–262.

[62] Z. Zhang, D. Lyu, P. Arcaini, L. Ma, I. Hasuo, and J. Zhao, "FalsifAI: Falsification of AI-Enabled Hybrid Control Systems Guided by Time-Aware Coverage Criteria," Apr. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.6464099

[63] ——, "Effective hybrid system falsification using monte carlo tree search guided by QB-robustness," in *Computer Aided Verification*, A. Silva and K. R. M. Leino, Eds. Cham: Springer International Publishing, 2021, pp. 595–618.

[64] Z. Zhang, P. Arcaini, and X. Xie, "Online reset for signal temporal logic monitoring," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 11, pp. 4421–4432, 2022.

[65] T. Dreossi, T. Dang, A. Donzé, J. Kapinski, X. Jin, and J. V. Deshmukh, "Efficient guiding strategies for testing of temporal properties of hybrid systems," in *NASA Formal Methods: 7th International Symposium, NFM 2015, Pasadena, CA, USA, April 27-29, 2015, Proceedings 7*. Springer, 2015, pp. 127–142.

[66] J. V. Deshmukh, A. Donzé, S. Ghosh, X. Jin, G. Juniwal, and S. A. Seshia, "Robust online monitoring of signal temporal logic," *Formal Methods in System Design*, vol. 51, pp. 5–30, 2017.

[67] Mathworks, "Adaptive cruise control system using model predictive control," 2021. [Online]. Available: https://www.mathworks.com/help/mpc/ug/adaptive-cruise-control-using-model-predictive-controller.html

[68] ——, "Lane keeping assist system using model predictive control," 2021. [Online].

Available: https://www.mathworks.com/help/mpc/ug/lane-keeping-assist-system-using-model-predictive-control.html

[69] ——, "Automatic parking valet with unreal engine simulation," 2021. [Online]. Available: https://www.mathworks.com/help/reinforcement-learning/ug/automatic-parking-valet-with-mpc-and-unreal-engine.html

[70] KINOVA, "Kinova gen3 ultra lightweight robot," 2022. [Online]. Available: https://www.kinovarobotics.com/product/gen3-robots.html

[71] Mathworks, "Ball balance control," 2021. [Online]. Available: https://www.mathworks.com/help/reinforcement-learning/ug/train-sac-agent-for-ball-balance-control.html

[72] M. Riedmiller, "Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method," in *Machine Learning: ECML 2005: 16th European Conference on Machine Learning, Porto, Portugal, October 3-7, 2005. Proceedings 16*. Springer, 2005, pp. 317–328.

[73] M. Massaoudi, S. S. Refaat, I. Chihi, M. Trabelsi, F. S. Oueslati, and H. Abu-Rub, "A novel stacked generalization ensemble-based hybrid lgbm-xgb-mlp model for short-term load forecasting," *Energy*, vol. 214, p. 118874, 2021.

[74] S. Butte, A. Prashanth, and S. Patil, "Machine learning based predictive maintenance strategy: a super learning approach with deep neural networks," in *2018 IEEE Workshop on Microelectronics and Electron Devices (WMED)*. IEEE, 2018, pp. 1–5.

[75] M. R. Mahmoudi, H. Akbarzadeh, H. Parvin, S. Nejatian, V. Rezaie, and H. Alinejad-Rokny, "Consensus function based on cluster-wise two level clustering," *Artificial Intelligence Review*, vol. 54, no. 1, pp. 639–665, 2021.

[76] J. Vandoni, E. Aldea, and S. Le Hégarat-Mascle, "Evidential query-by-committee active learning for pedestrian detection in high-density crowds," *International Journal of Approximate Reasoning*, vol. 104, pp. 166–184, 2019.

[77] S. I. Moon, H. Hirayama, Y. Tsukamoto, S. Nanba, and H. Shinbo, "Ensemble learning method-based slice admission control for adaptive ran," in *2020 IEEE Globecom Workshops (GC Wkshps*. IEEE, 2020, pp. 1–6.

[78] M. Savargiv, B. Masoumi, and M. R. Keyvanpour, "A new ensemble learning method based on learning automata," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–16, 2020.

[79] J. L. C. Villacrés, Z. Zhao, T. Braun, and Z. Li, "A particle filter-based reinforcement learning approach for reliable wireless indoor positioning," *IEEE journal on selected areas in communications*, vol. 37, no. 11, pp. 2457–2473, 2019.

[80] H. Liu, C. Yu, H. Wu, Z. Duan, and G. Yan, "A new hybrid ensemble deep reinforcement learning model for wind speed short term forecasting," *Energy*, vol. 202, p. 117794, 2020.

[81] S. Ghosh, S. Laguna, S. H. Lim, L. Wynter, and H. Poonawala, "A deep ensemble multi-agent reinforcement learning approach for air traffic control," *arXiv preprint arXiv:2004.01387*, 2020.

[82] C. Chen and H. Liu, "Dynamic ensemble wind speed prediction model based on hybrid deep reinforcement learning," *Advanced Engineering Informatics*, vol. 48, p. 101290, 2021.

[83] X. Liu, M. Qin, Y. He, X. Mi, and C. Yu, "A new multi-data-driven spatiotemporal pm2. 5 forecasting model based on an ensemble graph reinforcement learning convolutional network," *Atmospheric Pollution Research*, vol. 12, no. 10, p. 101197, 2021.

[84] R. Saphal, B. Ravindran, D. Mudigere, S. Avancha, and B. Kaul, "Seerl: Sample efficient ensemble reinforcement learning," *arXiv preprint arXiv:2001.05209*, 2020.

[85] T. Liu, Z. Tan, C. Xu, H. Chen, and Z. Li, "Study on deep reinforcement learning techniques for building energy consumption forecasting," *Energy and Buildings*, vol. 208, p. 109675, 2020.

[86] D. C. Rose, J. F. Mair, and J. P. Garrahan, "A reinforcement learning approach to rare trajectory sampling," *New Journal of Physics*, vol. 23, no. 1, p. 013013, 2021.

[87] X. Du, X. Xie, Y. Li, L. Ma, Y. Liu, and J. Zhao, "Deepstellar: Model-based quantitative analysis of stateful deep learning systems," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019, pp. 477–487.

[88] X. Du, Y. Li, X. Xie, L. Ma, Y. Liu, and J. Zhao, "Marble: Model-based robustness analysis of stateful deep learning systems," in *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering*, ser. ASE '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 423–435.

[89] S. F. da Silva, J. J. Eckert, F. L. Silva, L. C. Silva, and F. G. Dedini, "Multi-objective optimization design and control of plug-in hybrid electric vehicle powertrain for minimization of energy consumption, exhaust emissions and battery degradation," *Energy Conversion and Management*, vol. 234, p. 113909, 2021.

[90] L. Ji, J. Shi, Q. Hong, Y. Fu, X. Chang, Z. Cao, Y. Mi, Z. Li, and C. Booth, "A multi-objective control strategy for three phase grid-connected inverter during unbalanced voltage sag," *IEEE Transactions on Power Delivery*, vol. 36, no. 4, pp. 2490–2500, 2020.

[91] J. Chen, H. Shu, X. Tang, T. Liu, and W. Wang, "Deep reinforcement learning-based multi-objective control of hybrid power system combined with road recognition under time-varying environment," *Energy*, vol. 239, p. 122123, 2022.

[92] K. Lee, S. Lee, and J. Lee, "Interactive character animation by learning multi-objective control," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 6, pp. 1–10, 2018.

[93] Ó. Gonzales-Zurita, J.-M. Clairand, E. Peñalvo-López, and G. Escrivá-Escrivá, "Review on multi-objective control strategies for distributed generation on inverter-based microgrids," *Energies*, vol. 13, no. 13, p. 3483, 2020.

[94] J. Jin and X. Ma, "A multi-objective agent-based control approach with application in intelligent traffic signal system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3900–3912, 2019.

[95] A. Rodríguez-Molina, E. Mezura-Montes, M. G. Villarreal-Cervantes, and M. Aldape-Pérez, "Multi-objective meta-heuristic optimization in intelligent control: A survey on the controller tuning problem," *Applied Soft Computing*, vol. 93, p. 106342, 2020.

[96] K. Lee, M. Laskin, A. Srinivas, and P. Abbeel, "Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6131–6141.

[97] X.-l. Chen, L. Cao, C.-x. Li, Z.-x. Xu, and J. Lai, "Ensemble network architecture for deep reinforcement learning," *Mathematical Problems in Engineering*, vol. 2018, 2018.

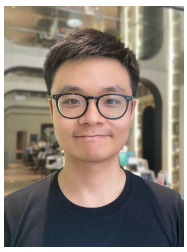
[98] S. Albahli, "A deep ensemble learning method for effort-aware just-in-time defect prediction," *Future Internet*, vol. 11, no. 12, p. 246, 2019.

[99] M. Mirmozaffari, M. Yazdani, A. Boskabadi, H. Ahady Dolatsara, K. Kabirifar, and N. Amiri Golilarz, "A novel machine learning approach combined with optimization models for eco-efficiency evaluation," *Applied Sciences*, vol. 10, no. 15, p. 5210, 2020.

[100] J. Xiao, "Svm and knn ensemble learning for traffic incident detection," *Physica A: Statistical Mechanics and its Applications*, vol. 517, pp. 29–35, 2019.

[101] Y. Xu, Z. Yu, C. P. Chen, and Z. Liu, "Adaptive subspace optimization ensemble method for high-dimensional imbalanced data classification," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

Jiayang Song is a PhD student at University of Alberta, Canada. He received his B.E. from Western University, Canada and his M.E. from University of Toronto, Canada. His research topics include testing, analysis, repairing and enhancement of AI systems and their applications to quality assurance of trustworthy AI-enabled cyber-physical systems.



Xuan Xie is a PhD student at University of Alberta, Canada. He obtained his B.E. from Sun Yat-sen University, China, and his master's degree from Max Planck Institute for Software Systems, Germany. His research topics include analysis, testing, and monitoring of AI-enabled cyber-physical systems, neural network verification and software testing and verification.

Lei Ma is currently an associate professor with The University of Tokyo, Japan and University of Alberta, Canada. He was honorably selected as a Canada CIFAR AI Chair and Fellow at Alberta Machine Intelligence Institute (Amii). Previously, he received the B.E. degree from Shanghai Jiao Tong University, China, and the M.E. and Ph.D. degrees from The University of Tokyo, Japan. His recent research centers around the interdisciplinary fields of soft- ware engineering (SE) and trustworthy artificial intelligence with a special focus on the quality, reliability, safety and security aspects of AI Systems.