Intelligent Resilience Testing for Decision-Making Agents With Dual-Mode Surrogate Adaptation

Jingxuan Yang, Weichao Xu, Yuchen Shi, Yi Zhang, *Senior Member, IEEE*, Shuo Feng, *Member, IEEE*, and Huaxin Pei

***Abstract*—Testing and evaluating decision-making agents remain challenging due to unknown system architectures, limited access to internal states, and the vastness of high-dimensional scenario spaces. Existing testing approaches often rely on surrogate models of decision-making agents to generate large-scale scenario libraries. However, discrepancies between surrogate models and real decision-making agents significantly limit their generalizability and practical applicability. To address this challenge, this paper proposes intelligent resilience testing (IRTest), a unified online adaptive testing framework designed to rapidly adjust to diverse decision-making agents. IRTest initializes with an offline-trained surrogate prediction model and progressively reduces the surrogate-to-real gap during testing through two complementary adaptation mechanisms: (i) online neural fine-tuning in data-rich regimes, and (ii) lightweight importance-sampling-based weighting correction in data-limited regimes. A surrogate-guided adaptive sampling strategy is developed to guide scenario generation and balance exploration and exploitation in complex testing spaces. Extensive experiments across varying levels of task complexity and system heterogeneity demonstrate that IRTest consistently improves failure-discovery efficiency, testing robustness, and cross-system generalizability. These results highlight the potential of IRTest as a practical solution for scalable, adaptive, and resilient testing of decision-making agents.**

***Note to Practitioners*—This work considers the critical problem of efficiently testing and validating complex decision-making agents under unknown system architectures and high-dimensional scenario spaces. Efficient testing of decision-making agents is essential for improving system reliability and ensuring safety in real-world deployments, especially in safety-critical applications. Traditional intelligent testing methods mainly rely on offline-trained surrogate models to generate testing scenarios, which often suffer from poor generalization when applied to real agents with different structures or behaviors. In this work, an intelligent resilience testing framework is proposed that enables online adaptive testing by progressively correcting the surrogate-to-real gap through real-time testing feedback. This work shows that the proposed framework can achieve higher failure-discovery efficiency, stronger testing robustness, and better cross-system generalizability than conventional surrogate-based testing methods. As a practical consequence of this research, the proposed IRTest framework can effectively adapt surrogate models under both data-rich and data-limited testing conditions, enabling scalable and reliable testing with limited real-world interactions. This approach can also be applied to a wide range of AI-enabled systems, including autonomous driving, robotics, intelligent transportation systems, and other safety-critical decision-making applications**

***Index Terms*—Intelligent resilience testing, decision-making agents, surrogate model, adaptive adversarial testing.**

Received 10 December 2025; revised 24 April 2026 and 24 August 2026; accepted 30 September 2026. Date of publication 8 October 2026; date of current version 14 October 2026. This article was recommended for publication by Associate Editor Z. Zhang and Editor B. Wang upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62503259 and in part by Beijing Natural Science Foundation under Grant L257004. *(Jingxuan Yang and Weichao Xu contributed equally to this work.) (Corresponding author: Huaxin Pei.)*

Jingxuan Yang, Weichao Xu, and Yuchen Shi are with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: yangjx20@mails.tsinghua.edu.cn; xuwc19@tsinghua.org.cn; shiyuche21@mails.tsinghua.edu.cn).

Yi Zhang is with the Department of Automation, Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing 100084, China, and also with Jiangsu Province Collaborative Innovation Center of Modern Urban Traffic Technologies, Nanjing 210096, China (e-mail: zhyi@mail.tsinghua.edu.cn).

Shuo Feng is with the Department of Automation, the Institute for Embodied Intelligence and Robotics, and Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing 100084, China, and also with Beijing Key Laboratory of Embodied Intelligence Systems, Beijing 100084, China (e-mail: fshuo@tsinghua.edu.cn).

Huaxin Pei is with Dense-AI, Beijing 100084, China (e-mail: phx17@tsinghua.org.cn).

Digital Object Identifier 10.1109/TASE.2026.3741612

I. Introduction

TESTING and evaluating decision-making agents remain challenging in both development and deployment phases [1]. The internal architectures of many decision-making agents are unknown, their real-time operational states may be difficult to access, and their testing scenario spaces are often high-dimensional and complex [2], [3], [4]. A widely adopted approach is to evaluate these agents in naturalistic testing environments (NTEs) and statistically analyze their performance over large-scale tests [5]. However, NTE-based testing often suffers from low efficiency, high cost, and insufficient coverage of critical failure-prone scenarios, particularly in safety-critical applications [6], [7].

With the rapid advancement of machine learning, intelligent testing technology, grounded in the concept of "AI for AI", has demonstrated the ability to identify capability boundaries of decision-making agents and significantly reduce testing costs [8], [9], [10], [11]. Many existing approaches rely on surrogate models of decision-making agents to generate large-scale testing scenario libraries [12], [13], [14]. These knowledge-based or AI-based surrogate models provide an effective means of accelerating testing in high-dimensional spaces while reducing reliance on real-world testing resources.

However, the effectiveness of surrogate-based testing strongly depends on the consistency between the surrogate model and the decision-making agent under test [15]. Scenario libraries generated offline may perform poorly when the target agent differs from the surrogate in architecture, algorithm,

1558-3783 © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

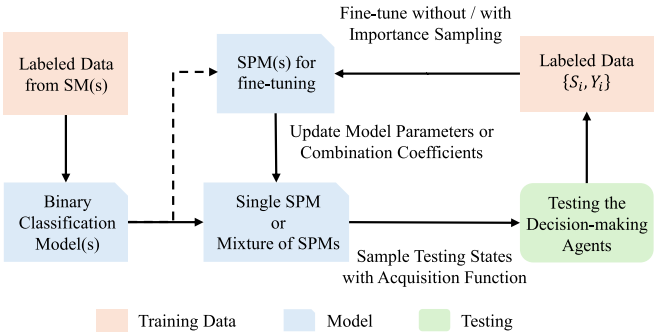


Fig. 1. The workflow of IRTest.

or behavioral characteristics [16]. This surrogate-to-real gap can substantially degrade testing performance and limits the generalizability of existing intelligent testing methods. Therefore, an effective testing method should retain the efficiency offered by surrogate models while adapting to previously unseen decision-making agents.

Recent studies have explored adaptive and learning-based testing techniques using reinforcement learning, adversarial optimization, and generative modeling, as reviewed in Section II. Nevertheless, important limitations remain. Many methods require large amounts of training data or specialized surrogate models, while others lack an efficient mechanism for correcting surrogate-to-real discrepancies in high-dimensional testing spaces. These limitations motivate a central question in this work: how to leverage surrogate models for efficient testing while dynamically correcting their prediction gaps during testing, ensuring both reliability and adaptability?

To address this problem, we introduce intelligent resilience testing (IRTest), an online adaptive testing framework that progressively adapts to different decision-making agents using real-time testing feedback. IRTest starts from a surrogate prediction model (SPM) trained offline and continuously optimizes the SPM during testing. It supports two complementary adaptation modes according to the available testing resources. In data-rich regimes, the SPM is directly fine-tuned using newly collected testing data. In data-limited regimes, multiple SPMs are combined and their weights are adaptively corrected through importance sampling. A surrogate-guided adaptive sampling strategy further uses the optimized SPM to select testing scenarios, balancing the exploration of new failure-prone regions with the exploitation of identified high-risk regions. The overall framework is illustrated in Fig. 1.

The main contributions of this study are summarized as follows:

- **Intelligent Resilience Testing Framework.** We introduce the concept of IRTest, a unified online adaptive testing paradigm that systematically reduces the surrogate-to-real gap and enhances testing robustness across diverse decision-making agents.
- **Dual-Mode Surrogate Adaptation Strategies.** We develop two complementary adaptation mechanisms: (i) an online fine-tuning scheme for data-rich regimes using neural surrogate approximation; and (ii) an importance-sampling-based weighting correction approach for data-limited regimes, ensuring rapid and efficient adaptation.
- **Surrogate-Guided Adaptive Sampling Strategy.** We design a surrogate-guided adaptive sampling strategy that incorporates updated SPM predictions and the naturalistic distribution, enabling IRTest to efficiently explore and exploit high-risk regions of high-dimensional testing spaces.
- **Comprehensive Evaluation Across Testing Regimes.** We conduct extensive experiments demonstrating that IRTest consistently improves failure discovery efficiency, testing stability, and cross-system generalizability.

The rest of the paper is organized as follows. Section II reviews related literature and outlines key challenges in intelligent resilience testing. Section III formulates the problem and presents the proposed methodology. Section IV demonstrates the performance of the proposed framework across various testing conditions. Finally, Section V concludes the paper and discusses future research directions.

II. Related Work

In this section, we review related work from three perspectives: knowledge-based testing methods, AI-based testing methods, and adaptive testing methods.

A. Knowledge-Based Testing Methods

Existing testing methods for decision-making agents often leverage domain knowledge or datasets to design specific scenario generation rules or frameworks. A significant body of research focuses on establishing systematic architectures for testing scenario generation based on prior knowledge. For instance, [17], [18], [19] employ ontologies to structurally represent scenarios, facilitating scenario generation through combinatorial techniques. While these methods offer simplicity and interpretability, they do not account for the dynamic interactions between intelligent agents within the environment.

Another category of approaches models the distributions of scenario parameters by analyzing existing datasets, enabling parameter sampling from these distributions [20], [21], [22]. A commonly used technique in this context is Monte Carlo sampling, which selects parameter values without considering the actual characteristics of the system under test, thereby losing environment-specific contextualization.

Although knowledge-based testing methods perform well in automating testing scenario generation, they are inherently limited when applied to decision-making agents. Due to the high-dimensional nature of the testing space and the "closed-box" characteristics of decision-making agents, knowledge-based methods often result in low testing efficiency and high resource costs, making it challenging to conduct comprehensive testing within limited resources.

B. AI-Based Testing Methods

With the rapid advancement of artificial intelligence, AI-driven testing scenario generation has demonstrated significant advantages, particularly in accelerating the testing

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

process and reducing costs [23], [24], [25], [26], [27]. The main concept of AI-based testing is to learn to predict the criticality of testing scenarios, thereby efficiently generating critical testing scenarios while avoiding redundant testing [28], [29], [30], [31]. These methods are generally designed to achieve two primary testing objectives.

The first objective is to efficiently generate adversarial testing scenarios to expose capability deficiencies in the target system. Techniques such as gradient optimization, domain randomization, and generative models are employed to create adversarial image and point cloud data, aiding in the identification of weaknesses in system perception [32], [33], [34]. Additionally, several studies [35], [36], [37], [38] formulate adversarial scenario generation as an optimization problem and leverage techniques such as deep reinforcement learning to evaluate the decision-making capabilities of decision-making agents.

Beyond adversarial testing, numerous methods have been proposed to accelerate the assessment of critical performance indicators, providing theoretical support for the practical deployment of decision-making agents [39], [40], [41], [42], [43]. Approaches based on deep reinforcement learning and sparse adversarial sampling enable the efficient generation of critical testing scenarios in high-dimensional spaces while statistically ensuring the unbiasedness of test results.

Compared to knowledge-based methods, AI-based testing offers significant advantages in improving testing efficiency and reducing costs. However, these methods heavily depend on the scale and quality of training data, limiting their adaptability to new decision-making agents.

C. Adaptive Testing Methods

To leverage testing information from other decision-making agents, increasing attention has been given to the generation of test scenarios using surrogate models. However, due to discrepancies between surrogate models and the actual systems under test, scenario libraries generated in this manner often lack generalizability, limiting their applicability to decision-making agents with different mechanisms and performance characteristics.

In this context, developing adaptive testing methods is crucial to enable the online optimization and adaptation of testing scenario libraries based on the specific characteristics of the system under test. Existing adaptive search algorithms—such as those based on Gaussian process regression [44], reinforcement learning [45], unsupervised clustering [46], [47], and gradient boosting trees [48], [49]—can generate challenging and diverse test scenarios online for certain decision-making agents without relying on surrogate models [50]. However, these methods are largely restricted to simple scenarios and struggle with inefficiencies in high-dimensional parameter spaces, making them insufficient for testing decision-making agents in complex applications. While some approaches [51], [52], [53], [54] have been designed to handle high-dimensional scenarios, their applicability remains largely confined to connected and automated vehicle systems.

In high-dimensional testing spaces, employing surrogate models to predict the criticality of testing scenarios becomes an essential strategy for intelligent testing. However, the inherent discrepancy between surrogate models and real-world systems remains a fundamental challenge, leading to poor generalizability in intelligent testing techniques. To address this, this paper introduces the concept of intelligent resilience testing, which aims to adaptively refine the surrogate-guided sampling distribution, thereby enabling efficient and comprehensive testing of the system under test.

III. Intelligent Resilience Testing

In this section, we develop IRTest, an online adaptive testing framework that incrementally optimizes the SPMs with the goal of effective and efficient testing for decision-making agents. IRTest trains neural models as SPMs and continuously adapts them during testing, as illustrated in Fig. 2. Specifically, the problem formulation is presented in Section III-A. Section III-B introduces the offline pre-training of the SPM on the surrogate decision-making agent. The adaptive testing procedures for data-rich regimes are detailed in Section III-C, including the surrogate-guided adaptive sampling strategy for selecting the next testing state and the online fine-tuning of the SPM. Finally, Section III-D addresses the challenges of data-limited regimes by presenting a combination-coefficient regression method enhanced with importance sampling. For clarity, the major notations used in the IRTest framework are summarized in Table I.

A. Problem Formulation

Let $f^*(\boldsymbol{s}) := \mathbb{P}_p(F|\boldsymbol{S} = \boldsymbol{s}) \in [0,1],\ \forall \boldsymbol{s} \in \mathcal{S}$ denote the probability of failure of the agents given the initial m states $\boldsymbol{s}$, where F denotes the failure event, $\mathcal{S}$ is the state space. We can use a surrogate prediction model f to predict f^*. Define the surrogate-to-real gap as

$$g(f^*\|f)(\boldsymbol{s}) := f^*(\boldsymbol{s}) - f(\boldsymbol{s}),\ \forall \boldsymbol{s} \in \mathcal{S}. \tag{1}$$

The goal of IRTest is to reduce the magnitude of the surrogate-to-real gap $g(f^*\|f)$ and increase testing efficiency. To achieve this goal, we use testing results to optimize f online.

B. Offline Pre-Training

In IRTest, we employ an SPM f, which maps the testing state space $\mathcal{S}$ to failure probabilities in the range [0,1], to predict the ground truth f^*. The SPM is derived from a surrogate agent model (SAM) operating in the same application environment as the decision-making agent under test. The discrepancy between the SAM and the target agent is characterized in terms of their induced failure regions. Let $\mathcal{F}_{\text{SAM}}$ and $\mathcal{F}_{\text{target}}$ denote the sets of failure states corresponding to the SAM and the target agent, respectively. We assume that $\mathbb{P}_p(\mathcal{F}_{\text{SAM}} \cap \mathcal{F}_{\text{target}}) > 0$, i.e., the two models share a non-negligible overlap in failure regions. This overlap enables the SAM to provide informative guidance for identifying failure-prone states of the target system. In general, a larger overlap between $\mathcal{F}_{\text{SAM}}$ and $\mathcal{F}_{\text{target}}$ leads to better performance of the proposed framework. Due to the complexity of the environment and the closed-box nature of decision-making

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

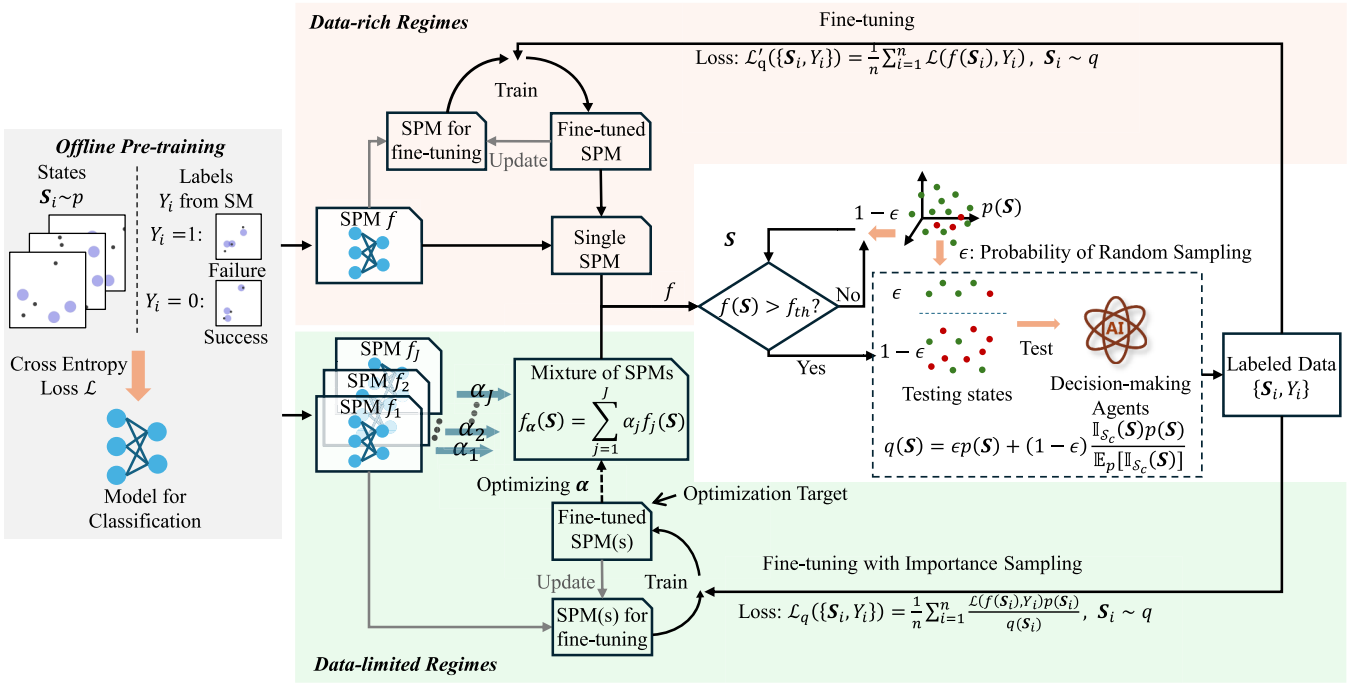


Fig. 2. Illustration of the IRTest framework.

TABLE I
SUMMARY OF NOTATION

Notation	Definition	Notation	Definition
$\mathcal{S}$	Testing state space	$\boldsymbol{s}$	A testing state in $\mathcal{S}$
$\boldsymbol{S}$	Random variable representing the testing state	F	Failure event of the decision-making agent
$p(\boldsymbol{s})$	Naturalistic distribution of testing states	$q(\boldsymbol{s})$	Surrogate-guided sampling distribution
$f^*(\boldsymbol{s})$	Ground-truth failure probability of the target agent	$f(\boldsymbol{s})$	The SPM of $f^*(\boldsymbol{s})$
$g(f^*\|\|f)(\boldsymbol{s})$	Surrogate-to-real gap	f_{th}	Threshold for identifying predicted critical states
$\mathcal{F}_{\text{SAM}}$	Failure-state set of the surrogate agent model	$\mathcal{F}_{\text{target}}$	Failure-state set of the target agent
$\mathcal{S}_c$	Predicted critical-state set	$\mathcal{S}_{-c}$	Complement of $\mathcal{S}_c$
ϵ	Probability of random sampling for exploration	$R_p(q)$	Support-level reachability of q with respect to p
Y_i	Binary outcome of the i-th test	$\mathcal{D}$	Testing dataset collected from the target agent
$f_j(\boldsymbol{s})$	j-th SPM constructed from the corresponding SAM	J	Number of SPMs in the mixture
$\boldsymbol{\alpha}$	Vector of SPM combination coefficients	α_j	Combination coefficient of the j-th SPM
$f_{\boldsymbol{\alpha}}(\boldsymbol{s})$	Mixture of SPMs parameterized by $\boldsymbol{\alpha}$	$f_j^{(t)}$	Updated j-th SPM at iteration t
$\alpha_j^{(t)}$	Combination coefficient of the j-th SPM at iteration t	$J_{\boldsymbol{\alpha}}$	Objective for combination-coefficient optimization
$\mathcal{L}$	Binary classification loss	$W(\boldsymbol{s})$	Importance weight, $p(\boldsymbol{s})/q(\boldsymbol{s})$
$\mathcal{S}_+$	Set of positive (failure) samples	$\mathcal{S}_-$	Set of negative (successful) samples
$p'(\boldsymbol{s})$	Rebalanced naturalistic distribution	$q'(\boldsymbol{s})$	Rebalanced sampling distribution
$W'(\boldsymbol{s})$	Importance weight after class rebalancing	T	Maximum simulation time for each test
T_{e}	Maximum number of fine-tuning epochs	T_{f}	Fine-tuning frequency
N	Maximum number of tests	n	Number of samples used in loss estimation

agents, we propose using a neural network (NN), such as a multilayer perceptron (MLP), as the SPM in IRTest.

The task of predicting f^* is modeled as a binary classification problem, where the two categories correspond to whether the agent under test fails in a given state. Using historical testing data from the SAM, we construct an offline training dataset by labeling each input state $\boldsymbol{s}$ with a binary label Y, where $Y = 1$ indicates failure and $Y = 0$ indicates success in scenarios initialized at state $\boldsymbol{s}$. By training the NN on this dataset, the network outputs a confidence score (failure probability) for each input $\boldsymbol{s}$, which is interpreted as the SPM, i.e.,

$$f(\boldsymbol{s}) = \text{NN}(\boldsymbol{s}) \in [0, 1]. \tag{2}$$

Training the SPM f is highly challenging due to the curse of rarity (CoR) [6], i.e., the positive samples (failure cases) are so infrequent that an exceedingly large number of samples is required to obtain sufficient information about these rare events. Furthermore, training f on a highly imbalanced dataset makes it difficult to achieve both high precision and high recall, especially when the imbalance ratio is large (e.g., greater than 10^6). To mitigate the challenges posed by CoR,

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

techniques such as class rebalancing can be applied, in which positive and negative samples are selected with a 1:1 ratio within each batch.

The pre-trained SPM is then used to test the target decision-making agent and is further optimized online during testing to reduce the surrogate-to-real gap and improve testing efficiency, as detailed in Subsections III-C and III-D.

C. Online Optimization in Data-Rich Regimes

The failure probability predicted by the pre-trained SPM will be used to select testing states for evaluating the decision-making agent under test. Additionally, actual testing data from the target system will be leveraged to refine the SPM. In data-rich regimes, a large number of tests can be conducted on the target system to gather sufficient testing data. In this case, the workflow of IRTest will involve sampling testing states using a surrogate-guided adaptive sampling strategy and performing online fine-tuning based on the testing data, as shown in Fig. 2.

1) Surrogate-Guided Adaptive Sampling: IRTest primarily focuses on states where $f(\boldsymbol{s})$ exceeds a given threshold, as these testing cases have a higher probability of triggering failures in the tested system, thereby increasing the proportion of critical testing scenarios among all sampled scenarios. We design a surrogate-guided adaptive sampling distribution to select testing states based on prediction results. Specifically, denote $\mathcal{S}_c := \{\boldsymbol{s} \in \mathcal{S} : f(\boldsymbol{s}) > f_{\text{th}}\}$, where $f_{\text{th}} \in (0,1)$ is a threshold. The surrogate-guided adaptive sampling distribution is then given by

$$q(\boldsymbol{s}) = \epsilon p(\boldsymbol{s}) + (1-\epsilon)\frac{\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{s})p(\boldsymbol{s})}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]}, \ \forall \boldsymbol{s} \in \mathcal{S}, \tag{3}$$

where $\epsilon \in (0,1)$ is a parameter that balances exploration and exploitation, and p denotes the naturalistic distribution. To sample a testing state from q, we first sample a testing state $\boldsymbol{s}$ from p. Then, with probability ϵ, we use $\boldsymbol{s}$ directly as the testing state. Otherwise, we continue sampling states $\boldsymbol{s}'$ from p until $\boldsymbol{s}' \in \mathcal{S}_c$, at which point $\boldsymbol{s}'$ becomes the selected testing state.

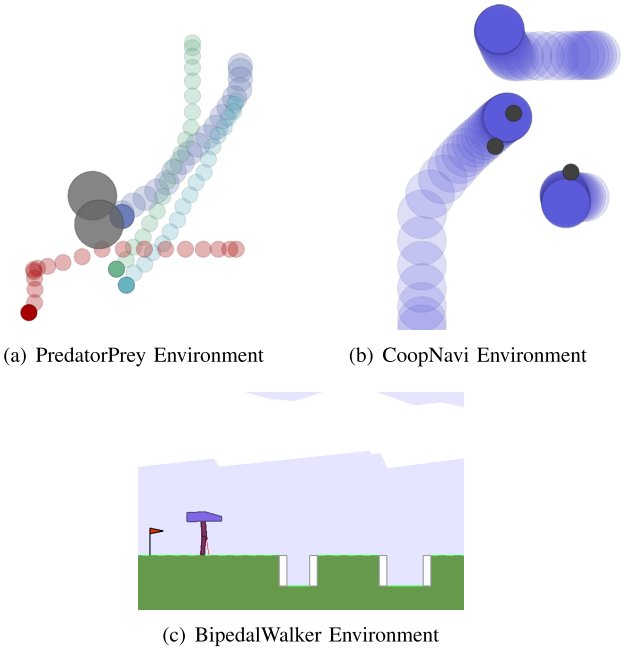
This sampling process, governed by the distribution q and informed by predicted criticality, enables targeted exploration of the testing space, increasing the likelihood of triggering failure events. Furthermore, the introduction of the random sampling probability ϵ ensures that exploration is not overly constrained by focusing exclusively on $\mathcal{S}_c$. It is worth noting that this sampling distribution q ensures that all states with non-zero probability under the natural distribution p remain reachable, as it always samples testing states from p with probability $\epsilon > 0$. Specifically, we define the support-level reachability of $q(\boldsymbol{s})$ with respect to $p(\boldsymbol{s})$ as

$$R_p(q) = \mathbb{P}_p\big(\{\boldsymbol{s} \in \mathcal{S} : q(\boldsymbol{s}) > 0\}\big), \tag{4}$$

i.e., the probability (under p) of states that can be sampled with non-zero probability under q. It is straightforward that $R_p(p) = 1$.

Lemma 1 (Support-Level Reachability): Let the sampling distribution be defined as

$$q(\boldsymbol{s}) = \epsilon p(\boldsymbol{s}) + (1-\epsilon)\frac{\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{s})p(\boldsymbol{s})}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]}, \quad \forall \boldsymbol{s} \in \mathcal{S}, \tag{5}$$



(a) PredatorPrey Environment (b) CoopNavi Environment

(c) BipedalWalker Environment

Fig. 3. Illustration of the PredatorPrey, CoopNavi, and BipedalWalker environments.

where $\epsilon \in (0,1)$. Then, the support-level reachability of q satisfies $R_p(q) = 1$.

Proof: Since $\epsilon > 0$, we have $q(\boldsymbol{s}) \geq \epsilon p(\boldsymbol{s})$ for all $\boldsymbol{s}$. Therefore, for any $\boldsymbol{s}$ such that $p(\boldsymbol{s}) > 0$, it holds that $q(\boldsymbol{s}) > 0$. This implies

$$\{\boldsymbol{s} \in \mathcal{S} : p(\boldsymbol{s}) > 0\} \subseteq \{\boldsymbol{s} \in \mathcal{S} : q(\boldsymbol{s}) > 0\}. \tag{6}$$

Hence,

$$R_p(q) = \mathbb{P}_p(\{\boldsymbol{s} \in \mathcal{S} : q(\boldsymbol{s}) > 0\}) = 1. \tag{7}$$

□

Remark 1: The above result provides a support-level guarantee, indicating that all states with non-zero probability under $p(\boldsymbol{s})$ remain reachable under $q(\boldsymbol{s})$. However, this does not imply that all such states will be observed under a finite sampling budget, especially in high-dimensional settings. In practice, the effectiveness of support-level reachability depends on the number of samples and the underlying distribution.

2) Fine-Tuning of SPM: To identify the failure cases of agents with high efficiency and high coverage, we propose generating testing states with q. The performance of the target system is then evaluated in scenarios initiated from these testing states. To enhance the precision and recall of the SPM for the decision-making agent under test, the SPM is fine-tuned by training on the set of visited states $\mathcal{D}$. Each testing state $\boldsymbol{s}_i$ sampled with q, along with the binary label Y_i determined by the actual performance of the agents under test, is stored as a state-label pair to form $\mathcal{D}$. We train the SPM on $\mathcal{D}$ using a procedure similar to the offline pre-training process. However, this fine-tuning is challenging due to the risk of catastrophic forgetting, where after being trained on $\mathcal{D}$, the SPM f may forget the patterns learned before, leading to significant prediction errors. Existing methods designed to address catastrophic forgetting can be broadly categorized into

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

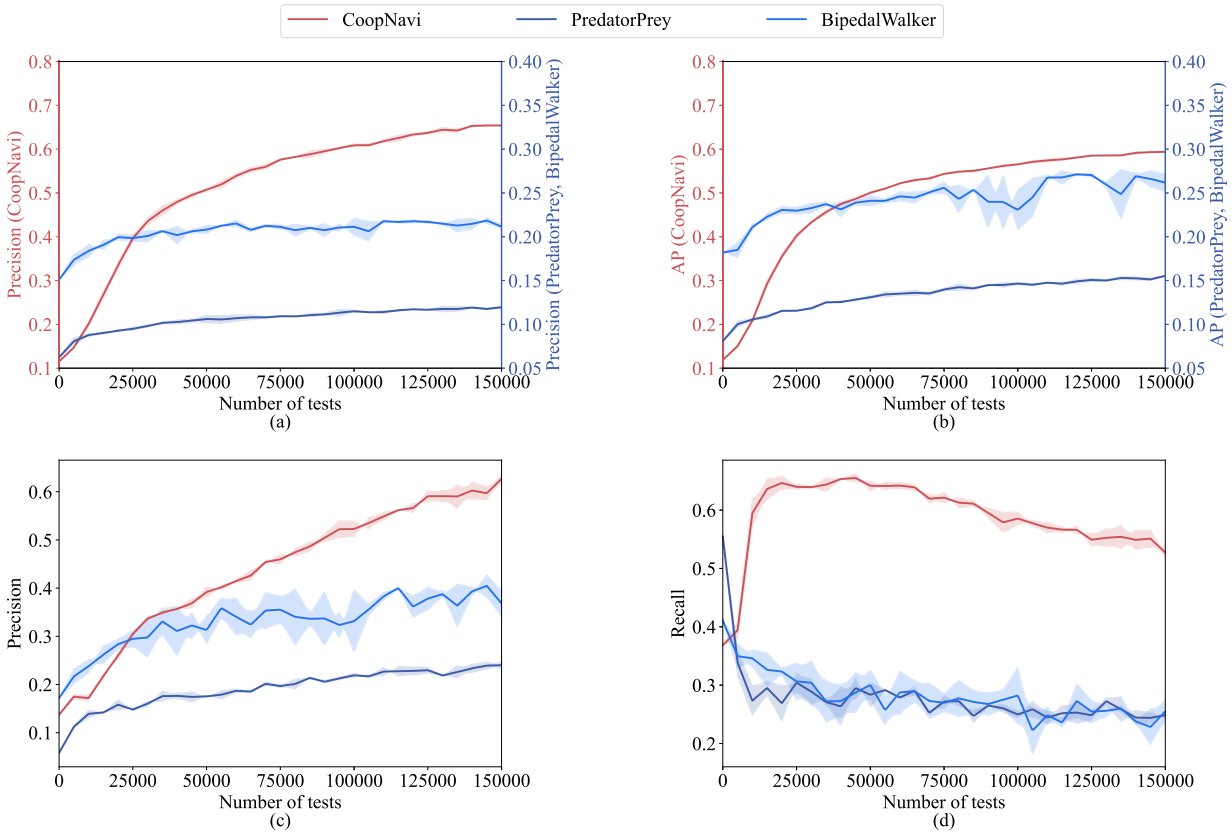


Fig. 4. IRTest-R results: precision when recall is 0.5, average precision, precision and recall across the three environments, where $\epsilon = 0.2$ and classification threshold $f_{\text{th}} = 0.5$. The shaded regions indicate one standard deviation above and below the mean across repeated runs.

three types: replay methods, regularization-based methods, and parameter-isolation methods [55].

In this paper, we adopt a simple yet effective approach to mitigate catastrophic forgetting. Specifically, we use a fixed number of training epochs to fine-tune the SPM at a fixed frequency. This design is motivated by the non-stationary nature of the testing data distribution. Specifically, as the testing process evolves, the collected data increasingly deviates from the natural distribution of the environment, making it difficult to construct a representative validation set for reliable convergence assessment. Therefore, instead of relying on validation-based early stopping, we adopt a fixed-epoch training strategy as a practical alternative to mitigate overfitting. The number of epochs is treated as a tunable hyperparameter. The fixed number of epochs does not require the SPM to fully converge at each update. Instead, it limits the magnitude of model adaptation to newly collected data at each testing stage. This helps prevent excessive fitting to recent failure patterns, while repeated fine-tuning over successive testing iterations allows the SPM to progressively incorporate new information. Therefore, the overall adaptation is achieved through a sequence of controlled incremental updates rather than complete convergence at each individual stage. Empirical results in Figs. 4 and 5 demonstrate that, under the adopted hyperparameter setting, repeated fixed-epoch fine-tuning progressively improves the SPM performance without observable performance degradation caused by overfitting.

The overall procedures of IRTest in data-rich regimes are summarized in Algorithm 1. During the testing process, each testing state is sampled from the distribution q, using the information of the surrogate model to enhance testing efficiency. The testing results are added to the set of visited states $\mathcal{D}$. Once a predefined number of tests T_{f} have been conducted, the testing is paused and the SPM is trained on $\mathcal{D}$ for a fixed number of epochs T_{e} to fine-tune it online. After online optimization, $\mathcal{S}_c$ is updated with the new f, and testing continues with new states generated from q. This process is repeated until the termination conditions defined by the tester are met.

D. Online Optimization in Data-Limited Regimes

Due to the large number of parameters in neural networks, a substantial amount of data is required to achieve optimal results when fine-tuning the SPM directly on the training data. However, in practice, testing the target system often encounters the challenge of limited testing resources, primarily due to high testing cost, which necessitates using a small number of tests to achieve effective results. To address this challenge, an alternative weighting correction approach based on importance sampling is proposed for data-limited regimes, as illustrated in Fig. 2. The weighting correction method shares the same surrogate-guided adaptive sampling distribution as described in Section III-C and follows similar testing and optimization procedures. The key distinction between online optimization

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

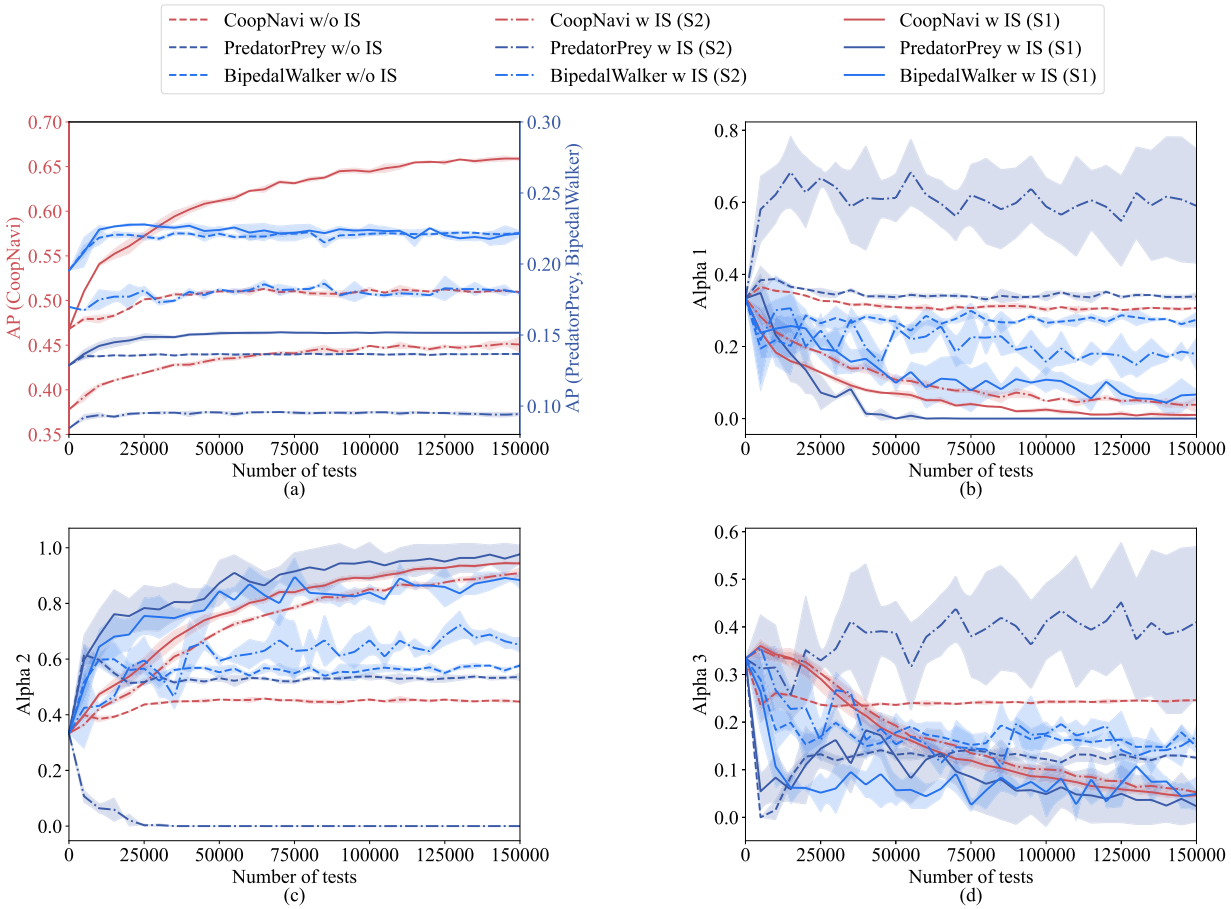


Fig. 5. IRTest-L results: AP and combination coefficients across the three environments, where $\epsilon = 0.05$ and classification threshold $f_{\text{th}} = 0.5$. The shaded regions indicate one standard deviation above and below the mean across repeated runs.

methods in different regimes lies in the acquisition and optimization of SPMs.

1) Combination Coefficient Optimization: To reduce the testing data required by the IRTest process, we use a mixture of SPMs as the predictor f. Specifically, the mixture of SPMs is given by

$$f_{\boldsymbol{\alpha}}(\boldsymbol{s}) := \sum_{j=1}^{J} \alpha_j f_j(\boldsymbol{s}), \ \ \forall \boldsymbol{s} \in \mathcal{S}, \tag{8}$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_J]^\top \in \mathbb{R}^J$ is the vector of combination coefficients, and f_j denotes the j-th SPM obtained from different SAMs. Thus, the prediction function f in data-limited regimes can be viewed as the mixture of SPMs, i.e., $f(\boldsymbol{s}) = f_{\boldsymbol{\alpha}}(\boldsymbol{s})$. In this case, the online optimization problem reduces to the task of optimizing the combination coefficients $\boldsymbol{\alpha}$ of the SPMs.

Let $\mathcal{D}$ represent the set of visited states. Each SPM is updated with newly acquired data $\mathcal{D}$ to obtain updated models. These updated SPMs are then combined using the previously determined weights to optimize the combination coefficients. The corresponding optimization problem is formulated as

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \ \ & J_{\boldsymbol{\alpha}} := \frac{1}{2} \sum_{\boldsymbol{s} \in \mathcal{D}} \left[\sum_{j=1}^{J} \alpha_j^{(t-1)} \hat{f}_j^{(t)}(\boldsymbol{s}) - f_{\boldsymbol{\alpha}}(\boldsymbol{s}) \right]^2 \\ \text{s.t.} \ \ & \mathbf{1}^\top \boldsymbol{\alpha} = 1, \quad \boldsymbol{\alpha} \geqslant \mathbf{0}, \end{aligned} \tag{9}$$

where $\alpha_j^{(t)}$ are the combination coefficients at iteration t, $\hat{f}_j^{(t)}$ are the updated SPMs at iteration t, and $t \in \mathbb{N}_{\geqslant 1}$ is the update index. Note that this optimization problem is a quadratic programming (QP) problem whose optimal solutions can be found efficiently. By using combination coefficients regression, the number of parameters to be optimized is reduced to J, thereby lowering the data requirements. The key to solving these problems lies in training $\hat{f}_j^{(t)}$ with newly acquired data to obtain the optimal combination coefficients.

2) Importance Sampling: Due to the discrepancy between the naturalistic distribution p and the sampling distribution q, the training data of the SPMs and the fine-tuning data $\mathcal{D}$ are not drawn from the same distribution. This mismatch introduces learning bias, as the models are trained on datasets with different sampling probabilities, potentially leading to suboptimal combination coefficients. To reduce the learning bias, we propose modifying the loss function by incorporating the importance sampling method [56]. Specifically, the original loss function is given by

$$\mathcal{L}_p(\{\boldsymbol{S}_i, Y_i\}) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(\boldsymbol{S}_i), Y_i), \ \ \boldsymbol{S}_i \sim p, \tag{10}$$

where Y_i are the binary labels indicating whether the agent fails ($Y_i = 1$) or succeeds ($Y_i = 0$) in scenarios starting from

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 Intelligent Resilience Testing for Data-Rich Regimes

Input: naturalistic distribution p, SPM f, max simulation time T, max fine-tuning epoch T_{e}, fine-tuning frequency T_{f}, threshold $f_{\mathrm{th}} \in (0,1)$, probability of random sampling ϵ, max number of tests N

Initialize the set of visited states $\mathcal{D} = \varnothing$;
$i \leftarrow 0$;
while $i < N$ **do**
  $i \leftarrow i + 1$;
  if $\texttt{Uniform}(0,1) < \epsilon$ **then**
    $\boldsymbol{s}_i \leftarrow \texttt{Sampling}(p)$
  else
    $\boldsymbol{s}_i \leftarrow \texttt{Sampling}(p)$;
    while $f(\boldsymbol{s}_i) \leqslant f_{\mathrm{th}}$ **do**
      $\boldsymbol{s}_i \leftarrow \texttt{Sampling}(p)$;
    end
  end
  // Get label Y_i by performing test with state s_i and max simulation time T
  $Y_i \leftarrow \texttt{Testing}(\boldsymbol{s}_i, T)$;
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{s}_i, Y_i)\}$;
  if $i \% T_{\mathrm{f}} == 0$ **then**
    // Train f on $\mathcal{D}$ for T_{e} epochs
    $e \leftarrow 0$;
    while $e < T_{\mathrm{e}}$ **do**
      $f \leftarrow \texttt{Train}(f, \mathcal{D})$;
      $e \leftarrow e + 1$;
    end
end

state $\boldsymbol{S}_i$. Suppose that we use another sampling probability distribution q, the loss becomes

$$\mathcal{L}'_q(\{\boldsymbol{S}_i, Y_i\}) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(f(\boldsymbol{S}_i), Y_i), \ \ \boldsymbol{S}_i \sim q. \tag{11}$$

By applying importance sampling, the adjusted loss function is

$$\mathcal{L}_q(\{\boldsymbol{S}_i, Y_i\}) = \frac{1}{n}\sum_{i=1}^{n} \frac{\mathcal{L}(f(\boldsymbol{S}_i), Y_i)p(\boldsymbol{S}_i)}{q(\boldsymbol{S}_i)}, \ \ \boldsymbol{S}_i \sim q. \tag{12}$$

According to importance sampling theory [56], the importance sampling estimate is unbiased, i.e., $\mathbb{E}_p[\mathcal{L}_p(\{\boldsymbol{S}_i, Y_i\})] = \mathbb{E}_q[\mathcal{L}_q(\{\boldsymbol{S}_i, Y_i\})]$. Let $W := p/q$ be the importance weight, which can be computed as

$$\begin{aligned} W(\boldsymbol{s}) = \frac{p(\boldsymbol{s})}{q(\boldsymbol{s})} &= \frac{p(\boldsymbol{s})}{\epsilon p(\boldsymbol{s}) + (1-\epsilon)\frac{\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{s})p(\boldsymbol{s})}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]}} \\ &= \frac{1}{\epsilon + (1-\epsilon)\frac{\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{s})}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]}} \\ &= \begin{cases} \dfrac{1}{\epsilon + \frac{1-\epsilon}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]}}, & \boldsymbol{s} \in \mathcal{S}_c, \\ \dfrac{1}{\epsilon}, & \boldsymbol{s} \in \mathcal{S}_{-c}, \end{cases} \end{aligned} \tag{13}$$

where $\mathcal{S}_{-c} = \mathcal{S} \setminus \mathcal{S}_c$. To provide theoretical insight into the stability of the importance sampling weights, we present the following result.

Lemma 2 (Bounded Importance Weights): Let the sampling distribution q be defined as

$$q(\boldsymbol{s}) = \epsilon p(\boldsymbol{s}) + (1-\epsilon)\frac{\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{s})p(\boldsymbol{s})}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]}, \ \ \forall \boldsymbol{s} \in \mathcal{S}, \tag{14}$$

where $\epsilon \in (0,1)$. Then, for any $\boldsymbol{s}$ satisfying $p(\boldsymbol{s}) > 0$,

$$W(\boldsymbol{s}) = \frac{p(\boldsymbol{s})}{q(\boldsymbol{s})} \leqslant \frac{1}{\epsilon}. \tag{15}$$

Consequently, the variance of the importance weight is bounded by

$$\mathrm{Var}_q[W(\boldsymbol{S})] \leqslant \frac{1}{\epsilon} - 1. \tag{16}$$

Proof: Since the first component of $q(\boldsymbol{s})$ is $\epsilon p(\boldsymbol{s})$ and the second component is nonnegative, we have

$$q(\boldsymbol{s}) \geqslant \epsilon p(\boldsymbol{s}) \tag{17}$$

for all $s \in \mathcal{S}$. Therefore, for any $\boldsymbol{s}$ such that $p(\boldsymbol{s}) > 0$,

$$W(\boldsymbol{s}) = \frac{p(\boldsymbol{s})}{q(\boldsymbol{s})} \leqslant \frac{1}{\epsilon}. \tag{18}$$

Moreover, by the standard importance sampling identity, $\mathbb{E}_q[W(\boldsymbol{s})] = 1$. Since $0 \leqslant W(\boldsymbol{s}) \leqslant 1/\epsilon$, we have

$$W^2(\boldsymbol{s}) \leqslant \frac{1}{\epsilon} W(\boldsymbol{s}). \tag{19}$$

Taking expectation under q gives

$$\mathbb{E}_q[W^2(\boldsymbol{S})] \leqslant \frac{1}{\epsilon}\mathbb{E}_q[W(\boldsymbol{S})] = \frac{1}{\epsilon}. \tag{20}$$

Hence,

$$\mathrm{Var}_q[W(\boldsymbol{S})] = \mathbb{E}_q[W^2(\boldsymbol{S})] - \mathbb{E}_q^2[W(\boldsymbol{S})] \leqslant \frac{1}{\epsilon} - 1. \tag{21}$$

□

Remark 2: Lemma 2 shows that the random-sampling component in the proposed surrogate-guided adaptive sampling distribution serves two purposes. In addition to maintaining support-level reachability, it acts as a defensive sampling component that prevents the proposal distribution from becoming too small relative to the naturalistic distribution. In particular, even if $q(\boldsymbol{s})$ is small in an absolute sense for a rare state, it always satisfies $q(\boldsymbol{s}) \geqslant \epsilon p(\boldsymbol{s})$, and hence the corresponding importance weight cannot exceed $1/\epsilon$. Therefore, the variance explosion caused by arbitrarily large importance weights is avoided under the proposed sampling scheme. The parameter ϵ controls a trade-off between exploitation and importance-sampling stability: decreasing ϵ increases the emphasis on predicted critical states, while increasing ϵ results in a tighter upper bound on the importance weights and their variance.

Furthermore, if we use the class rebalancing technique to train the SPM, then the original sampling probability distribution is changed to

$$p'(\boldsymbol{s}) = \begin{cases} \dfrac{p(\boldsymbol{s})}{2\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]}, & \boldsymbol{s} \in \mathcal{S}_+, \\ \dfrac{p(\boldsymbol{s})}{2\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_-}(\boldsymbol{S})]}, & \boldsymbol{s} \in \mathcal{S}_-, \end{cases} \tag{22}$$

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

where $\mathcal{S}_+$ is the set of positive samples and $\mathcal{S}_-$ is the set of negative samples. Similarly, the corresponding sampling distribution q' becomes

$$q'(\boldsymbol{s}) = \begin{cases} \dfrac{q(\boldsymbol{s})}{2\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]}, & \boldsymbol{s} \in \mathcal{S}_+, \\ \dfrac{q(\boldsymbol{s})}{2\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_-}(\boldsymbol{S})]}, & \boldsymbol{s} \in \mathcal{S}_-. \end{cases} \tag{23}$$

Therefore, the rebalanced importance weight is

$$\begin{aligned} W'(\boldsymbol{s}) &= \frac{p'(\boldsymbol{s})}{q'(\boldsymbol{s})} \\ &= \begin{cases} \dfrac{\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]}{\left(\epsilon + \frac{1-\epsilon}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]}\right)\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]}, & \boldsymbol{s} \in \mathcal{S}_c \cap \mathcal{S}_+, \\ \dfrac{\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_-}(\boldsymbol{S})]}{\left(\epsilon + \frac{1-\epsilon}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]}\right)\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_-}(\boldsymbol{S})]}, & \boldsymbol{s} \in \mathcal{S}_c \cap \mathcal{S}_-, \\ \dfrac{\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]}{\epsilon\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]}, & \boldsymbol{s} \in \mathcal{S}_{-c} \cap \mathcal{S}_+, \\ \dfrac{\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_-}(\boldsymbol{S})]}{\epsilon\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_-}(\boldsymbol{S})]}, & \boldsymbol{s} \in \mathcal{S}_{-c} \cap \mathcal{S}_-. \end{cases} \end{aligned} \tag{24}$$

For $\boldsymbol{s} \in \mathcal{S}_k$, where $k \in \{+, -\}$, the rebalanced importance weight satisfies

$$W'(\boldsymbol{s}) = W(\boldsymbol{s})\frac{\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_k}(\boldsymbol{S})]}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_k}(\boldsymbol{S})]}. \tag{25}$$

Consequently, provided that both classes have positive probabilities under p,

$$W'(\boldsymbol{s}) \leqslant \frac{1}{\epsilon}\max_{k\in\{+,-\}}\frac{\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_k}(\boldsymbol{S})]}{\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_k}(\boldsymbol{S})]}. \tag{26}$$

Unlike the bound in Lemma 2, this bound depends on the class probabilities.

According to the rebalanced importance weight $W'(\boldsymbol{s})$, testing states belonging to different categories are assigned different importance weights. To compute these weights, three key values need to be numerically calculated: $\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]$, $\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]$ and $\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]$. Here, $\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]$ represents the failure rate when states are sampled according to the naturalistic distribution p, while $\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]$ represents the failure rate when states are sampled according to the new sampling distribution q. The latter is related to the naturalistic failure rate and the precision of the mixture of SPMs f, i.e., $\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})] = \epsilon\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})] + (1-\epsilon)P$, where P denotes the precision of the SPMs. Finally, $\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]$ represents the probability that the prediction of f is positive when sampling states from p.

In practice, accurately estimating the target agent's naturalistic failure rate can require a prohibitively large number of tests, and it is not feasible to directly access the true precision of the SPM. To address this, we sample testing states from $p(\boldsymbol{s})$ and obtain the corresponding evaluation outcomes (e.g., failure indicators) via rollouts of the SAMs. These testing data will be used to estimate quantities such as $\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]$, $\mathbb{E}_q[\mathbb{I}_{\mathcal{S}_+}(\boldsymbol{S})]$, and $\mathbb{E}_p[\mathbb{I}_{\mathcal{S}_c}(\boldsymbol{S})]$. For instance, the probability of positive samples in the testing data is used as an estimate for the failure rate of the agent under test in states sampled from p. It is important to note that training data from the SPMs should not be used for this purpose, as the performance of the SPMs on the training data is typically overly optimistic, which deviates significantly from their performance in real-world scenarios. The estimation of the natural failure rate may be biased, especially under limited testing budgets. A direct estimation based on random testing of the real agents would require a large number of samples when the failure rate is low, which is often impractical [15]. To address this issue, we adopt a surrogate-assisted estimation strategy. Specifically, we leverage SAMs (e.g., different versions of the same algorithm or models trained under similar environments) to approximate the failure behavior. This provides a surrogate-based approximation of the required probabilities, although its accuracy depends on the similarity between the SAMs and the target agent. Based on the combination of multiple SPMs

Algorithm 2 Intelligent Resilience Testing for Data-Limited Regimes

Input: naturalistic distribution p, SPMs f_j and their testing data $\mathcal{D}_j$, $j = 1, \ldots, J$, max simulation time T, max fine-tuning epoch T_{e}, fine-tuning frequency T_{f}, threshold $f_{\mathrm{th}} \in (0, 1)$, probability of random sampling ϵ, max number of tests N

Initialize the set of visited states $\mathcal{D} = \varnothing$ and combination coefficients $\boldsymbol{\alpha} = [1/J, \ldots, 1/J]$;
Estimate distribution q with $\mathcal{D}_j$, $j = 1, \ldots, J$, see Section III-D2;
$\hat{f}_j \leftarrow f_j$, $j = 1, \ldots, J$;
$i \leftarrow 0$;
while $i < N$ **do**
  $i \leftarrow i + 1$;
  if `Uniform(0,1)` $< \epsilon$ **then**
    $\boldsymbol{s}_i \leftarrow$ `Sampling`(p)
  else
    $\boldsymbol{s}_i \leftarrow$ `Sampling`(p);
    while $\sum_{j=1}^{J} \alpha_j f_j(\boldsymbol{s}_i) \leqslant f_{\mathrm{th}}$ **do**
      $\boldsymbol{s}_i \leftarrow$ `Sampling`(p);
    end
  end
  $Y_i \leftarrow$ `Testing`$(\boldsymbol{s}_i, T)$;
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{s}_i, Y_i)\}$;
  if $i \% T_{\mathrm{f}} == 0$ **then**
    // Train $\hat{f}_j$ with importance sampling on $\mathcal{D}$ for T_{e} epochs, see Section III-D2
    $e \leftarrow 0$;
    while $e < T_{\mathrm{e}}$ **do**
      $\hat{f}_j \leftarrow$ `Train_with_IS`$(\hat{f}_j, \mathcal{D})$, $j = 1, \ldots, J$;
      $e \leftarrow e + 1$;
    end
    $\hat{f} \leftarrow \sum_{j=1}^{J} \alpha_j \hat{f}_j$;
    // Define $f_{\boldsymbol{\alpha}} := \sum_{j=1}^{J} \alpha_j f_j$, and $J_{\boldsymbol{\alpha}} := \frac{1}{2}\sum_{\boldsymbol{s}\in\mathcal{D}}[\hat{f}(\boldsymbol{s}) - f_{\boldsymbol{\alpha}}(\boldsymbol{s})]^2$, see Section III-D1
    $\boldsymbol{\alpha} \leftarrow \arg\min_{\boldsymbol{\alpha}:\mathbf{1}^\top\boldsymbol{\alpha}=1,\boldsymbol{\alpha}\geqslant\mathbf{0}} J_{\boldsymbol{\alpha}}$
  end
end

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

TABLE II
THE EXPERIMENTAL DETAILS OF DIFFERENT ENVIRONMENTS

Environment	d	Parameters				Training dataset (3:1:1)	Test dataset	SAM
		T	T_e	T_f	Termination			
PredatorPrey	60	50	2	5000	1.5×10^5	10^6	2×10^5	Different version
CoopNavi	60	50						Different version
BipedalWalker	10	300				5×10^5		Different algorithm

TABLE III
SAMS ACROSS DIFFERENT ENVIRONMENTS

Environment	SAM-1	SAM-2 (S1)	SAM-3	SAM-2 (S2)
PredatorPrey	MADDPG (80k episodes)	MADDPG (90k episodes)	MADDPG (100k episodes)	MADDPG (70k episodes)
CoopNavi	MADDPG (240k episodes)	MADDPG (270k episodes)	MADDPG (300k episodes)	MADDPG (260k episodes)
BipedalWalker	TQC	PPO	A2C	SAC

and online optimization with importance sampling, the overall procedures of IRTest for data-limited regimes are summarized in Algorithm 2.

IV. RESULTS

A. Experimental Setup

1) Target Environments: We evaluate IRTest in three well-established environments, as illustrated in Fig. 3.

PredatorPrey: The PredatorPrey environment is a multi-agent game within the Multi-Agent Particle Environment (MPE) developed by OpenAI [57]. In this setting, multiple predators must cooperate to capture prey on a map containing obstacles. Failure to capture the prey within a predefined time limit is considered a failure. In our setup, three predators are controlled by the target algorithm, and we introduce one prey and two obstacles. This configuration yields a 12-dimensional initial state, encompassing the positions of all six entities. To better capture the dynamic interactions between the target algorithm and the environment, we consider the first five frames of simulation collectively, concatenating them into a 60-dimensional testing state.

CoopNavi: CoopNavi is another multi-agent environment within the MPE [57], where multiple agents collaborate to navigate toward designated landmarks while avoiding collisions. In our setup, the state at each frame is represented by a 12-dimensional vector, capturing the positions of three agents and three landmarks. Similar to the PredatorPrey environment, the testing state is constructed as a 60-dimensional vector by concatenating the states from the first five frames. A failure is recorded if the agents collide with one another or fail to reach the landmarks within the maximum allowed frames.

BipedalWalker: In the BipedalWalker environment [58], a bipedal agent must navigate diverse terrains while avoiding falls, which are considered failures. In our experiments, a fixed terrain sequence consisting of pits, stumps, and grass is used, with the length of each terrain segment treated as a parameter. This configuration produces a 10-dimensional testing state.

2) Baselines: We consider three representative baselines.

Random Testing: Random testing is the most basic baseline method. It generates test cases by uniformly sampling initial states from the environment without any guidance from surrogate models.

MDPFuzz: MDPFuzz [59] is a fuzzing-based testing framework for Markov Decision Processes (MDPs). It follows an iterative pipeline consisting of seed selection, mutation, and corpus update to progressively explore the state space. As a general-purpose testing method for MDP-based systems, it provides an effective way to uncover failure-inducing cases.

SeqDivFuzz: SeqDivFuzz [60] extends MDPFuzz with a diversity-aware filtering mechanism. It trains a neural similarity model on pre-collected trajectory data to estimate sequence similarity and uses it to filter candidate test cases by comparing them with non-failure trajectories. Although it shares a similar motivation with surrogate-model-based testing, it relies only on data from the current system under test and does not involve adaptive testing or online model updates.

To reduce the influence of randomness and ensure reliable comparisons, the experiments involving both the proposed methods and the baseline methods are independently repeated with three random seeds.

3) Experiment Design: The high dimensionality and continuous nature of the testing states make exhaustive testing impractical. To address this issue, we construct a dataset by collecting a large number of testing states along with their corresponding failure labels. The dataset is then divided into training, validation, and test sets in a 3:1:1 ratio. We utilize this dataset to train a multi-layer perceptron model with the architecture MLP(d, 256, 256, 256, 256, 1) for 100 epochs as the surrogate prediction model, where d equals 60, 60, and 10 for PredatorPrey, CoopNavi, and BipedalWalker, respectively. The SPMs are trained using cross-entropy loss and the Adam optimizer. To mitigate the issue of label imbalance, we apply a class rebalancing technique, ensuring a 1:1 ratio of positive to negative samples in each batch. The failure labels in the

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
SUMMARY OF DATA USAGE IN IRTEST

Data source	Stage	Purpose
Surrogate rollouts	Training	Train SPMs
	Validation	Select the best SPM checkpoints
	Testing	Estimate importance-sampling parameters
Target-agent rollouts	Online testing	Adaptive SPM fine-tuning
	Offline evaluation	Evaluate SPM performance

TABLE V
AVERAGE PRECISION OF DIFFERENT METHODS UNDER A LIMITED NUMBER OF TESTS

Environment	SPM-1	SPM-2	SPM-3	IRTest-R (10000)	IRTest-L (10000)
PredatorPrey	0.0807	0.151	0.0738	0.104	0.141
CoopNavi	0.119	0.668	0.306	0.213	0.541
BipedalWalker	0.182	0.214	0.123	0.208	0.223

TABLE VI
RUNNING TIME ANALYSIS OF IRTEST IN DIFFERENT ENVIRONMENTS (UNIT: S)

Method	Environment	Sampling	Execution	Fine-tuning
IRTest-R	PredatorPrey	19187.86	13798.15	315.00
	CoopNavi	27864.21	11215.26	348.49
	BipedalWalker	32351.12	67230.59	539.63
IRTest-L	PredatorPrey	10352.88	12303.90	894.28
	CoopNavi	15786.42	8967.74	1172.92
	BipedalWalker	3826.41	52527.49	1065.53

training dataset for the SPM are obtained using a surrogate agent model, which is distinct from the agent under test. Experimental details are presented in Table II.

The performance of adaptive testing methods depends on the quality of the initial SPMs. When the initial model is significantly misaligned with the target system, early-stage sampling may focus on suboptimal regions. In practice, however, the initial SPM is typically constructed based on prior knowledge. For instance, different versions of the same algorithm or models trained under similar environments often share common structural properties. In this work, we initialize the SPM using previous versions of the algorithm under test or different algorithms trained in the same environment, which is intended to provide relevant initial guidance for testing the target agent. This design improves the robustness of the proposed framework during early-stage exploration. Table III summarizes the construction of the SAMs. In PredatorPrey and CoopNavi, we use different versions of the agents as SAMs, whereas in BipedalWalker, SAMs are instantiated using agents controlled by different reinforcement learning algorithms. Specifically, for the MPE-based environments, we adopt the open-source implementation of MADDPG [57] and construct SAMs by selecting model checkpoints from different stages of the same training process. For BipedalWalker, we directly utilize pre-trained policies provided by the Stable-Baselines3 repository [61], and instantiate SAMs using different reinforcement learning algorithms, including TQC [62], PPO [63], A2C [64] and SAC [65]. Based on these SAMs, the corresponding SPMs are then trained and used in subsequent testing procedures.

Data-rich regimes: In data-rich regimes, the target agent under test is the same as SAM-2, while the SPM is constructed using SAM-1. We test the target agents following Algorithm 1. The maximum fine-tuning epoch is set to $T_e = 2$, the fine-tuning frequency to $T_f = 5000$, and testing terminates once the number of tests reaches a large value (e.g., 150,000). To objectively assess the effectiveness of IRTest, we pre-collect a dataset of 200,000 testing samples with real labels obtained from the tested agent. This dataset is used to evaluate the performance of the SPM after fine-tuning. It is worth noting that this dataset is used solely for evaluating the SPMs, which is generated via random sampling and kept fixed to ensure fair and consistent comparison across different SPMs. In contrast, the evaluation of the proposed IRTest framework is conducted in a fully sequential manner, where samples are generated online at each iteration based on the current model state. Therefore, IRTest does not rely on the pre-collected dataset and remains consistent with its intended online adaptive testing paradigm.

Data-limited regimes: In this case, we employ a combination of all three SPMs and optimize their combination coefficients according to Algorithm 2 to test the target agents under limited testing resources. Among the three SPMs, the SAMs for the first and third SPMs differ from the tested agent, while the SAM for the second SPM corresponds to the current agent. We refer to this configuration as Setting 1 (S1), in which the SPM-2 is constructed from the same model as the target agent. As a result, the optimal combination coefficient is expected to be [0, 1, 0]. To further investigate the effect of surrogate-model quality, we introduce Setting 2 (S2), where the SAM used to construct SPM-2 is replaced with a different model from the target agent. Specifically, for PredatorPrey and CoopNavi, SAM-2 uses a different checkpoint from the target model, while for BipedalWalker, SAM-2 is instantiated using a different reinforcement learning algorithm, i.e., SAC [65]. As in data-rich regimes, we use the same parameters, T_e, T_f, and termination condition, and evaluate the system under test using a pre-collected test dataset. It is important to note that, in data-limited regimes, the primary focus is on the initial performance of IRTest. Setting the same maximum number of tests allows us to study the limiting performance of IRTest in this case.

For clarity, the usage of surrogate and target-agent rollouts throughout IRTest is summarized in Table IV. Surrogate rollouts are used for SPM training, checkpoint selection, and importance-sampling estimation, while target-agent rollouts

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

TABLE VII
PERFORMANCE OF IRTEST-R WITH DIFFERENT ϵ AND f_{th} ACROSS ALL THREE ENVIRONMENTS

Environment	ϵ	f_{th}	Precision at recall=0.5			AP			Precision			Recall		
			0k	25k	150k	0k	25k	150k	0k	25k	150k	0k	25k	150k
PredatorPrey	**0**	**0.5**	0.0624	0.0847	0.0676	0.0807	0.115	0.127	**0.0581**	**0.230**	**0.347**	0.555	0.135	0.0995
	0.05	**0.1**	0.0624	0.0944	0.118	0.0807	0.112	0.133	0.0337	0.0320	0.0351	**0.897**	**0.922**	0.944
	0.05	**0.3**	0.0624	**0.106**	**0.136**	0.0807	0.116	0.150	0.0443	0.0637	0.0796	0.734	0.714	0.705
	0.05	**0.5**	0.0624	0.0881	0.0941	0.0807	0.114	0.145	**0.0581**	0.186	0.315	0.555	0.216	0.144
	0.2	**0.5**	0.0624	0.0929	0.117	0.0807	0.115	**0.156**	**0.0581**	0.144	0.248	0.555	0.324	0.240
	0.5	**0.1**	0.0624	0.0942	0.113	0.0807	0.116	0.130	0.0337	0.0353	0.0325	**0.897**	0.911	**0.953**
	0.5	**0.3**	0.0624	0.0960	0.121	0.0807	0.105	0.141	0.0443	0.0613	0.0681	0.734	0.704	0.756
	0.5	**0.5**	0.0624	0.0913	0.128	0.0807	0.104	0.149	**0.0581**	0.111	0.154	0.555	0.393	0.422
	0.7	**0.5**	0.0624	0.0970	0.120	0.0807	**0.117**	0.140	**0.0581**	0.0994	0.114	0.555	0.487	0.520
CoopNavi	**0**	**0.5**	0.115	0.392	0.657	0.119	0.400	0.580	0.137	0.344	0.766	0.368	0.568	0.396
	0.05	**0.1**	0.115	0.374	0.628	0.119	0.388	0.584	0.109	0.0747	0.198	**0.533**	**0.943**	0.871
	0.05	**0.3**	0.115	0.398	**0.663**	0.119	0.410	**0.599**	0.125	0.174	0.404	0.430	0.822	0.737
	0.05	**0.5**	0.115	0.377	0.639	0.119	0.402	0.583	0.137	0.367	0.715	0.368	0.511	0.435
	0.2	**0.1**	0.115	0.393	0.623	0.119	0.401	0.583	0.109	0.0880	0.174	**0.533**	0.918	**0.881**
	0.2	**0.3**	0.115	**0.411**	0.649	0.119	**0.412**	0.594	0.125	0.169	0.338	0.430	0.826	0.777
	0.2	**0.5**	0.115	0.391	0.653	0.119	0.405	0.592	0.137	0.303	0.639	0.368	0.640	0.514
	0.2	**0.7**	0.115	0.323	0.454	0.119	0.368	0.493	**0.148**	**0.824**	**0.936**	0.297	0.0536	0.141
	0.5	**0.5**	0.115	0.396	0.636	0.119	0.401	0.587	0.137	0.265	0.469	0.368	0.705	0.671
	0.7	**0.5**	0.115	0.370	0.616	0.119	0.395	0.575	0.137	0.225	0.426	0.368	0.757	0.701
BipedalWalker	**0**	**0.5**	0.151	0.197	0.181	0.182	**0.233**	0.201	**0.172**	**0.374**	0.303	0.411	0.224	0.0447
	0.05	**0.5**	0.151	0.188	**0.222**	0.182	0.231	0.259	**0.172**	0.352	**0.451**	0.411	0.236	0.158
	0.2	**0.1**	0.151	0.198	0.213	0.182	0.225	0.229	0.138	0.102	0.110	**0.575**	**0.950**	**0.928**
	0.2	**0.3**	0.151	0.200	0.205	0.182	0.226	0.226	0.156	0.144	0.148	0.478	0.717	0.727
	0.2	**0.5**	0.151	0.195	0.209	0.182	0.225	**0.267**	**0.172**	0.282	0.353	0.411	0.315	0.270
	0.5	**0.5**	0.151	0.203	0.208	0.182	0.225	0.245	**0.172**	0.236	0.232	0.411	0.405	0.447
	0.7	**0.5**	0.151	**0.207**	0.197	0.182	0.231	0.231	**0.172**	0.194	0.187	0.411	0.535	0.531

are generated online during testing for adaptive SPM fine-tuning. The pre-collected test dataset for the target agent is used only for offline evaluation of the SPMs and is not required by IRTest in practical applications.

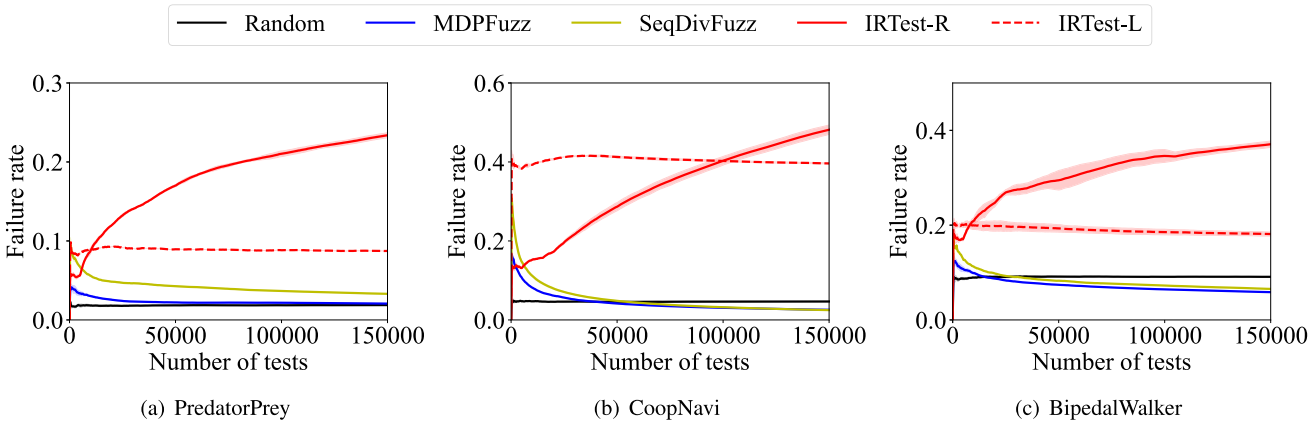
B. Testing Results in Data-Rich Regimes

In this section, we evaluate the effectiveness of IRTest in data-rich regimes (denoted as IRTest-R) by analyzing the performance of the SPM on the test dataset as the number of tests increases. The evaluation considers four key aspects: (a) precision at a recall level of 0.5, (b) average precision (AP), (c) precision, and (d) recall at the current classification threshold. Metrics (a) and (b) are used to evaluate the overall performance of the SPM on the test dataset, while metrics (c) and (d) correspond to the operating points of the SPM, capturing its real-time performance during the IRTest process. Experimental results for a random-sampling probability of $\epsilon = 0.2$ and a classification threshold of $f_{\text{th}} = 0.5$ across three environments are presented in Fig. 4.

As illustrated in Fig. 4(b), the AP of the SPM on the test dataset exhibits a steady improvement. Specifically, the AP in PredatorPrey, CoopNavi, and BipedalWalker increased from 0.0807 to 0.156, 0.119 to 0.592, and 0.182 to 0.267, corresponding to improvements of 93.3%, 397%, and 46.7%, respectively. A higher AP signifies enhanced classification performance, achieving a more favorable trade-off between the failure rate of testing states and the coverage of the testing space. These results demonstrate that by adaptively fine-tuning the SPM during testing, IRTest effectively mitigates the surrogate-to-real gap.

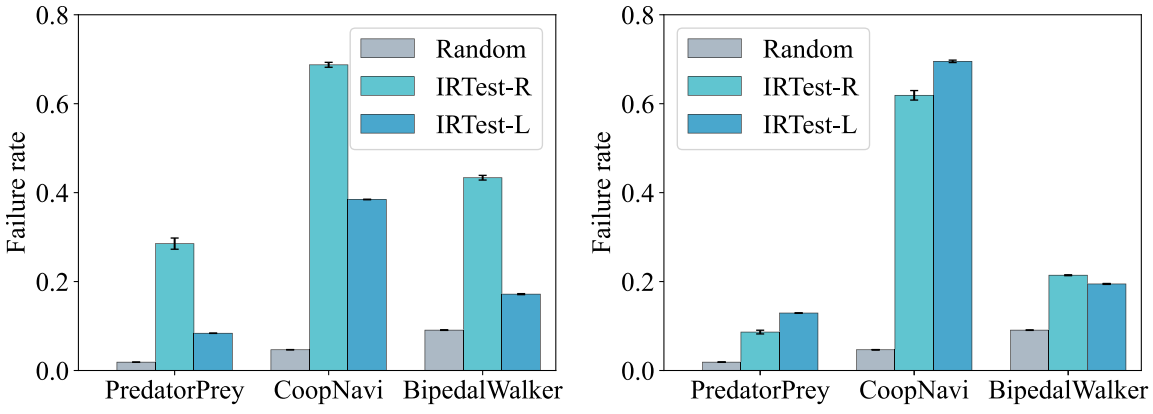
Selection of Hyper-parameters: The random sampling probability ϵ and classification threshold f_{th} are crucial parameters for optimizing performance. Results for different values of ϵ and f_{th} are provided in Table VII in Appendix. In IRTest, the random sampling probability is introduced to prevent the SPM from converging to a restricted region of the testing space, thereby promoting a more thorough exploration. As shown in Table VII, the choice of ϵ significantly affects the fine-tuning of the SPM. Specifically, in the PredatorPrey environment, when $f_{\text{th}} = 0.5$, the AP values for $\epsilon = 0, 0.05, 0.2, 0.5$ eventually reach 0.127, 0.145, 0.156, and 0.149, respectively. Meanwhile, the precision at these operating points decreases from 0.347 to 0.154 as ϵ increases, while recall exhibits an increasing trend. The experimental results suggest that an excessively small ϵ leads to overreliance on the SPM's predictions, restricting coverage to easily classified regions and thereby failing to adequately explore critical testing states. Conversely, a large ϵ diminishes the influence of the SPM, causing the sampling process to skew toward random sampling and thus reducing testing efficiency. Regarding the classification threshold f_{th}, its impact on IRTest exhibits an inverse relationship to that of ϵ. As shown in Table VII, a higher f_{th} combined with a lower ϵ leads to increased precision at the SPM operating point but at the expense of reduced recall. The optimal parameter configuration across the three

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.



(a) PredatorPrey (b) CoopNavi (c) BipedalWalker

Fig. 6. Failure rates of IRTest versus baseline methods. The shaded regions indicate one standard deviation above and below the mean across repeated runs.



(a) $f_{\text{th}} = 0.5$ (b) recall = 0.5

Fig. 7. Comparison of the failure rates of different methods at the final stage: (a) $f_{\text{th}} = 0.5$ and (b) recall = 0.5. The error bars indicate $\pm$ one standard deviation across repeated runs.

tested environments is $\epsilon = 0.2$ and $f_{\text{th}} = 0.5$, striking a balance between exploration and efficiency. Based on this analysis, we recommend selecting moderate parameter values for practical applications. While these values generally yield the best overall performance, adjustments may be necessary depending on specific objectives. For instance, if the goal is to identify more critical test cases, slightly decreasing ϵ or increasing f_{th} helps increase the proportion of failures among the generated test cases. However, it is important to note that excessively high precision settings may exclude a substantial number of test states where $f(\boldsymbol{s}) \leqslant f_{\text{th}}$, thereby increasing the number of candidate states evaluated before a test case is accepted.

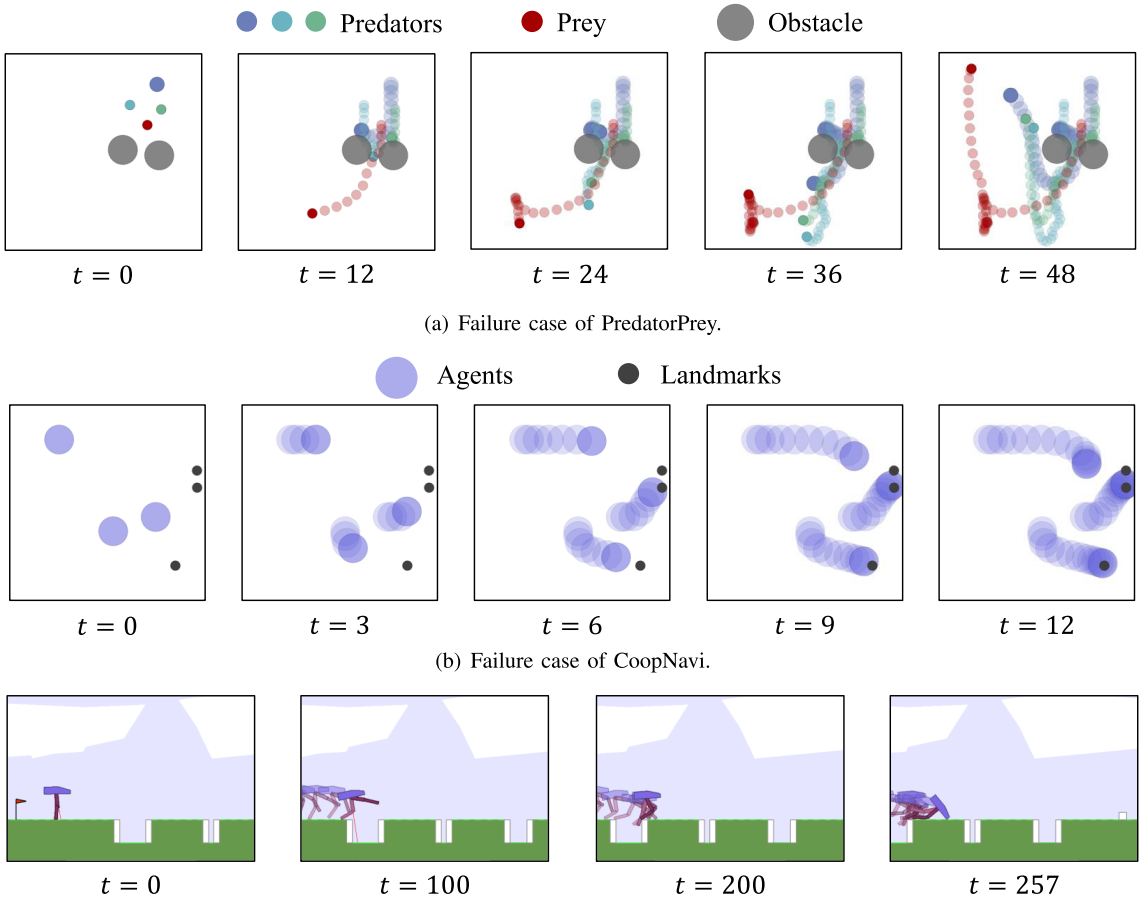
C. Testing Results in Data-Limited Regimes

In scenarios with limited testing resources, IRTest employs a mixture of SPMs as the predictor f. To evaluate its effectiveness in this setting (denoted as IRTest-L), we analyze (a) the AP of the mixture of SPMs on the test dataset and (b-d) the combination coefficients α_1, α_2, and α_3 for the three SPMs. Fig. 5 illustrates the performance of IRTest-L across the three target environments. Furthermore, to assess the impact of importance sampling, comparison experiments with and without importance sampling (in Setting 1) are conducted, denoted as "w IS" and "w/o IS", respectively.

Experimental results show that AP increases across all three environments as the number of tests grows, particularly in PredatorPrey and CoopNavi. Initially, AP rises sharply, reaching a high level quickly before stabilizing near its maximum value. This suggests that IRTest effectively reduces the surrogate-to-real gap with relatively few tests. The results without importance sampling are generally lower than those with importance sampling, as importance sampling assigns varying weights based on sampling probability, thereby addressing issues related to training on datasets with different sampling distributions.

In Setting 1, the optimized combination coefficients for IRTest with importance sampling gradually approach the expected configuration [0, 1, 0], reaching [0, 1, 10^{-5}], [0.011, 0.944, 0.045], and [0.075, 0.925, 10^{-5}], respectively. In Setting 2, where SPM-2 is constructed from a different model, the optimized coefficients adapt to different combinations across the three environments rather than converging to [0, 1, 0]. Notably, the AP consistently increases with the number of tests in both settings, demonstrating that IRTest-L can effectively optimize the combination coefficients and improve the

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.



(a) Failure case of PredatorPrey.

(b) Failure case of CoopNavi.

(c) Failure case of BipedalWalker.

Fig. 8. Examples of failure cases discovered by IRTest.

performance of the mixture of SPMs under different surrogate-model configurations. While the quality of the initial SPMs may affect the initial and final AP to some extent, IRTest-L consistently achieves performance improvement through adaptive online optimization of SPMs and combination coefficients.

To compare IRTest-L (S1) with IRTest-R, Table V presents the AP of both methods using a limited number of tests (10^4). Under resource constraints, IRTest-L achieves improvements of 35.6%, 154%, and 7.21% over IRTest-R in the three environments, respectively. Table V also reports the AP of three SPMs after offline pre-training, where SPM-2, trained with data from the target agent, demonstrates the highest AP. By performing combination coefficient regression across all three SPMs, IRTest-L rapidly approximates the AP of SPM-2 with 10^4 tests and even surpasses it in BipedalWalker by effectively integrating multiple SPMs. These results highlight that Algorithm 2, by optimizing fewer coefficients, efficiently combines multiple SPMs and prioritizes the most effective models, ultimately leading to enhanced performance.

Selection of Hyper-parameters: Comprehensive ablation studies for different ϵ values and classification thresholds f_{th} are provided in Appendix. We use $\epsilon = 0.05$ and $f_{\text{th}} = 0.5$ as a common configuration for IRTest-L across the three environments, although the best-performing configuration depends on the environment, evaluation metric, and testing budget. The results of IRTest-L (S2) show that when employing the importance-sampling-based weighting correction method, it is essential to ensure that the SPMs either include at least one well-performing candidate or can be effectively combined to achieve a superior model. In practice, the decision to use this method should consider the deviation between all SPMs and the optimal model f^*.

D. Discovery of Failure Cases

Sections IV-B and IV-C analyzed the performance of SPMs on the test dataset during IRTest. This section examines IRTest's ability to identify failure cases in data-rich and data-limited regimes. We compare random testing, MDPFuzz, and SeqDivFuzz with the proposed IRTest-R and IRTest-L, and examine how the failure rate (i.e., the proportion of discovered failure cases) evolves with the number of tests, as shown in Fig. 6. We conduct 150k test executions in total. MDPFuzz and SeqDivFuzz are implemented using their open-source code with a seed corpus of size 20k, which provides a sufficiently large set of initial seeds for these baseline methods. For IRTest, the random sampling probability and classification threshold are set to $\epsilon = 0.05$ and $f_{\text{th}} = 0.5$, respectively.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

TABLE VIII
PERFORMANCE OF IRTEST-L WITH DIFFERENT ϵ AND f_{th} ACROSS ALL THREE ENVIRONMENTS

Environment	IS	ϵ	f_{th}	AP			Precision			Recall			α_2		
				0k	25k	150k	0k	25k	150k	0k	25k	150k	0k	25k	150k
PredatorPrey	w/o	0.05	0.1	0.129	0.145	0.144	0.0327	0.0357	0.035	**0.982**	0.971	0.975	0.333	0.722	0.669
			0.3	0.129	0.14	0.14	0.0536	0.0581	0.0584	0.869	0.87	0.867	0.333	0.623	0.654
			0.5	0.129	0.136	0.137	**0.0860**	0.0919	0.0924	0.642	0.659	0.664	0.333	0.521	0.532
		0.2		0.129	0.136	0.137	**0.0860**	**0.0925**	**0.0925**	0.642	0.662	0.667	0.333	0.541	0.542
		0.5		0.129	0.145	0.146	**0.0860**	0.0923	0.0916	0.642	0.699	0.7	0.333	0.728	0.732
	w	0.05	0.1	0.129	0.144	0.140	0.0327	0.0346	0.0337	**0.982**	**0.977**	**0.980**	0.333	0.625	0.526
			0.3	0.129	0.144	0.144	0.0536	0.0585	0.0561	0.869	0.870	0.883	0.333	0.667	0.590
			0.5	0.129	**0.150**	**0.151**	**0.0860**	0.0897	0.0870	0.642	0.711	0.714	0.333	**0.804**	**1.00**
		0.2		0.129	0.147	0.150	**0.0860**	0.0907	0.0897	0.642	0.702	0.715	0.333	0.694	0.777
		0.5		0.129	0.149	0.148	**0.0860**	0.0904	0.0902	0.642	0.711	0.709	0.333	0.760	0.714
CoopNavi	w/o	0.05	0.1	0.468	0.526	0.566	0.153	0.158	0.168	**0.938**	**0.942**	**0.94**	0.333	0.481	0.602
			0.3	0.468	0.519	0.536	0.25	0.307	0.316	0.849	0.864	0.863	0.333	0.477	0.523
			0.5	0.468	0.501	0.509	**0.408**	0.434	0.439	0.623	0.659	0.667	0.333	0.435	0.447
		0.2		0.468	0.501	0.513	**0.408**	0.434	0.441	0.623	0.659	0.672	0.333	0.434	0.455
		0.5		0.468	0.511	0.537	**0.408**	0.437	**0.459**	0.623	0.667	0.715	0.333	0.443	0.502
	w	0.05	0.1	0.468	0.429	0.481	0.153	0.15	0.153	**0.938**	0.937	**0.94**	0.333	0.315	0.379
			0.3	0.468	0.542	0.598	0.25	0.316	0.344	0.849	0.863	0.856	0.333	0.521	0.723
			0.5	0.468	**0.572**	**0.658**	**0.408**	0.448	0.403	0.623	0.78	0.804	0.333	**0.583**	**0.944**
		0.2		0.468	0.548	0.615	**0.408**	**0.449**	0.415	0.623	0.701	0.799	0.333	0.489	0.805
		0.5		0.468	0.52	0.579	**0.408**	0.429	0.435	0.623	0.663	0.789	0.333	0.427	0.657
BipedalWalker	w/o	0.05	0.1	0.195	0.213	0.219	0.100	0.102	0.104	**0.971**	0.971	0.967	0.333	0.571	0.603
			0.3	0.195	0.216	**0.225**	0.139	0.136	0.138	0.793	0.831	0.817	0.333	0.561	0.620
			0.5	0.195	0.218	0.222	**0.205**	**0.208**	0.206	0.418	0.471	0.487	0.333	0.540	0.585
		0.2		0.195	0.209	0.223	**0.205**	0.202	**0.211**	0.418	0.490	0.461	0.333	0.495	0.572
		0.5		0.195	0.225	**0.225**	**0.205**	**0.208**	0.208	0.418	0.484	0.486	0.333	0.616	0.615
	w	0.05	0.1	0.195	0.211	0.213	0.100	0.102	0.102	**0.971**	**0.972**	**0.972**	0.333	0.593	0.606
			0.3	0.195	0.197	0.201	0.139	0.130	0.131	0.793	0.847	0.846	0.333	0.614	0.619
			0.5	0.195	**0.229**	0.223	**0.205**	0.197	0.176	0.418	0.532	0.608	0.333	**0.734**	**0.925**
		0.2		0.195	0.224	0.220	**0.205**	0.188	0.181	0.418	0.562	0.588	0.333	0.725	0.765
		0.5		0.195	0.214	0.217	**0.205**	0.184	0.182	0.418	0.572	0.584	0.333	0.666	0.726

As shown in Fig. 6, both IRTest-R and IRTest-L consistently achieve higher failure rates than the baselines. While MDP-Fuzz and SeqDivFuzz perform well in the early stage, their failure rates decline as the seed corpus becomes exhausted. In contrast, IRTest does not rely on a seed corpus and thus avoids this issue. Moreover, even when such degradation is not considered, IRTest still demonstrates superior long-term performance due to its adaptive testing mechanism, which continuously improves the surrogate models.

The two IRTest variants exhibit different trends. IRTest-R shows a steady increase, whereas IRTest-L remains relatively stable. In IRTest-R, the SPM is continuously fine-tuned with newly collected data that exhibit higher failure rates, leading to progressively improved precision. In contrast, IRTest-L retains the original SPMs as fixed basis models in the sampling mixture while using fine-tuned auxiliary SPMs to guide the optimization of combination coefficients. Additionally, importance sampling mitigates the impact of the new data distribution. As a result, IRTest-L relies more on the performance of the original SPMs and focuses on identifying an optimal combination, whereas IRTest-R emphasizes the iterative refinement of existing SPMs.

To further analyze this behavior, we compute the instantaneous failure rates of random testing and IRTest after 1.5×10^5 tests based on Eq. 3 and the analysis in Section III-D2. The results are shown in Fig. 7. Specifically, Fig. 7(a) and Fig. 7(b) present the failure rates at $f_{\text{th}} = 0.5$ and under a fixed recall of 0.5, respectively. The results indicate that IRTest significantly outperforms random testing. In addition, IRTest-R achieves a higher failure rate than IRTest-L at $f_{\text{th}} = 0.5$, consistent with the above analysis. Since IRTest uses a fixed classification threshold, the result at $f_{\text{th}} = 0.5$ reflects the actual final-stage performance. In contrast, the sampling mixture in IRTest-L remains restricted to combinations of the original SPMs, which may limit further improvements in predictive performance.

It is important to note that IRTest-R prioritizes improving the precision of SPMs, which may be accompanied by reduced recall at the selected operating threshold. However, as shown by the results on BipedalWalker in Fig. 7, with a sufficiently large testing budget, IRTest-R can still achieve strong overall performance. Meanwhile, the introduction of ϵ preserves support-level reachability by ensuring that all states with non-zero probability under the naturalistic distribution remain reachable during testing. Therefore, in data-rich regimes, IRTest-R improves the failure rate while preserving support-level exploration of the naturalistic testing space. In contrast, under limited budgets, recall becomes more critical. In practice, the choice of algorithm and threshold should depend on both the available test budget and the urgency of covering the test space.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

Overall, IRTest substantially outperforms baseline methods in failure discovery and effectively improves testing performance under both regimes when an appropriate classification threshold is used. Example failure cases identified by IRTest are shown in Fig. 8.

E. Computational Cost Analysis

IRTest introduces additional computational overhead mainly from the sampling and surrogate-model training procedures. During sampling, only the initial state or a small number of preceding states are evaluated when determining $f(\boldsymbol{s})$, avoiding the execution of complete test trajectories for all candidate samples. Moreover, the surrogate models are implemented using lightweight MLPs, resulting in relatively low training and inference costs.

To further quantify the computational overhead, we measured the wall-clock time of the major components of IRTest in the three environments, as reported in Table VI. The measured time is divided into sampling, test execution, and model fine-tuning. Since other processes running on the system may introduce fluctuations in the absolute running time across different experiments, we focus primarily on the relative computational cost among the components within each experiment. The results show that the sampling time is generally on the same order of magnitude as the necessary test-execution time, while the model fine-tuning time is substantially smaller. In some cases, the sampling time is even lower than the test-execution time. Overall, these results demonstrate that IRTest introduces only a moderate computational overhead, indicating that its additional time cost is acceptable in the intended testing scenarios.

V. Conclusion and Future Work

This paper introduced intelligent resilience testing, a unified framework designed to enhance the generalizability and adaptability of testing methods for decision-making agents. By integrating an offline surrogate prediction model with online adaptive correction mechanisms, IRTest effectively reduces the surrogate-to-real gap during testing. Two complementary adaptation schemes were developed: a neural fine-tuning strategy for data-rich regimes and an importance-sampling-based weighting correction method for data-limited regimes. Furthermore, a surrogate-guided adaptive sampling strategy enables efficient identification of failure-prone scenarios in high-dimensional testing spaces. Extensive experimental results demonstrate that IRTest significantly improves failure-discovery efficiency, testing stability, and robustness across heterogeneous decision-making agents.

Despite these promising results, several limitations warrant further research. First, the performance of IRTest still depends on the initial quality of the surrogate model. Poor surrogate initialization may slow adaptation in highly complex systems. Second, while the surrogate-guided adaptive sampling strategy is effective, its scalability may diminish in extremely high-dimensional scenario spaces, motivating the development of more expressive surrogate-guided adaptive sampling distributions or dimensionality-reduction-based kernels. Third, the current framework assumes consistent test outcome definitions across systems. Extending IRTest to handle heterogeneous or multi-task performance metrics remains an important direction.

Appendix

A. Ablation Studies

Tables VII–VIII present the experimental results of IRTest-R and IRTest-L. As discussed in Section IV, a moderate random sampling probability and classification threshold are required to achieve optimal AP improvement, though adjustments may be needed depending on specific objectives.

References

[1] L. He et al., "Towards provable probabilistic safety for scalable embodied ai systems," 2025, *arXiv:2506.05171*.

[2] R. Delgado, M. Campusano, and A. Bergel, "Fuzz testing in behavior-based robotics," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 9375–9381.

[3] J. Nan, R. Zhang, G. Yin, W. Zhuang, Y. Zhang, and W. Deng, "Safe and interpretable human-like planning with transformer-based deep inverse reinforcement learning for autonomous driving," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 12134–12146, 2025.

[4] M. Hua et al., "Multi-agent reinforcement learning for connected and automated vehicles control: Recent advancements and future prospects," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 16266–16286, 2025.

[5] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?" *Transp. Res. A, Policy Pract.*, vol. 94, pp. 182–193, Dec. 2016.

[6] H. X. Liu and S. Feng, "Curse of rarity for autonomous vehicles," *Nature Commun.*, vol. 15, no. 1, pp. 1–5, Jun. 2024.

[7] S. Feng et al., "Breaking through safety performance stagnation in autonomous vehicles with dense learning," *Nature Commun.*, vol. 17, no. 1, pp. 1–13, Feb. 2026.

[8] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, and F. Diermeyer, "Survey on scenario-based safety assessment of automated vehicles," *IEEE Access*, vol. 8, pp. 87456–87477, 2020.

[9] S. Zhao et al., "Genetic algorithm-based SOTIF scenario construction for complex traffic flow," *Automot. Innov.*, vol. 6, no. 4, pp. 1–16, Nov. 2023.

[10] C. Ren, J. Xu, C. Chen, S. Zhu, Y. Ma, and X. Guan, "Digital twin enabled flight control system testing: Design, development, and implementation," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 11318–11332, 2025.

[11] Y. Li, T. Qin, J. Zou, H. Li, C.-B. Yan, and X. Guan, "A test case generation scheduling model based on running event interval characteristics," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 10216–10231, 2025.

[12] L. Li, W.-L. Huang, Y. Liu, N.-N. Zheng, and F.-Y. Wang, "Intelligence testing for autonomous vehicles: A new approach," *IEEE Trans. Intell. Vehicles*, vol. 1, no. 2, pp. 158–166, Jun. 2016.

[13] H. Sun, S. Feng, X. Yan, and H. X. Liu, "Corner case generation and analysis for safety assessment of autonomous vehicles," *Transp. Res. Rec.*, vol. 2675, no. 11, pp. 587–600, Nov. 2021.

[14] S. Li, J. Yang, H. He, Y. Zhang, J. Hu, and S. Feng, "Few-shot scenario testing for autonomous vehicles based on neighborhood coverage and similarity," 2024, *arXiv:2402.01795*.

[15] S. Feng et al., "Dense reinforcement learning for safety validation of autonomous vehicles," *Nature*, vol. 615, no. 7953, pp. 620–627, Mar. 2023.

[16] S. Feng, Y. Feng, H. Sun, Y. Zhang, and H. X. Liu, "Testing scenario library generation for connected and automated vehicles: An adaptive framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 2, pp. 1213–1222, Feb. 2022.

[17] F. Klueck, Y. Li, M. Nica, J. Tao, and F. Wotawa, "Using ontologies for test suites generation for automated and autonomous driving functions," in *Proc. IEEE Int. Symp. Softw. Rel. Eng. Workshops (ISSREW)*, Oct. 2018, pp. 118–123.

[18] Y. Li, J. Tao, and F. Wotawa, "Ontology-based test generation for automated and autonomous driving functions," *Inf. Softw. Technol.*, vol. 117, Jan. 2020, Art. no. 106200.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.

[19] J. Duan, F. Gao, and Y. He, "Test scenario generation and optimization technology for intelligent driving systems," *IEEE Intell. Transp. Syst. Mag.*, vol. 14, no. 1, pp. 115–127, Jan. 2022.

[20] P. Sun et al., "Scalability in perception for autonomous driving: Waymo open dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2020, pp. 2446–2454.

[21] X. Yan, Z. Zou, S. Feng, H. Zhu, H. Sun, and H. X. Liu, "Learning naturalistic driving environment with statistical realism," *Nature Commun.*, vol. 14, no. 1, p. 2037, Apr. 2023.

[22] H. He et al., "KnowMoformer: Knowledge-conditioned motion transformer for controllable traffic scenario simulation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2024, pp. 1–6.

[23] A. Li, S. Chen, L. Sun, N. Zheng, M. Tomizuka, and W. Zhan, "SceGene: Bio-inspired traffic scenario generation for autonomous driving testing," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14859–14874, Sep. 2022.

[24] J. Wang et al., "AdvSim: Generating safety-critical scenarios for self-driving vehicles," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2021, pp. 9909–9918.

[25] T. Menzel, G. Bagschik, and M. Maurer, "Scenarios for development, test and validation of automated vehicles," in *Proc. IEEE Intell. Vehicles Symp.* (IV), Jun. 2018, pp. 1821–1827.

[26] Y. Tian, K. Pei, S. Jana, and B. Ray, "DeepTest: Automated testing of deep-neural-network-driven autonomous cars," in *Proc. IEEE/ACM 40th Int. Conf. Softw. Eng. (ICSE)*, May 2018, pp. 303–314.

[27] D. Rempe, J. Philion, L. J. Guibas, S. Fidler, and O. Litany, "Generating useful accident-prone driving scenarios via a learned traffic prior," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jul. 2022, pp. 17305–17315.

[28] L. Li et al., "Artificial intelligence test: A case study of intelligent vehicles," *Artif. Intell. Rev.*, vol. 50, no. 3, pp. 441–465, Oct. 2018.

[29] L. Li et al., "Parallel testing of vehicle intelligence via virtual-real interaction," *Sci. Robot.*, vol. 4, no. 28, p. 4106, Mar. 2019.

[30] L. Li, N. Zheng, and F.-Y. Wang, "A theoretical foundation of intelligence testing and its application for intelligent vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 10, pp. 6297–6306, Oct. 2021.

[31] R. Bai, J. Yang, W. Gong, Y. Zhang, Q. Lu, and S. Feng, "Accurately predicting probabilities of safety-critical rare events for intelligent systems," in *Proc. IEEE 20th Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2024, pp. 3243–3249.

[32] J. Tu et al., "Physically realizable adversarial examples for LiDAR object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13716–13725.

[33] M. Abdelfattah, K. Yuan, Z. J. Wang, and R. Ward, "Towards universal physical attacks on cascaded camera-LiDAR 3D object detection models," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2021, pp. 3592–3596.

[34] J. Zhou and J. Beyerer, "Corner cases in data-driven automated driving: Definitions, properties and solutions," in *Proc. IEEE Intell. Vehicles Symp.* (IV), Jun. 2023, pp. 1–8.

[35] W. Ding, B. Chen, B. Li, K. J. Eun, and D. Zhao, "Multimodal safety-critical scenarios generation for decision-making algorithms evaluation," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1551–1558, Apr. 2021.

[36] Z. He, J. Zhang, D. Yao, Y. Zhang, and H. Pei, "Adversarial generation of safety-critical lane-change scenarios for autonomous vehicles," in *Proc. IEEE 26th Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2023, pp. 6096–6101.

[37] B. Chen, X. Chen, Q. Wu, and L. Li, "Adversarial evaluation of autonomous vehicles in lane-change scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 10333–10342, Aug. 2022.

[38] S. Khatiri, S. Panichella, and P. Tonella, "Simulation-based test case generation for unmanned aerial vehicles in the neighborhood of real flights," in *Proc. IEEE Conf. Softw. Test., Verification Validation (ICST)*, Apr. 2023, pp. 281–292.

[39] S. Feng, X. Yan, H. Sun, Y. Feng, and H. X. Liu, "Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment," *Nature Commun.*, vol. 12, no. 1, pp. 1–14, Feb. 2021.

[40] S. Feng, Y. Feng, C. Yu, Y. Zhang, and H. X. Liu, "Testing scenario library generation for connected and automated vehicles, part I: Methodology," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1573–1582, Mar. 2021.

[41] S. Feng, Y. Feng, H. Sun, S. Bao, Y. Zhang, and H. X. Liu, "Testing scenario library generation for connected and automated vehicles, part II: Case studies," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 9, pp. 5635–5647, Sep. 2021.

[42] D. Zhao et al., "Accelerated evaluation of automated vehicles safety in lane-change scenarios based on importance sampling techniques," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 3, pp. 595–607, Mar. 2017.

[43] D. Zhao, X. Huang, H. Peng, H. Lam, and D. J. LeBlanc, "Accelerated evaluation of automated vehicles in car-following maneuvers," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 733–744, Mar. 2018.

[44] X. Gong, S. Feng, and Y. Pan, "An adaptive multi-fidelity sampling framework for safety analysis of connected and automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 14393–14405, Dec. 2023.

[45] M. Koren, S. Alsaif, R. Lee, and M. J. Kochenderfer, "Adaptive stress testing for autonomous vehicles," in *Proc. IEEE Intell. Vehicles Symp.* (IV), Jul. 2018, pp. 1–7.

[46] G. E. Mullins, P. G. Stankiewicz, R. C. Hawthorne, and S. K. Gupta, "Adaptive generation of challenging scenarios for testing and evaluation of autonomous vehicles," *J. Syst. Softw.*, vol. 137, pp. 197–215, Mar. 2018.

[47] G. E. Mullins, P. G. Stankiewicz, and S. K. Gupta, "Automated generation of diverse and challenging scenarios for test and evaluation of autonomous vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 1443–1450.

[48] J. Sun, H. Zhou, H. Xi, H. Zhang, and Y. Tian, "Adaptive design of experiments for safety evaluation of automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14497–14508, Sep. 2022.

[49] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.

[50] G. Mullins et al., "Delivering test and evaluation tools for autonomous unmanned vehicles to the fleet," *Johns Hopkins APL Tech. Dig. (Appl. Phys. Lab.)*, vol. 33, pp. 279–288, Apr. 2017.

[51] J. Yang, H. He, Y. Zhang, S. Feng, and H. X. Liu, "Adaptive testing for connected and automated vehicles with sparse control variates in overtaking scenarios," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2022, pp. 2791–2797.

[52] J. Yang, H. Sun, H. He, Y. Zhang, H. X. Liu, and S. Feng, "Adaptive safety evaluation for connected and automated vehicles with sparse control variates," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 2, pp. 1761–1773, Feb. 2024.

[53] J. Yang, R. Bai, H. Ji, Y. Zhang, J. Hu, and S. Feng, "Adaptive testing environment generation for connected and automated vehicles with dense reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 26, no. 4, pp. 5135–5145, Apr. 2025.

[54] J. Yang, Z. Wang, D. Wang, Y. Zhang, Q. Lu, and S. Feng, "Adaptive safety performance testing for autonomous vehicles with adaptive importance sampling," *Transp. Res. C, Emerg. Technol.*, vol. 179, Oct. 2025, Art. no. 105256.

[55] M. De Lange et al., "A continual learning survey: Defying forgetting in classification tasks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 7, pp. 3366–3385, Jul. 2022.

[56] A. B. Owen, *Monte Carlo Theory, Methods and Examples*. Stanford, CA, USA: Stanford Univ. Press, 2013.

[57] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6382–6393.

[58] G. Brockman et al., "OpenAI gym," 2016, *arXiv:1606.01540*.

[59] Q. Pang, Y. Yuan, and S. Wang, "MDPFuzz: Testing models solving Markov decision processes," in *Proc. 31st ACM SIGSOFT Int. Symp. Softw. Test. Anal.*, Jul. 2022, pp. 378–390.

[60] K. Wang, Y. Wang, J. Wang, and Q. Wang, "Fuzzing with sequence diversity inference for sequential decision-making model testing," in *Proc. IEEE 34th Int. Symp. Softw. Rel. Eng. (ISSRE)*, Oct. 2023, pp. 706–717.

[61] A. Raffin. (2020). Rl Baselines3 Zoo. [Online]. Available: https://github.com/DLR-RM/rl-baselines3-zoo

[62] A. Kuznetsov, P. Shvechikov, A. Grishin, and D. Vetrov, "Controlling overestimation bias with truncated mixture of continuous distributional quantile critics," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5556–5566.

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

[64] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

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

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.



Jingxuan Yang received the bachelor's degree from the School of Mechanical Engineering and Automation, Harbin Institute of Technology, Shenzhen, China, in 2020, and the Ph.D. degree from the Department of Automation, Tsinghua University, Beijing, China, in 2025. He is currently a Post-Doctoral Research Fellow with the Department of Automation, Tsinghua University. His current research interests include AI safety, embodied AI safety, autonomous vehicle safety, adaptive testing, and adversarial testing.



Weichao Xu received the B.S. and M.S. degrees from the Department of Automation, Tsinghua University, Beijing, China, in 2023 and 2026, respectively. His current research interests include intelligent vehicle-infrastructure cooperative systems, autonomous driving perception, and intelligent system testing.



Yuchen Shi received the B.S. degree from Tsinghua University, Beijing, China, in 2021, where he is currently pursuing the Ph.D. degree with the Department of Automation. His research interests include autonomous driving and multi-agent reinforcement learning.



Yi Zhang (Senior Member, IEEE) received the B.S. and M.S. degrees from Tsinghua University, China, in 1986 and 1988, respectively, and the Ph.D. degree from the University of Strathclyde, U.K., in 1995. He is currently a Professor in control science and engineering with Tsinghua University. His current research interests focus on intelligent transportation systems. His active research areas include intelligent vehicle-infrastructure cooperative systems, analysis of urban transportation systems, urban road network management, traffic data fusion and dissemination, and urban traffic control and management. His research fields also cover the advanced control theory and applications, advanced detection and measurement, and systems engineering.



Shuo Feng (Member, IEEE) received the bachelor's and Ph.D. degrees from the Department of Automation, Tsinghua University, China, in 2014 and 2019, respectively. He was a Post-Doctoral Research Fellow with the Department of Civil and Environmental Engineering and also an Assistant Research Scientist with the University of Michigan Transportation Research Institute (UMTRI), University of Michigan, Ann Arbor. He is currently an Associate Professor with the Department of Automation, Tsinghua University. His research interests lie in the development and validation of safety-critical machine learning, particularly for connected and automated vehicles. He was a recipient of the Best Ph.D. Dissertation Award from the IEEE Intelligent Transportation Systems Society in 2020 and ITS Best Paper Award from INFORMS TSL Society in 2021. He is an Associate Editor of IEEE TRANSACTIONS ON INTELLIGENT VEHICLES and an Academic Editor of *Automotive Innovation*.



Huaxin Pei received the B.S. degree from the University of Electronic Science and Technology of China, China, in 2017, and the Ph.D. degree in control science and engineering from the Department of Automation, Tsinghua University, China, in 2023. He is currently the Co-Founder of Dense-AI. His current research interests include embodied AI, world model, and intelligent system testing.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 16,2026 at 02:05:00 UTC from IEEE Xplore. Restrictions apply.