Variational Adversarial Training Towards Policies with Improved Robustness

Juncheng Dong[⋆†] **Hao-Lun Hsu**[⋆†] **Qitong Gao**[†] **Vahid Tarokh**[†] **Miroslav Pajic**[†]
[⋆]Equal contribution. [†]Duke University.

Abstract

Reinforcement learning (RL), while being the benchmark for policy formulation, often struggles to deliver robust solutions across varying scenarios, leading to marked performance drops under environmental perturbations. Traditional adversarial training, based on a two-player max-min game, is known to bolster the robustness of RL agents, but it faces challenges: first, the complexity of the worst-case optimization problem may induce *over-optimism*, and second, the choice of a specific set of potential adversaries might lead to *over-pessimism* by considering implausible scenarios. In this work, we first observe that these two challenges do not balance out each other. Thus, we propose to apply variational optimization to optimize over the worst-case distribution of the adversary instead of a single worst-case adversary. Moreover, to counteract over-optimism, we train the RL agent to maximize the lower quantile of the cumulative rewards under worst-case adversary distribution. Our novel algorithm demonstrates a significant advancement over existing robust RL methods, corroborating the importance of the identified challenges and the effectiveness of our approach. To alleviate computational overhead associated with the proposed approach, we also propose a simplified version with lower computational burden and only minimal performance degradation. Extensive experiments validate that our approaches consistently yield policies with superior robustness.

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.

1 INTRODUCTION

Deep reinforcement learning (RL) has shown success toward synthesizing optimal strategies over environments with complex underlying dynamics (Arulkumaran et al., 2017; Vinyals et al., 2019; Ibarz et al., 2021; Gao et al., 2022). However, given the large parameter search space under the function approximation schema and the limited scale of exploration over the state-action space during training due to sophisticated dynamics and environmental stochasticity (Shen et al., 2020), restricted performance guarantees can be provided for the resulting policies. Thus, there are often concerns regarding the robustness of RL (Pinto et al., 2017), i.e., whether RL-derived policies can perform consistently well under unforeseeable external disturbances applied to the agent upon deployment. One framework proven to effectively enhance the robustness of RL agents is *robustness through adversarial training* (Gu et al., 2019; Kamalaruban et al., 2020; Pinto et al., 2017; Vinitsky et al., 2020; Zhang et al., 2021; Pattanaik et al., 2017). Specifically, the RL agent is assumed to share the environment with a hostile agent (adversary). Formulated as a max-min optimization problem, existing works optimize the *worst-case* performance of RL agents under a predefined set of disturbances (adversaries).

However, two significant challenges remain to be resolved by existing approaches. The first challenge is the *over-optimism* caused by the difficulty of the inner optimization problem. Without a closed-form solution, the optimal solution is approximated by a first-order method such as gradient descent that can be trapped in a local optimum with high probability, resulting in an over-optimistic estimation of the worst-case performance. The second challenge is the *over-pessimism* caused by the selection of a candidate adversary set that may include unfeasible or unlikely scenarios.

In most practical real-world scenarios, it is often challenging, arguably, to have complete knowledge of the environmental disturbances and the potential adversarial attacks. Hence, when constructing the adversary/disturbance set for worst-case optimization, most approaches only consider simple restrictions, such as the norm of the parameter or the entropy of the policy, leading to an adversary set that

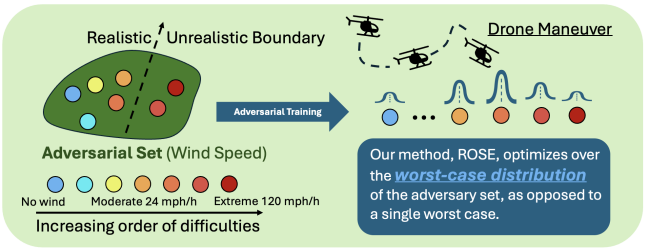


Figure 1: Optimizing over distributions of the adversary set prevents the RL agent from focusing on unlikely scenarios. For instance, when designing a drone controller, it is detrimental to overly focus on extreme scenarios, such as improbable wind conditions.

lacks precision and thus is too broad. This can result in an over-conservative agent under the scheme of worst-case optimization. For instance, when training a controller for a drone, if the adversary is allowed to modify the environmental parameters to some physically unfeasible values, the agent must sacrifice its performance in real-world scenarios to compensate for its performance in these unfeasible or highly unlikely environments so that the worst-case performance is optimized.

Notably, these two challenges do not naturally compensate for each other and thus require separate treatments. In this work, we propose optimizing over the ***worst-case distribution*** of the adversary set instead of a single worst-case adversary within the set. Our approach helps distribute the RL agent's attention across the entire disturbance/adversary set, preventing it from fixating on extreme scenarios that are unlikely to occur during real-world deployment. Specifically, we leverages a variational optimization technique, Stein Variational Policy Gradient (SVPG) (Liu et al., 2017), to identify the worst-case adversary distribution. Moreover, to counteract over-optimism, we train the RL agent to maximize the lower quantile of the cumulative rewards under the worst-case distribution. This approach ensures that the RL agent is not misled by "good scenarios" within the adversary set, where it receives high cumulative rewards, thereby preventing over-optimism. Figure 1 presents a visual illustration on the advantages of the proposed method. By addressing these problems, we show that our proposed method significantly outperforms other robust RL baselines (e.g., (Pinto et al., 2017; Vinitsky et al., 2020; Tanabe et al., 2022)), corroborating the importance of these identified problems.

However, while our method outperforms existing robust RL baselines, it employs variational optimization that incurs computational burden, which may limit its use in practice. Thus, we also introduce a simplified robust RL approach that, while significantly decreasing the computational cost, still markedly outperforms other robust RL baselines, further indicating the importance of the identified problems. Through extensive experiments on a broad range of widely used tasks with strong baselines, we demonstrate that the policies generated by our methods achieve enhanced robustness, and the improved robustness is consistent across various types of environmental disturbances.

2 PRELIMINARIES

Notation. For any finite set A, we use $|A|$ to denote its cardinality. For any positive integer m, we use $[m]$ to represent the set of integers $\{1, \dots, m\}$. For any set $\mathcal{M}$, we use $\Delta(\mathcal{M})$ to denote the set of all possible probability distributions on $\mathcal{M}$. We use $\lceil c \rceil$ to denote the smallest integer larger than c.

Adversarial Training for Robustness. Adversarial training can be formulated as a Multi-Agent Markov Decision Process (MDP) (Kaelbling et al., 1996; Busoniu et al., 2008), defined by a tuple of 6 elements $(\mathcal{S}, \mathcal{A}^p, \mathcal{A}^a, \mathcal{P}, r, \gamma, p_0)$; here, $\mathcal{S}$ is the set of states, $\mathcal{A}^p$ and $\mathcal{A}^a$ are respectively the sets of actions that the agent (protagonist) and the adversary can take, $\mathcal{P} : \mathcal{S} \times \mathcal{A}^p \times \mathcal{A}^a \to \Delta(\mathcal{S})$ is the transition function that describes the distribution of the next state given the current state and actions taken by the agent and the adversaries, $r : \mathcal{S} \times \mathcal{A}^p \times \mathcal{A}^a \to \mathbb{R}$ is the reward function for the agent (the reward function for the adversary is set to $-r$ as this work considers a zero-sum game framework), $\gamma \in [0, 1)$ is the discounting factor, and p_0 is the distribution of the initial state. We use $\pi_\theta : \mathcal{S} \to \Delta(\mathcal{A}^p)$ and $\pi_\phi : \mathcal{S} \to \Delta(\mathcal{A}^a)$ to respectively denote the polices of the agent and the adversaries, where θ and ϕ are their parameters. Let $s_t \in \mathcal{S}$ be the state of the environment at time t, $a_t^p \in \mathcal{A}^p$ (respectively $a_t^a \in \mathcal{A}^a$) the action of the agent (respectively adversary) at time t. We use

$$R(\theta, \phi) := \mathbb{E}_{s_0 \sim p_0}\big[\sum\nolimits_{t=0}^{\infty} \gamma^t r(s_t, a_t^p, a_t^a)\big] \qquad (1)$$

where $a_t^p \sim \pi_\theta(s_t), a_t^a \sim \pi_\phi(s_t)$ to represent the cumulative discounted reward that the agent π_θ can receive under the disturbance of the adversary (i.e., disturbance) following the policy π_ϕ. The objective of adversarial training (two-player max-min game) for robustness (Pinto et al., 2017; Vinitsky et al., 2020) is defined as

$$\max_{\theta \in \Theta} \min_{\phi \in \Phi} R(\theta, \phi), \qquad (2)$$

where Θ and Φ are respectively pre-defined parameter spaces for the agent and the adversary. In this approach, the RL agent π_θ optimizes the worst-case performance under disturbance.

3 VARIATIONAL ADVERSARIAL TRAINING

Instead of optimizing for a *single* worst-case adversary as in (2), we treat the adversary/disturbance as random and propose optimizing over the worst-case *distribution* of the adversary set.[1] Specifically, let $\mathcal{Q}$ be a set of probability distributions on the adversary set Φ, the objective of variational adversarial training (VAT) is defined as

$$\max_{\theta\in\Theta}\min_{q\in\mathcal{Q}}\mathbb{E}_{\phi\sim q(\phi)}[R(\theta,\phi)]. \tag{3}$$

The choice of $\mathcal{Q}$ is critical here. From a computational perspective, the inner minimization problem over $\mathcal{Q}$ should be tractable. More importantly, $\mathcal{Q}$ should mitigate the distraction caused by the unlikely scenarios/adversaries that were included in Φ either by mistake or through imprecise selection. As an example of an unsound approach, if $\mathcal{Q}$ is allowed to contain all probability distributions, the worst-case distribution is the Dirac measure on the single worst-case adversary which may represent an unlikely disturbance in practice. In this work, we choose $\mathcal{Q}$ to include continuous probability distributions centering around a prototypical distribution $q_0(\phi)$, i.e.,

$$\mathcal{Q}=\{q:\mathrm{KL}(q,q_0)\le\epsilon;\ q\text{ is smooth}\}; \tag{4}$$

here, $\mathrm{KL}(q,q_0)=\int q\log\frac{q}{q_0}$ is the Kullback–Leibler (KL) divergence (Van Erven and Harremos, 2014) between q and q_0, and ϵ is a predefined hyperparameter that controls the size of $\mathcal{Q}$. The prototypical distribution $q_0(\phi)$ can be used to capture prior knowledge of the target problem, e.g., if certain types of disturbances are known to be more likely to occur. Otherwise uninformative priors, e.g., uniform distribution, can be used.

Compared to (2), the framework in (3) offers several improvements. Firstly, the expectation over adversary distribution q distributes the attention of the RL agent across the adversary set to prevent the RL agent from focusing on unlikely/unfeasible scenarios. Secondly, optimizing over distributions aligns with real-world scenarios where adversary/disturbance is random. This better reflects environmental stochasticity (Rahimian and Mehrotra, 2019). Additionally, due to their inherent complexity, maximin optimization problems are often solved by iteratively optimizing the outer and inner problems. Compared to the single worst-case adversary, optimizing over distributions exhibit less dramatic variation, as changes in distribution are naturally smoother. This can enhance the stability of adversarial training.

[1]Note that while the term *adversary* is used, as common in the context of robust RL, we focus on robustness under environmental uncertainty/disturbances, and not in the standard security context. Thus, it is valid to assume that adversarial policies are stochastic.

Despite these improvements, VAT introduces a challenging optimization problem. We next present our algorithm *Robust Reinforcement Learning with Structured Adversarial Ensemble* (ROSE) to solve Problem (3). ROSE, whose pseudocode is detailed in Algorithm 1, is an iterative algorithm that sequentially solves the inner and outer optimization problems. In the sequel we will use superscript to denote the index of iteration number. For instance, θ^t denotes the parameter of the RL agent in the t-th iteration of the algorithm. At the beginning time step $t=0$, ROSE initializes θ^0 randomly. For $t\in\{1,\dots,T\}$, ROSE first optimizes for the optimal adversary distribution for θ^{t-1}, i.e.,

$$q^t\in\operatorname*{argmin}_{q\in\mathcal{Q}}\mathbb{E}_{q(\phi)}[R(\theta^{t-1},\phi)]. \tag{5}$$

Then ROSE solves for the optimal RL agent given the optimal adversary distribution, i.e.,

$$\theta^t\in\operatorname*{argmax}_{\theta}\mathbb{E}_{q^t(\phi)}[R(\theta,\phi)].$$

This process is repeated (see Algorithm 1) until convergence (or until the maximum iteration is reached).

Variational Inference for the Optimal Adversary Distribution. For a given θ^{t-1}, our goal is to solve the optimization problem

$$q^t\in\operatorname*{argmin}_{q\in\mathcal{Q}}\mathbb{E}_{\phi\sim q(\phi)}[R(\theta^{t-1},\phi)], \tag{6}$$

with $\mathcal{Q}=\{q:\mathrm{KL}(q,q_0)\le\epsilon\}$. By the duality of constrained optimization, Problem (6) is equivalent to

$$\operatorname*{argmin}_{q}\mathbb{E}_{\phi\sim q}[R(\theta^{t-1},\phi)]+\lambda\,\mathrm{KL}(q,q_0), \tag{7}$$

whose solution satisfy $q^t(\phi)\propto\exp\left(-\frac{R(\theta^{t-1},\phi)}{\lambda}\right)q_0(\phi)$. To solve this problem, in this work we use SVPG (Liu et al., 2017), a non-parametric variational method that approximates the target distribution. Specifically, SVPG will generate a set of samples $\{\phi_i^\star\}_{i=1}^m$ that are approximately distributed according to q^t. This makes it particularly suitable for solving Problem (3) because the generated samples can directly be used to approximate the objective of the outer problem in the iterative updates: if we have samples $\{\phi_i^t\}_{i=1}^m=\{\phi_i^\star\}_{i=1}^m$ distributed according to q^t, then

$$\theta^t\in\operatorname*{argmax}_{\theta}\mathbb{E}_{q^t(\phi)}[R(\theta,\phi)]\approx\frac{1}{m}\sum\nolimits_{i=1}^{m}R(\theta,\phi_i^t).$$

Now, consider a positive definite kernel $k:\Phi\times\Phi\to\mathbb{R}$, e.g., the Radial Basis Function (RBF) kernel

$$k(x,x')=\exp(-\tau\|x-x'\|^2), \tag{8}$$

where $\tau>0$ is a hyperparameter. SVPG iteratively transports a set of samples $\{\phi_i\}_{i=1}^m$ such that the set after transportation $\{\phi_i^\star\}$ are approximately sampled from q^t; for all

Algorithm 1 Robust Reinforcement Learning with Structured Adversarial Ensemble (ROSE)

Input: Number of samples for VAT m; Lower quantile level $\alpha \in (0, 1)$; Step size for updating the adversary policies β; Total number of iterations T; Number of transportation steps T_a.
Output: Parameter for the RL agent policy θ_T.

Randomly initialize θ and $\{\phi_i\}_{i=1}^m$
$t \leftarrow 0, \theta^t \leftarrow \theta, \phi_i^t \leftarrow \phi_i \ \ \forall i \in [m]$
for $t = 0 : T-1$ **do**
  // Variational Inference for the Worst-case Adversary Distribution
  for $0 : T_a - 1$ **do**
    for $i = 1 : m$ **do**
      Update ϕ_i^t following (9)
    end for
  end for
  // Identify the Lower α-quantile Rewards
  for $i = 1 : m$ **do**
    Estimate $R(\theta^{t-1}, \phi_i^t)$ by rolling out the agent π_{θ^t} with the adversary $\pi_{\phi_i^t}$
  end for
  Construct $\{\sigma(j)\}_{j=1}^m$ with the estimations.
  // Optimize RL agent
  Optimize the RL agent's rewards under disturbance from the worst-$\lceil \alpha m \rceil$ adversaries:

$$\theta^t \in \operatorname*{argmax}_\theta \frac{1}{\lceil \alpha m \rceil} \sum\nolimits_{j=1}^{\lceil \alpha m \rceil} R(\theta, \phi_{\sigma(j)}),$$

end for

$i \in [m]$, one step of transport is

$$\phi_i \leftarrow \phi_i - \frac{\beta}{m} \sum_{j=1}^m k(\phi_j, \phi_i) \nabla_{\phi_j} \left(\frac{R(\theta^{t-1}, \phi_j)}{\lambda} + \log q_0(\phi_j) \right) + \nabla_{\phi_j} k(\phi_j, \phi_i), \tag{9}$$

where β is a predefined step size. This process is repeated for T steps to have the final set $\{\phi_i^\star\}$. Notably, the second term $\nabla_{\phi_j} k(\phi_j, \phi_i)$ in (9) prevents the adversary policies from collapsing. This increases the effective size of the adversary policies and aligns with findings from recent works in robust RL that diversity constraints can effectively improve the result of adversarial training (e.g., (Yuan et al., 2023)).

Policy Optimization. With the set of samples $\{\phi_i^\star\}_{i=1}^m$ from q^t, the outer optimization problem can be approximated as:

$$\theta^t \in \operatorname*{argmax}_\theta \frac{1}{m} \sum\nolimits_{i=1}^m R(\theta, \phi_i^\star). \tag{10}$$

In other words, the outer optimization problem is to maximize the RL agent's cumulative rewards under the disturbances from the group of adversaries $\{\phi_i^\star\}_{i=1}^m$. This can be efficiently solved by canonical policy optimization methods such as Proximal Policy Optimization (PPO).

Combating Over-Optimism. While the framework in (3) can help mitigate over-pessimism (e.g., prevent overfitting to a single worst-case adversary), the trained RL agent may be overly optimistic about its worst-case performance because the worst-case adversary distribution in (6) cannot be solved exactly. To address this, we propose optimizing the RL agent's *cumulative rewards in the lower quantile of the worst-case adversary distribution* so that "good scenarios" (where the RL agent receives high rewards) do not lead the agent to overestimate its worst-case performance. This concept is formally known as the Conditional Value at Risk (CVaR) (Rockafellar et al., 2000).

For a random variable X, we define its Value at Risk (VaR) at level α where $\alpha > 0$ as

$$\mathrm{VaR}_X(\alpha) = \inf\{x : \mathbb{P}(X \le x) \ge \alpha\}$$

and its Conditional Value at Risk (CVaR) as $\mathrm{CVaR}_X(\alpha) = \mathbb{E}[X | X \le \mathrm{VaR}_X(\alpha)]$.

Consider the random variable $R(\theta, \phi)$ whose stochasticity originates from the random variable $\phi \sim q^\star$ where $q^\star$ is the worst case adversary distribution. Note that in (10) we are essentially optimizing its expectation $\mathbb{E}_\phi[R(\theta, \phi)]$ through Monte-Carlo approximation. To optimize the CVaR of $R(\theta, \phi)$, we can optimize the following Monte-Carlo estimation of $\mathrm{CVaR}_{R(\theta,\phi)}(\alpha)$:

$$\operatorname*{argmax}_\theta \frac{1}{\lceil \alpha m \rceil} \sum j = 1^{\lceil \alpha m \rceil} R(\theta, \phi_{\sigma(j)}^\star), \tag{11}$$

where $\{\sigma(j)\}_{j=1}^m$ is a permutation of $\{1, \ldots, m\}$ such that $R(\theta^{t-1}, \phi_{\sigma(1)}^\star) \le \cdots \le R(\theta^{t-1}, \phi_{\sigma(m)}^\star)$. Here instead of optimizing the *average* rewards across adversaries as in (10), we are focusing on the lower α-quantile of the rewards, i.e., the challenging scenarios represented by the worst-$\lceil \alpha m \rceil$ adversaries, to prevent over-optimism. As we show in the experimental section (Section 5), this significantly improves the performance. As α converges to 0, this approach is converging to optimizing the performance of RL agent under disturbance from the worst-case adversary in a set of dynamic adversaries. We provide demonstrative theoretical result illustrating that this approach can help resolve over-optimism. Due to space constraint, the results are presented in Appendix A. Notably, our theoretical results reveal that a family of adversaries can efficiently (in terms of the number of adversaries) solve the worst-case optimization problem in adversarial training if they are *diverse* enough. On the other hand, the second term in (9) exactly *results in the diversity of adversaries*.

In terms of training, to identify the worst-k adversaries where $k = \lceil \alpha m \rceil$, we first estimate $R(\theta, \phi_i)$ for $i \in [m]$ by rolling out the agent π_θ with the i-th adversary in the environment to have an estimation $\widehat{R}_i$ from which we construct $\{\sigma(j)\}_{j=1}^m$. For each adversary i in $\{\sigma(j)\}_{j=1}^k$, we roll out the agent π_θ with π_{ϕ_i} to collect b_p trajectories, each trajectory consisting of $\{(s_0, a_0, r_0, s_1) \times \cdots \times$

$(s_H, a_H, r_H, s_{H+1})\}$, where for $0 \le h \le H$, a_h is the action by the agent π_θ, and r_h is the reward received by the agent. We then combine all the collected trajectories into a single training dataset, resulting in a total of $k \cdot b_p$ trajectories. Finally we use Trust Region Policy Optimization (TRPO) (Schulman et al., 2015) to update the parameter of the RL agent. Note that any RL policy optimization method can be used – ablation studies in Section 5 and Appendix D investigate the effect of the RL algorithm.

4 SIMPLIFICATION: ROSE-S

The transportation steps of the adversary policies in ROSE in (9) can induce significant computational burden when the number of adversary policies is large as update of every adversary requires information about all other adversary policies. To this end, we propose a simplified approach that independently updates the adversary policies, thus supporting more parallel training.

ROSE-S, summarized in Algorithm 2, is an iterative algorithm that sequentially update the policy π_θ and an adversarial ensemble $\{\phi_i\}_{i=1}^m$ to solve

$$\max_{\theta\in\Theta} \min_{\phi_1,\dots,\phi_m\in\Phi} \frac{1}{\lceil \alpha m \rceil} \sum\nolimits_{j=1}^{\lceil \alpha m \rceil} R(\theta, \phi_{\sigma(j)}), \qquad (12)$$

where $R(\theta,\phi) = \mathbb{E}\big[\sum_{t=0}^{\infty} \gamma^t r_t | \pi_\theta, \pi_\phi\big]$ is the expected (discounted) cumulative rewards that the agent π_θ can receive under the disturbance of the adversary π_ϕ, and $\{\sigma(j)\}_{j=1}^m$ is a permutation of $\{1,\dots,m\}$ such that $R(\theta,\phi_{\sigma(1)}) \le \cdots \le R(\theta,\phi_{\sigma(m)})$.

ROSE-S first randomly initializes the agent policy and the adversarial ensemble. In each iteration, it first updates the adversary ensemble and then updates the agent policy with the updated adversaries. Specifically, in the t-th iteration, for $i \in [m]$, we collect a batch of trajectories $\rho_i^t = \{\tau_i^{t,j}\}_{j=1}^{b_a}$ where b_a is the batch size for training the adversarial ensemble. The trajectories are collected by rolling out the agent π_θ and the i-adversary in the environment. Each trajectory in ρ_i^t consists of H transition tuples $\{(s_0, a_0, -r_0, s_1) \times \cdots \times (s_H, a_H, -r_H, s_{H+1})\}$, where for $0 \le h \le H$, a_h is the action by the i-th adversary and r_h is the reward received by the agent. After collecting the trajectories for all the adversaries, we use these trajectories to estimate $R(\theta,\phi_i)$ for all $i \in [m]$, and select the worst-k adversaries (with the lowest returns) where $k = \lceil \alpha m \rceil$. Then, the k selected adversaries are updated using the corresponding trajectories, while the remaining $m-k$ adversaries stay unchanged (any RL algorithm can be used in the update). After the adversarial ensemble has been updated, we update the agent policy π_θ. The policy update of ROSE-S is identical to that of ROSE. This procedure is executed until the parameter of the agent policy θ converges or for a maximum of T iteration.

Algorithm 2 ROSE with Simplification (ROSE-S)

Input: m: size of the adversarial ensemble ; k: the number of the worst adversaries to use; λ_p: step size for updating the agent policy; λ_a: step size for updating the adversary ensemble;
Output: $\widehat{\theta}$: parameter for the agent policy.

```
Randomly initialize θ and {φ_i}_{i=1}^m
t ← 0, θ^t ← θ, φ_i^t ← φ_i  ∀i ∈ [m]
for t = 0 : T − 1 do
  //  Update the adversarial ensemble
  for i = 1 : m do
    Estimate R(θ^t, φ_i^t) by rolling out the agent π_{θ^t} with the adversary π_{φ_i^t}
  end for
  Construct permutation {σ(j)}_{j=1}^m with the estimations R(θ^t, φ_{σ(j)}^t).
  for j = 1 : ⌈αm⌉ do
    φ_j^{t+1} ← φ_j^t − λ_a ∇_φ R(θ^t, φ_{σ(j)}^t)
  end for
  //  Update the agent policy
  for i = 1 : m do
    Estimate R(θ^t, φ_i^{t+1}) by rolling out the agent π_{θ^t} with the adversary π_{φ_i^{t+1}}
  end for
  Construct permutation {σ(j)}_{j=1}^m with the estimations R(θ^t, φ_{σ(j)}^{t+1}).
  for j = 1 : ⌈αm⌉ do
    θ^{t+1} ← θ^t − λ_p ∇_φ R(θ^t, φ_{σ(j)}^{t+1})
  end for
end for
θ̂ ← θ^T
```

5 EXPERIMENTS

We evaluate ROSE and ROSE-S with the following baselines: (*i*) RL agents trained without adversarial training, (*ii*) RARL (Robust Adversarial Reinforcement Learning): RL agent trained against a single adversary in a zero-sum game (Pinto et al., 2017), (*iii*) RAP (Robustness via Adversary Populations): agent trained with a uniform sampling from a population of adversaries (Vinitsky et al., 2020), and (*iv*) M2TD3 (Tanabe et al., 2022): a state-of-the-art method for robust RL that, unlike all the baselines with adversarial training, requires information about the uncertainty set of the environment. We note that, despite the additional information, our proposed methods still outperform M2TD3 in most scenarios with adversarial attacks (see Table 1), further corroborating the importance of the identified problems and the value of our proposed methods. The implementation of our algorithm is available at https://github.com/hlhsu/ROSE/.

We investigate two types of robustness: (*a*) robustness to disturbance on the agent (e.g., action noise and adversarial policies) and (*b*) robustness to environmental change (e.g., mass and friction). For fairness and consistency of the performance against RARL Pinto et al. (2017), we use TRPO to update policies for all baselines as well as the two versions of our proposed methods. Our adversarial setting

follows (Pinto et al., 2017), where the adversary learns to destabilize the protagonist by applying forces on specific points, which is denoted by red arrows in Figure 9. The details of the experiments can be found in Appendix D.

Robustness to Agent Disturbance. To investigate robustness to action disturbance, we conduct experiments on the Ant, InvertedPendulum, Hopper, Half-Cheetah, and Walker2d continuous control tasks in MuJoCo environments. We measured robustness by reporting the normalized return of the learned policies under three types of disturbances: *(i)* no disturbance, *(ii)* random noise added to actions, and *(iii)* disturbance from the worst-case adversary. For the worst-case scenario, we trained an adversary to minimize the reward of each policy (both baselines and our proposed methods, ROSE and ROSE-S) while keeping the policy parameters constant, using 10 random seeds.

The results show that learning with adversaries improves performance compared to the baseline in 1^{st} column in Table 1, even without changes in training and testing conditions, consistent with findings by Pinto et al. (2017). We also emphasize that ROSE-S outperforms RAP under disturbance because simply averaging over all the adversaries as RAP, does not give the worst-case performance enough attention, thus, leading to reduced robustness.We observe that M2TD3 training with an uncertainty parameter set is relatively competitive in the environment without disturbance while our ROSE-based methods demonstrate their strength in robustness to the action noise and learned adversarial policy. In particular, ROSE consistently achieves the highest return under disturbance from adversary policies *in all tasks*.

Intuitively, by comparing the adversary update rule of ROSE-S with that of ROSE, ROSE-S can be viewed as a version of ROSE that utilizes a $0/1$ kernel. Thus, ROSE-S still maintains a worst-case adversarial distribution, although in a much noisier manner. This interpretation highlights the advantage of ROSE over ROSE-S: by offering a more accurate estimation of the adversarial distribution, ROSE demonstrates superior performance under adversarial disturbances, as demonstrated in Table 1.

Robustness to Environmental Change. Robustness should also encompass different internal conditions, such as variations in mass and friction, which are critical for locomotion tasks in MuJoCo. During training, all policies were trained with specific mass and friction values. To evaluate their robustness and generalization, we tested the policies in environments with varying mass and friction coefficients. As shown in Figure 2, our methods, particularly in the Hopper and Half-Cheetah tasks, outperforms other baselines under these conditions. Notably, ROSE-S has exhibited symmetric performance in the Hopper task (1^{st} row (a)-(e) in Figure 2) with respect to the increase or decrease of the coefficients centered at 1.0, which are the training coefficients, which are the training coefficient. In contrast, the performance of other baselines do not display this characteristic trend. Moreover, when tested in environments that gradually shift away from the training environments, the performance drop of ROSE-S and ROSE are less rapid compared to other baselines. This demonstrates the stability and predictability of our methods. In addition, ROSE displays a better performance generally with all red-ish colors in Walker2d. Note that we omit the evaluation of Ant and InvertedPendulum because the policy for the Ant task is not significantly impacted by a change mass or friction and the InvertedPendulum task is easy with similar performance among all methods.

Ablation Studies. We provide ablation studies to better understand: (A1) computational cost between ROSE and ROSE-S; (A2) stability and convergence with increasing number of adversaries; (A3) the gain of addressing the potential over-pessimism; (A4) the effect of the total number of adversaries; (A5) the effect of value of k in the worst-k set; (A6) the update frequencies of all adversaries in ROSE-S; (A7) the effect of the RL algorithm that implements our methods. Since we notice that the computational cost of ROSE is higher from (A1) and the stability and convergence for both ROSE and ROSE-S are similar from (A2), we conduct ablation studies with ROSE-S.

A1. To validate that ROSE-S can effectively reduce the computational cost of ROSE, we mainly compare the computational cost of all approaches focusing on the parameter update process. We average the computational cost for the parameters update (θ, ϕ_i) within 10 times, which is shown in Figure 3. We emphasize that the additional computational cost between RAP and ROSE-S comes from the estimations of the group of adversaries because RAP only samples adversaries uniformly. Notably, ROSE takes an extra 35.5% computational cost per update against ROSE-S, which is a heavy computational burden.

A2. The number of iterations to converge and variance have no discernible difference with increasing numbers of adversaries in Figure 6 in Appendix C. This shows that although our proposed method is more complicated and demonstrates stronger performance, it does not incur extra difficulty in training.

A3. To understand the benefits of addressing over-pessimism, we investigate a variation of ROSE-S (referred to as *ROSE-all*): instead of updating the worst-k adversaries, we update all the adversarial policies. To validate our analysis in Section 3, we conduct experiments in Hopper environment and cross-validate the robustness. Empirical evidence demonstrates that worst-k update performs better. Due to space limitation, please refer to Appendix B.1 and B.2 for details.

A4/A5. We vary the size the adversarial ensemble and the value of k in the worst-k set. As can be seen from Table 3

Table 1: Performance of ROSE/ROSE-S and baselines under various disturbances using TRPO – the highest normalized return is in bold and the second highest is underlined (mean± standard deviation).

Method	Baseline (0 adv)	RARL (1 adv)	RAP	M2TD3 (extra info)	ROSE-S (ours)	ROSE (ours)
Ant (No disturbance)	0.77±0.16	0.81±0.12	0.83±0.08	0.84±0.22	**0.87±0.13**	0.84±0.14
Ant (Action noise)	0.66±0.19	0.67±0.16	0.67±0.09	0.66±0.16	**0.70±0.14**	0.69±0.15
Ant (Adversary)	0.21±0.18	0.25±0.17	0.30±0.14	0.29±0.11	0.38±0.16	**0.44±0.23**
InvertedPendulum (No disturbance)	**1.00±0**	0.96±0.11	0.99±0.04	**1.00±0**	0.99±0.03	0.99±0.08
InvertedPendulum (Action noise)	0.91±0.13	0.91±0.15	0.95±0.10	**0.97±0.16**	0.96±0.13	0.96±0.11
InvertedPendulum (Adversary)	0.86±0.16	0.88±0.18	0.90±0.19	0.90±0.21	0.92±0.12	**0.94±0.15**
Hopper (No disturbance)	0.78±0.003	0.79±0.02	0.84±0	0.97±0.11	0.95±0.01	**0.98±0.07**
Hopper(Action noise)	0.71±0.001	0.74±0.004	0.80±0	0.77±0.07	**0.91±0.006**	0.87±0.01
Hopper (Adversary)	0.42±0.03	0.54±0.04	0.70±0.007	0.83±0.25	0.84±0.14	**0.85±0.09**
Half-Cheetah (No disturbance)	0.77±0.05	0.72±0.03	0.76±0.02	0.81±0.06	**0.87±0.05**	0.82±0.08
Half-Cheetah(Action noise)	0.59±0.2	**0.76±0.04**	0.67±0.1	0.68±0.13	**0.76±0.16**	0.73±0.13
Half-Cheetah (Adversary)	0.16±0.1	0.19±0.05	0.24±0.36	0.50±0.10	0.52±0.21	**0.58±0.30**
Walker2d (No disturbance)	0.85±0.27	0.84±0.43	0.43±0.02	**0.88±0.31**	0.84±0.44	0.86±0.38
Walker2d (Action noise)	0.78±0.31	0.80±0.28	0.36±0.04	0.79±0.21	0.83±0.37	**0.84±0.23**
Walker2d (Adversary)	0.36±0.26	0.34±0.12	0.34±0.22	0.21±0.43	0.68±0.23	**0.70±0.17**

in Section D, when the value of k increases, we are approaching RAP and focusing less on worst-case optimization. When the value of k decreases too much, the performance also decreases. This aligns with our conjecture that a single adversary can get trapped into extreme cases, also leading to degraded performance.

A6. It is theoretically possible that the worst adversaries stay worst and thus untrained. However, we find in practice if initialized differently, the worst-k adversaries keep changing and all adversaries are updated frequently. We conduct an experiment to verify this in Figure 8 in Section D. The updates are distributed evenly across adversaries, demonstrating that the worst-k adversaries keep changing.

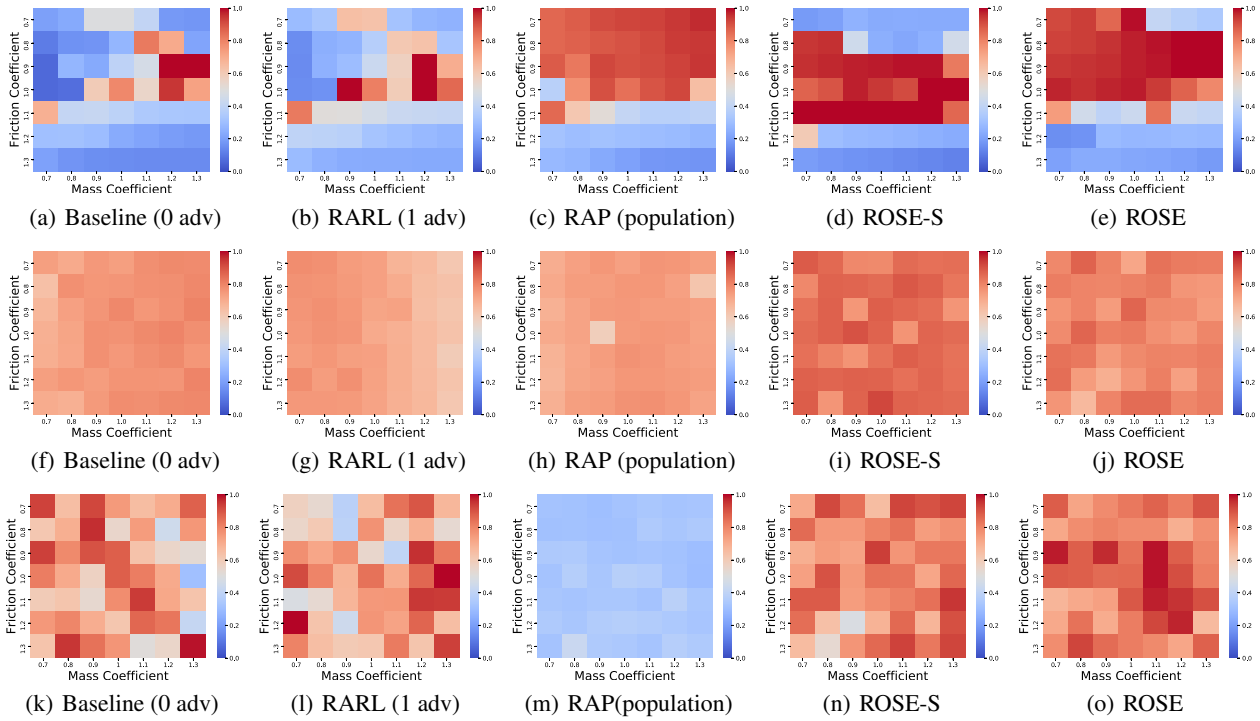
A7. To ensure that the superior performance of ROSE-based methods is consistent, we conduct additional experiments where all the baselines and ROSE-S are implemented with Proximal Policy Optimization (PPO) (Schulman et al., 2017a) and Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2019). Notably, DDPG is an off-policy RL algorithm in contrast to the on-policy TRPO and PPO. Due to space constraints, Appendix C provides details – it is shown that ROSE-S maintains its strong performance across various RL algorithms for implementation.

6 RELATED WORKS

Recent deep RL advancements, over TD learning (Kostrikov et al., 2021; Kumar et al., 2020), actor-critic (Haarnoja et al., 2018; Lee et al., 2020), model-based (Hafner et al., 2019; Kaiser et al., 2019) and RvS (Chen et al., 2021; Emmons et al., 2021) methods, have significantly impacted how autonomous agents can facilitate efficient decision making in real-world applications (e.g., healthcare (Gao et al., 2022; Tang and Wiens, 2021), robotics (Ibarz et al., 2021; Kalashnikov et al., 2018), natural language processing (Ziegler et al., 2019)). However, the large parameter search space and sample efficiency leave the robustness of RL policies unjustified. Thus, robust RL has received significant attention (Moos et al., 2022).

One research topic closely related to our method is domain randomization, which is a technique to increase the generalization capability over a set of pre-defined parameterized environments to allow the agent to learn the deviations between training and testing scenarios The environment parameters are commonly uniformly sampled during training (Tobin et al., 2017; Peng et al., 2018; Siekmann et al., 2021; Li et al., 2021b). Even though ADR (Mehta et al., 2020) is proposed to learn a parameter sampling strategy on top of domain randomization, all of the aforementioned methods are not learned over the worst-case scenarios. Moreover, in real-life applications, if not chosen carefully, the environment set can also lead to over-pessimism with a larger range while selecting a smaller range of the set will be over-optimistic. Hence, our proposed method can be readily extended into domain randomization by considering the environments as fixed adversaries.

Robustness to transition models has been widely investigated. It was initially studied by robust MDPs (Bagnell et al., 2001; Iyengar, 2005; Nilim and Ghaoui, 2003) through a model-based manner by assuming the uncertainty set of environmental transitions is known, which can be solved by dynamic programming. Following works address scenarios where the uncertainty set cannot be unknown but can be estimated. For example, (Panaganti and Kalathil, 2022; Shi et al., 2023) propose model-based algorithms that first estimate a base model and then solve the robust MDP; (Xu et al., 2023) proposes algorithms with polynomial guarantees for tabular cases while (Panaganti



(a) Baseline (0 adv) (b) RARL (1 adv) (c) RAP (population) (d) ROSE-S (e) ROSE

(f) Baseline (0 adv) (g) RARL (1 adv) (h) RAP (population) (i) ROSE-S (j) ROSE

(k) Baseline (0 adv) (l) RARL (1 adv) (m) RAP(population) (n) ROSE-S (o) ROSE

Figure 2: Average normalized return across 10 seeds tested via different mass coefficients on the x-axis and friction coefficients on the y-axis. High reward has red color; low reward has blue color. 1^{st} row: Hopper, 2^{nd} row: Half-Cheetah, 3^{rd}: Walker2d.

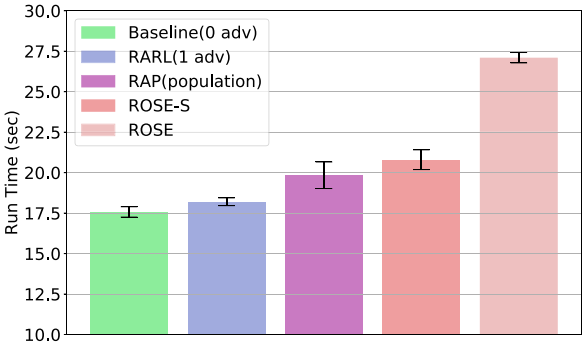


Figure 3: Computational cost comparison for parameters update.

et al., 2022; Shi and Chi, 2022) further extends the study of robust RL with only offline data. Unlike these works, we follow the approach of RARL (Pinto et al., 2017) that does not explicitly specify the set of environments but learns a robust policy by competing with an adversary. Subsequent works generalize the objective to unknown uncertainty sets and formulate the uncertainty as perturbations/disturbances in the environments (Shi et al., 2023; Abraham et al., 2020; Tanabe et al., 2022; Vinitsky et al., 2020; Pinto et al., 2017). Out of all the works about adversarial training for robustness, RAP (Vinitsky et al., 2020) shares a similar strategy as our simplified version ROSE-S introducing a group of adversaries but with distinct motivation. RAP fails to address the over-optimism as it considers the average performance against *all* adversaries. By addressing this challenge, our approach significantly outperforms RAP. While we focus on robustness to the transition model, there are two other types of robustness: robustness to the disturbance of actions (Tessler et al., 2019; Li et al., 2021a) and robustness to state/observation (Zhang et al., 2021; He et al., 2023). Some meta-RL works tackle distributional shift across tasks (Lin et al., 2020; Zahavy et al., 2021), which is orthogonal to the robustness type we consider.

7 CONCLUSION AND DISCUSSION

In this work, we have introduced ROSE, a robust RL approach that optimizes over the worst-cast distribution of the adversary set through SVPG. It addresses the two challenges, over-optimism and over-pessimism, often encountered by existing methods that leveraged adversarial training to achieve robustness. One limitation of our approach is the extra computation required by the variational inference on the adversary distribution. Thus, we have also introduced a simplified algorithm with lower computation cost and minimal performance degradation. Another computation bottleneck is the identification of the worst-k adversaries for optimization of the RL agent. However, this process can be easily distributed and paralleled as the adversaries attack independently.

Acknowledgments

This work is sponsored in part by the AFOSR under award number FA9550-19-1-0169 and by the NSF under the NAIAD Award 2332744 as well as the National AI Institute for Edge Computing Leveraging Next Generation Wireless Networks, Grant CNS-2112562.

References

Abraham, I., Handa, A., Ratliff, N., Lowrey, K., Murphey, T. D., and Fox, D. (2020). Model-based generalization under parameter uncertainty using path integral control. *IEEE Robotics and Automation Letters*, 5(2):2864–2871.

Arulkumaran, K., Deisenroth, M. P., Brundage, M., and Bharath, A. A. (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38.

Bagnell, J. A., Ng, A. Y., and Schneider, J. G. (2001). Solving uncertain markov decision processes. *Citeseer*.

Busoniu, L., Babuska, R., and De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2):156–172.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. (2021). Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. (2016). Benchmarking deep reinforcement learning for continuous control. *Proceedings of the 33rd International Conference on Machine Learning*.

Emmons, S., Eysenbach, B., Kostrikov, I., and Levine, S. (2021). Rvs: What is essential for offline rl via supervised learning? In *arXiv preprint arXiv:2112.10751*.

Gao, Q., Wang, D., Amason, J. D., Yuan, S., Tao, C., Henao, R., Hadziahmetovic, M., Carin, L., and Pajic, M. (2022). Gradient importance learning for incomplete observations. *International Conference on Learning Representations*.

Gu, Z., Jia, Z., and Choset, H. (2019). Adversary a3c for robust reinforcement learning. *arXiv preprint arXiv:1912.00330*.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. (2019). Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR.

He, S., Han, S., Su, S., Han, S., Zou, S., and Miao, F. (2023). Robust multi-agent reinforcement learning with state uncertainty. *Transactions on Machine Learning Research*.

Ibarz, J., Tan, J., Finn, C., Kalakrishnan, M., Pastor, P., and Levine, S. (2021). How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40(4-5):698–721.

Iyengar, G. N. (2005). Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280.

Kaelbling, L. P., Littman, M. L., and Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285.

Kaiser, Ł., Babaeizadeh, M., Miłos, P., Osiński, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., et al. (2019). Model based reinforcement learning for atari. In *International Conference on Learning Representations*.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. (2018). Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR.

Kamalaruban, P., Huang, Y.-T., Hsieh, Y.-P., Rolland, P., Shi, C., and Cevher, V. (2020). Robust reinforcement learning via adversarial training with langevin dynamics. *Advances in Neural Information Processing Systems*, 33:8127–8138.

Kostrikov, I., Nair, A., and Levine, S. (2021). Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. (2020). Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191.

Lee, A. X., Nagabandi, A., Abbeel, P., and Levine, S. (2020). Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *Advances in Neural Information Processing Systems*, 33:741–752.

Li, Y., Li, N., Tseng, H. E., Girard, A., Filev, D., and Kolmanovsky, I. (2021a). Safe reinforcement learning using robust action governor. In *Learning for Dynamics and Control*, pages 1093–1104. PMLR.

Li, Z., Cheng, X., Peng, X. B., Abbeel, P., Levine, S., Berseth, G., and Sreenath, K. (2021b). Reinforcement learning for robust parameterized locomotion control of bipedal robots. *2021 IEEE international conference on robotics and automation (ICRA)*, pages 2811–2817.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2019). Continuous control with deep reinforcement learning.

Lin, Z., Thomas, G., Yang, G., and Ma, T. (2020). Model-based adversarial meta-reinforcement learning. *Advances in Neural Information Processing Systems*, 33:10161–10173.

Liu, Y., Ramachandran, P., Liu, Q., and Peng, J. (2017). Stein variational policy gradient. *arXiv preprint arXiv:1704.02399*.

Mehta, B., Diaz, M., Golemo, F., Pal, C. J., and Paull, L. (2020). Active domain randomization. In *Conference on Robot Learning*, pages 1162–1176. PMLR.

Moos, J., Hansel, K., Abdulsamad, H., Stark, S., Clever, D., and Peters, J. (2022). Robust reinforcement learning: A review of foundations and recent advances. *Machine Learning and Knowledge Extraction*, 4(1):276–315.

Nilim, A. and Ghaoui, L. (2003). Robustness in markov decision problems with uncertain transition matrices. *Advances in neural information processing systems*, 16.

Panaganti, K. and Kalathil, D. (2022). Sample complexity of robust reinforcement learning with a generative model. *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, pages 9582–9602.

Panaganti, K., Xu, Z., Kalathil, D., and Ghavamzadeh, M. (2022). Robust reinforcement learning using offline data. *Advances in Neural Information Processing Systems*.

Pattanaik, A., Tang, Z., Liu, S., Bommannan, G., and Chowdhary, G. (2017). Robust deep reinforcement learning with adversarial attacks. *arXiv preprint arXiv:1712.03632*.

Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. (2018). Sim-to-real transfer of robotic control with dynamics randomization. *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810.

Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. (2017). Robust adversarial reinforcement learning. In *International Conference on Machine Learning*.

Rahimian, H. and Mehrotra, S. (2019). Distributionally robust optimization: A review. *arXiv preprint arXiv:1908.05659*.

Rockafellar, R. T., Uryasev, S., et al. (2000). Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France. PMLR.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017a). Proximal policy optimization algorithms.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017b). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shen, Q., Li, Y., Jiang, H., Wang, Z., and Zhao, T. (2020). Deep reinforcement learning with robust and smooth policy. In *International Conference on Machine Learning*, pages 8707–8718. PMLR.

Shi, L. and Chi, Y. (2022). Distributionally robust model-based offline reinforcement learning with near-optimal sample complexity. *arXiv preprint arXiv:2208.05767*.

Shi, L., Li, G., Wei, Y., Chen, Y., Geist, M., and Chi, Y. (2023). The curious price of distributional robustness in reinforcement learning with a generative model. *Advances in Neural Information Processing Systems*.

Siekmann, J., Godse, Y., Fern, A., and Hurst, J. (2021). Sim-to-real learning of all common bipedal gaits via periodic reward composition. *2021 IEEE international conference on robotics and automation (ICRA)*, pages 7309–7315.

Tanabe, T., Sato, R., Fukuchi, K., Sakuma, J., and Akimoto, Y. (2022). Max-min off-policy actor-critic method focusing on worst-case robustness to model misspecification. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K., editors, *Advances in Neural Information Processing Systems*.

Tang, S. and Wiens, J. (2021). Model selection for offline reinforcement learning: Practical considerations for healthcare settings. In *Machine Learning for Healthcare Conference*, pages 2–35. PMLR.

Tessler, C., Efroni, Y., and Mannor, S. (2019). Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning*, pages 6215–6224. PMLR.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. *Intelligent Robots and Systems*.

Van Erven, T. and Harremos, P. (2014). Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820.

Vinitsky, E., Du, Y., Parvate, K., Jang, K., Abbeel, P., and Bayen, A. (2020). Robust reinforcement learning using adversarial populations.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354.

Xu, Z., Panaganti, K., and Kalathil, D. (2023). Improved sample complexity bounds for distributionally robust reinforcement learning. In Ruiz, F., Dy, J., and van de

Meent, J.-W., editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 9728–9754. PMLR.

Yuan, L., Zhang, Z.-Q., Xue, K., Yin, H., Chen, F., Guan, C., Li, L.-H., Qian, C., and Yu, Y. (2023). Robust multi-agent coordination via evolutionary generation of auxiliary adversarial attackers.

Zahavy, T., Barreto, A., Mankowitz, D. J., Hou, S., O'Donoghue, B., Kemaev, I., and Singh, S. (2021). Discovering a set of policies for the worst case reward. *arXiv preprint arXiv:2102.04323*.

Zhang, H., Chen, H., Boning, D., and Hsieh, C.-J. (2021). Robust reinforcement learning on state observations with learned optimal adversary. *arXiv preprint arXiv:2101.08452*.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. (2019). Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] We give detailed discussion in the algorithm section.
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes] We also present ablation studies.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes] We upload our implementation in Python.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes] Right before the results.
   (b) Complete proofs of all theoretical results. [Yes] Right after the results.
   (c) Clear explanations of any assumptions. [Yes] See details in the appendix.

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes] The code of our implementation is avail- able at https://github.com/hlhsu/ROSE/
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Yes]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A THEORETICAL RESULTS

Here we illustrate the efficacy of using an ensemble to mitigate the over-optimism problem of robustness through adversarial. We will in sequence present three theoretical results. To help better appreciate the results, we first present the **insights** / take-away messages behind these results.

Insights from the Theoretical Results. From an intuitive perspective, Theorem A.3 and Theorem A.4 reveal that when the adversaries in the ensemble are distinct to each other (no matter the approach used to obtain them), the accuracy for approximating the true worst-case performance can be efficiently improved with increased number of adversaries. Lemma A.5 implies that robust optimization with an adversary ensemble solves the same optimization problem as the single adversary one. Yet a diverse set of adversaries helps better approximates the inner optimization problem, thereby relieving over-optimism.

The Detailed Results. First, consider a set of *fixed* adversaries denoted by $\widehat{\Phi} \doteq \{\phi_i\}_{i=1}^m$, where m is the total number of adversaries and for all $i \in [m],\ \phi_i \in \Phi$; further, consider the following transformed maximin optimization problem

$$\max_{\theta \in \Theta} \min_{\phi \in \widehat{\Phi}} R(\theta, \phi), \tag{13}$$

where the agent π_θ still optimizes the worst-case performance but only over a finite set of adversaries. We next prove that the proposed approach can efficiently approximate the inner optimization problem in (2).

Definition A.1 (L^∞ Norm)**.** *For a function $h : \mathcal{X} \to \mathbb{R}$, we define its L^∞ norm as $||h||_\infty = \sup_{x \in \mathcal{X}} |h(x)|$.*

Definition A.2 (ϵ-packing)**.** *Let $(\mathcal{U}, d)$ be a metric space where $d : \mathcal{U} \times \mathcal{U} \to \mathbb{R}^+$ is the metric function. Then a finite set $\mathcal{X} \subset \mathcal{U}$ is an ϵ-packing if no two distinct elements in $\mathcal{X}$ are ϵ-close to each other, i.e.,*

$$\inf_{x, x' \in \mathcal{X} : x \neq x'} d(x, x') > \epsilon.$$

Let R_Φ denote a function class defined as

$$R_\Phi \doteq \{R_\phi \doteq R(\theta, \phi) : \Theta \to \mathbb{R} | \phi \in \Phi\}.$$

Our first result illustrates that if one chooses a set of adversaries that are different enough, then the number of adversaries needed to approximate the inner optimization problem is in approximately linear order of the desired precision.

Theorem A.3. *Consider the metric space $(R_\Phi, ||\cdot||_\infty)$ where for any two functions $R_\phi, R_{\phi'} \in R_\Phi$, the distance between them is defined as $d(R_\phi, R_{\phi'}) \doteq ||R_\phi - R_{\phi'}||_\infty$. Assume that R_Φ has finite radius under this metric, i.e.,*

$$\sup_{\phi, \phi' \in \Phi} d(R_\phi, R_{\phi'}) \leq r_{\max}, \tag{14}$$

where $r_{\max} < \infty$ is a finite number. Let $\widehat{\Phi} = \{\phi_i\}_{i=1}^m \subset \Phi$. If $R_{\widehat{\Phi}}$ is a maximal ϵ-packing then $|R_{\widehat{\Phi}}| \geq \lceil \frac{r_{\max}}{\epsilon} \rceil$, where $\lceil c \rceil$ is the smallest integer that is larger than or equal to c. Moreover, for any $\theta \in \Theta$, let $\widehat{\phi} \doteq \operatorname{argmin}_{\phi \in \widehat{\Phi}} R(\theta, \phi)$ denote the approximated solution and $\phi^\star \doteq \operatorname{argmin}_{\phi \in \Phi} R(\theta, \phi)$ denote the optimal solution. Then, the approximation error of $\hat{\phi}$ in the inner optimization problem is bounded above by ϵ, i.e.,

$$|R(\theta, \phi^\star) - R(\theta, \widehat{\phi})| \leq \epsilon.$$

Proof. Since $R_{\widehat{\Phi}}$ is an ϵ-packing, balls of radius $\frac{\epsilon}{2}$ do not overlap. Consider $\mathcal{U}$ the union of the balls. Any point in $\mathcal{U}$ is clearly within distance $\frac{\epsilon}{2} < \epsilon$ from $R_{\widehat{\Phi}}$. Consider a point $\phi_* \notin \mathcal{U}$. If the ball of radius $\frac{\epsilon}{2}$ around ϕ_* is disjoint from $\mathcal{U}$, then $R_{\widehat{\Phi}} \cup \phi_*$ is an ϵ packing that strictly contains $R_{\widehat{\Phi}}$. This violates the maximality assumption on $R_{\widehat{\Phi}}$. Since $R_{\widehat{\Phi}}$ is an ϵ-packing,then balls of radius $\frac{\epsilon}{2}$ do not overlap. Consider $\mathcal{U}$ the union of the balls. Any point in $\mathcal{U}$ is clearly within distance $\frac{\epsilon}{2} < \epsilon$ from $R_{\widehat{\Phi}}$.

Consider a point $\phi_* \notin \mathcal{U}$. If the ball $B(\phi_*, \frac{\epsilon}{2})$ of radius $\frac{\epsilon}{2}$ around ϕ_* is disjoint from $\mathcal{U}$ then $R_{\widehat{\Phi}} \cup \phi_*$ is an ϵ-packing that strictly contains $R_{\widehat{\Phi}}$. This violates the maximality of $R_{\widehat{\Phi}}$. Thus $B(\phi_*, \frac{\epsilon}{2})$ has an intersection with at least a ball of radius $\frac{\epsilon}{2}$ around some point of $R_{\widehat{\Phi}}$. It follows from the triangle inequality that ϕ_* is within distance ϵ of this point. Since ϕ_* was

arbitrary, then $R_{\widehat{\Phi}}$ is an ϵ-covering and an ϵ-net. The fact that $|R_{\widehat{\Phi}}| \geq \lceil \frac{r_{\max}}{\epsilon} \rceil$ follows trivially from the fact that balls of radius $\frac{\epsilon}{2}$ around the points of $R_{\widehat{\Phi}}$ do not intersect and the triangle inequality.

Since $R_{\widehat{\Phi}}$ is an ϵ-net of R_Φ, for any $\phi^\star$ there exists $\phi \in \widehat{\Phi}$ such that $||R_\phi - R_{\phi^\star}||_\infty \leq \epsilon$. By definition of the L_∞ norm, this implies that for any $\theta \in \Theta$, $|R(\theta, \phi^\star) - R(\theta, \phi)| \leq \epsilon$. Also because $\hat{\phi} \doteq \operatorname{argmin}_{\phi \in \widehat{\Phi}} R(\theta, \phi)$, we have $R(\theta, \hat{\phi}) \leq R(\theta, \phi)$. Since $\phi^\star$ is defined as $\operatorname{argmin}_{\phi \in \Phi} R(\theta, \phi)$, it holds that

$$\begin{aligned} |R(\theta, \phi^\star) - R(\theta, \hat{\phi})| &= R(\theta, \hat{\phi}) - R(\theta, \phi^\star) \\ &\leq R(\theta, \phi) - R(\theta, \phi^\star) \\ &= |R(\theta, \phi^\star) - R(\theta, \phi)| \leq \epsilon, \end{aligned}$$

completing the proof. □

The assumption in (14) essentially requires that for any policy π_θ, its performance in two different environments cannot differ without bound. From another perspective, this is equivalent to suggesting that the adversary cannot be omnipotent. Under this assumption, if we can construct a set of adversaries that are distinct from each other, then the number of adversaries that is needed for approximation is about $O(\frac{1}{\epsilon})$, where ϵ can be interpreted as the desired level of accuracy towards the approximation. We next show that if we only want to use adversarial herding in order to approximate accurately with high probability (instead of an almost sure approximation as in Theorem A.3), then the number of required adversaries can be reduced.

Theorem A.4. *Assume that Φ is a metric space with a distance function $d : \Phi \times \Phi \mapsto \mathbb{R}$. Let σ be any probability measure on Φ. Let $\widehat{\Phi} = \{\phi_i\}_{i=1}^m$ be a set of independently sampled elements from Φ following identical measure σ. Consider a fixed $\theta \in \Theta$ and assume that $R(\theta, \phi)$ is an L_ϕ-Lipschitz continuous function of ϕ with respect to the metric space (Φ, d). Let $\widehat{\phi}$ and ϕ^* be defined the same as in Theorem A.3. For presentation simplicity, assume that $\sigma(\{\phi : d(\phi, \phi^*) \leq \epsilon\}) \geq L_\sigma \epsilon$. Let $0 < \delta < 1$. Then with probability $1 - \delta$, the approximation error of $\hat{\phi}$ on the inner optimization problem is upper bounded by ϵ if $m \geq \log(\delta)/\log(1 - \frac{L_\sigma}{L_\phi}\epsilon)$.*

Proof. Assume that we have $\widehat{\Phi} = \{\phi_i\}_{i=1}^m$ as a batch of d elements from Φ, independently sampled according to the probability measure σ. For any $c > 0$, we have

$$\begin{aligned} &\mathbb{P}(\exists \phi \in \widehat{\Phi} \quad s.t. \quad d(\phi, \phi^*) \leq c) \\ &= 1 - \mathbb{P}(\forall \phi \in \widehat{\Phi} : d(\phi, \phi^*) > c) \\ &= 1 - \mathbb{P}^m(\phi : d(\phi, \phi^*) > c) \\ &= 1 - (1 - \sigma(\{\phi : d(\phi, \phi^*) \leq c\}))^m. \end{aligned} \tag{15}$$

Additionally, if there exists $\phi \in \widehat{\Phi}$ such that $d(\phi, \phi^*) \leq c$, then by the assumption that R_ϕ is L_ϕ-Lipschitz continuous, $|R(\theta, \phi) - R(\theta, \phi^*)| \leq L_\phi \cdot c$. By definition of $\widehat{\Phi}$, it holds that

$$\begin{aligned} &|R(\theta, \widehat{\phi}) - R(\theta, \phi^*)| = R(\theta, \widehat{\phi}) - R(\theta, \phi^*) \\ &\leq R(\theta, \phi) - R(\theta, \phi^*) = |R(\theta, \phi) - R(\theta, \phi^*)| \\ &\leq L_\phi \cdot c. \end{aligned}$$

To prove the theorem, let $c = \frac{\epsilon}{L_\phi}$, and we would like to have

$$\begin{aligned} &1 - \delta \leq \mathbb{P}(\exists \phi \in \widehat{\Phi} \quad s.t. \quad d(\phi, \phi^*) \leq c) \\ &1 - \delta \leq 1 - (1 - \sigma(\{\phi : d(\phi, \phi^*) \leq c\}))^m \\ &(1 - \sigma(\{\phi : d(\phi, \phi^*) \leq c\}))^m \leq \delta \\ &m \leq \frac{\log(\delta)}{\log(1 - \sigma(\{\phi : d(\phi, \phi^*) \leq c)} \\ &m \leq \frac{\log(\delta)}{\log(1 - \frac{L_\sigma}{L_\phi}\epsilon)} \end{aligned} \tag{16} \tag{17}$$

Table 2: Performance of ROSE-S and baselines under various disturbances in **Hopper environment.**

Method	**ROSE-all**	**ROSE-worst**
Hopper (No disturbance)	0.86±0.07	**0.95±0.01**
Hopper(Action noise)	0.81±0.01	**0.91±0.006**
Hopper (Adversary)	0.63±0.22	**0.84±0.14**

where in Eq. (16) we used Eq. (15) and in (17) we used the fact that $c = \frac{\epsilon}{L_\phi}$ and the density assumption that $\sigma(\{\phi : d(\phi, \phi^*) \leq \epsilon\}) \geq L_\sigma \epsilon$. This concludes the proof. □

In Theorem A.4, we can replace L_σ with other dense conditions about measure of Φ and achieve analogous results. In contrast toTheorem A.3, if we can sample from a measure that is dense around the optimal ϕ, then the required number of adversaries can be decreased.

Specifically, if one would like to decrease of probability of bad approximation by half, the extra number of adversaries needed is about $O(\frac{1}{c})$ where c is a constant related to how dense one can sample close to the true optimal.

Now let $\phi_i \in \widehat{\Phi}$ be learners (Φ is an adversary ensemble), instead of fixed adversaries. The objective then becomes

$$\max_{\theta \in \Theta} \min_{\phi_1, \ldots, \phi_m \in \Phi} \min_{\phi \in \{\phi_i\}_{i=1}^m} R(\theta, \phi). \tag{18}$$

It is important to observe that the solution set of (18) is identical to that of the maximin problem in the original approach.

Lemma A.5. *The solution set to the optimization problem in* (2) *is identical to the solution set of the optimization problem in* (18)*. That is, for any $\theta \in \Theta$ and integer $m \geq 1$,*

$$\min_{\phi \in \Phi} R(\theta, \phi) = \min_{\phi_1, \ldots, \phi_m \in \Phi} \min_{\phi \in \{\phi_i\}_{i=1}^m} R(\theta, \phi).$$

Proof. There are only 3 possibilities regarding the order of $\min_{\phi \in \Phi} R(\theta, \phi)$ and $\min_{\phi_1, \ldots, \phi_m \in \Phi} \min_{\phi \in \{\phi_i\}_{i=1}^m} R(\theta, \phi)$:

- *(i)* $\min_{\phi \in \Phi} R(\theta, \phi) = \min_{\phi_1, \ldots, \phi_m \in \Phi} \min_{\phi \in \{\phi_i\}_{i=1}^m} R(\theta, \phi)$;
- *(ii)* $\min_{\phi \in \Phi} R(\theta, \phi) > \min_{\phi_1, \ldots, \phi_m \in \Phi} \min_{\phi \in \{\phi_i\}_{i=1}^m} R(\theta, \phi)$;
- *(iii)* $\min_{\phi \in \Phi} R(\theta, \phi) < \min_{\phi_1, \ldots, \phi_m \in \Phi} \min_{\phi \in \{\phi_i\}_{i=1}^m} R(\theta, \phi)$.

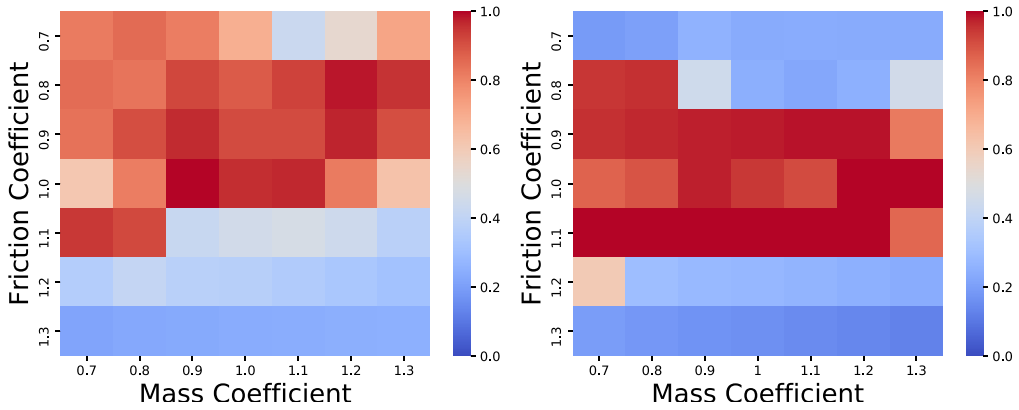
We prove by contradiction that *(ii)* and *(iii)* can not happen.

If *(ii)* holds, let $\widehat{\Phi}^*$ denote the optimal solution to the right hand side (RHS) and let $\hat{\phi} \doteq \min_{\phi \in \widehat{\Phi}^*} R(\theta, \phi)$. Because *(ii)* holds, we have $\min_{\phi \in \Phi} R(\theta, \phi) > R(\theta, \hat{\phi})$. However, this is impossible since $\hat{\phi} \in \Phi$ by definition of $\widehat{\Phi}$.

If *(iii)* holds, let $\phi^* \doteq \min_{\phi \in \Phi} R(\theta, \phi)$ be the optimal solution of the left hand side (LHS). Consider any $\widehat{\Phi}$ that includes ϕ^*, then $\min_{\phi \in \Phi} R(\theta, \phi) = R(\theta, \phi^*) \geq \min_{\phi \in \widehat{\Phi}} R(\theta, \phi) \geq \min_{\phi_1, \ldots, \phi_m \in \Phi} \min_{\phi \in \{\phi_i\}_{i=1}^m} R(\theta, \phi)$. This contradicts the fact that *(iii)* holds. Hence, the proof is complete. □

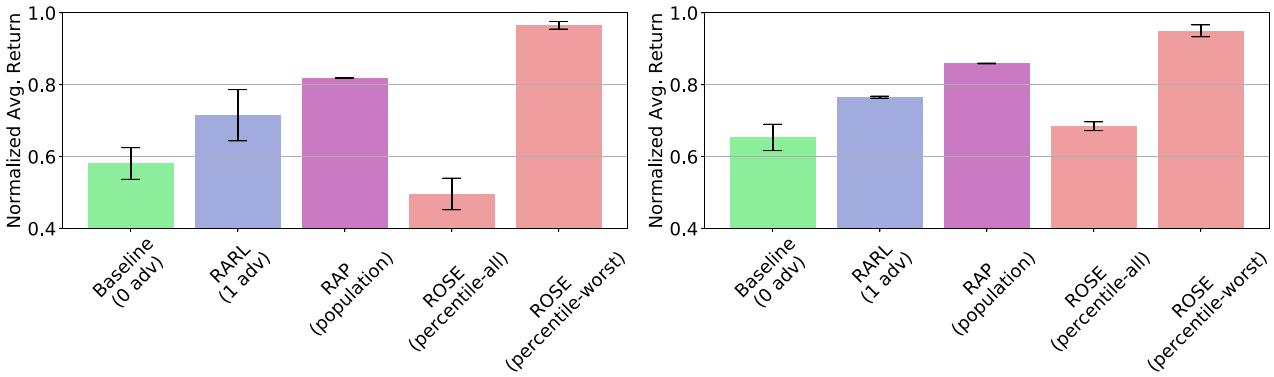
B ABLATION STUDIES FOR UNDERSTANDING THE COST OF OVER-PESSIMISM

To validate our theory in Section 3, we conduct extra experiments in the Hopper environment. We investigate two versions of ROSE that updates the adversarial head with different schemes: in each iteration during training, (*i*) ROSE-worst: only update the worst-k adversaries, where the worst-k adversaries are defined as in Section 3, and (*ii*) ROSE-all: update the whole population of adversaries.



(a) ROSE-all (b) ROSE-worst

Figure 4: Average normalized return across 10 seeds tested via different mass coefficients on the x-axis and friction coefficients on the y-axis for two variations of ROSE-S in the Hopper environment. High reward has red color; low reward has blue color.



(a) Adversarial policy from ROSE-all (b) Adversarial policy from ROSE-worst

Figure 5: Average normalized return for Hopper task cross-tested with the worst adversary from (1) ROSE-all and (2) ROSE-worst. Note that the cross-validation is based on the simplified version of ROSE.

B.1 Robustness to Disturbance on the Agent

We report the normalized return of ROSE-S with different update methods as the discussion in Table 2 for 3 types of disturbances during evaluation: (*i*) no disturbance, (*ii*) random adversary that adds noise to the actions of the agents, and (*iii*) the worst adversary that represents the worst case performance of a given policy. Empirical evidence demonstrates that ROSE-S (referred to as *ROSE-worst*) leads to more robustness to disturbance compared with ROSE-all, which supports our analysis in Section 3.

B.2 Robustness to Test Conditions (Environmental Changes)

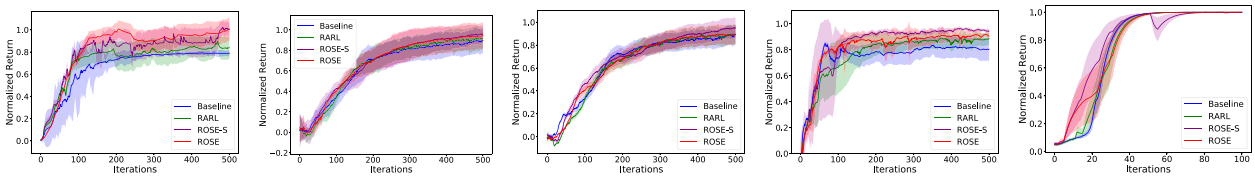
We follow the same evaluation metrics as we demonstrate in Section 5, considering training with a fixed pair of mass and friction values while evaluating the trained policies with varying mass and friction coefficients. We show that the ability to generalization is better with only updating the worst$-k$ adversaries during training in Figure 4.

B.3 Cross-Validation of ROSE-S

After the training process of ROSE-S is finished, we have access to a trained agent and a group of trained adversarial policies. To evaluate the effectiveness of training, we evaluate all the baseline methods and ROSE-worst/all under the disturbance from two adversaries: *(i)* the worst adversary in the trained adversarial ensemble of ROSE-worst and *(ii)* the

Table 3: Ablation Studies of Number of k in Half-Cheetah Environment (mean$\pm$ standard deviation).

Number of Adversaries N	5			10		
worst k with percentage of N	10%	30%	50%	10%	30%	50%
No disturbance	0.73$\pm$0.02	0.73$\pm$0.04	0.70$\pm$0.03	0.74$\pm$0.05	0.87$\pm$0.05	0.84$\pm$0.04
Action Noise	0.63$\pm$0.22	0.68$\pm$0.23	0.61$\pm$0.18	0.65$\pm$0.18	0.76$\pm$0.16	0.74$\pm$0.15
Worst Adversary	0.23$\pm$0.11	0.21$\pm$0.15	0.18$\pm$0.09	0.36$\pm$0.15	0.52$\pm$0.21	0.43$\pm$0.26



(a) Hopper (b) Walker2d (c) Half-Cheetah (d) Ant (e) Pendulum

Figure 6: Learning curve of robust RL methods via adversarial training with TRPO.

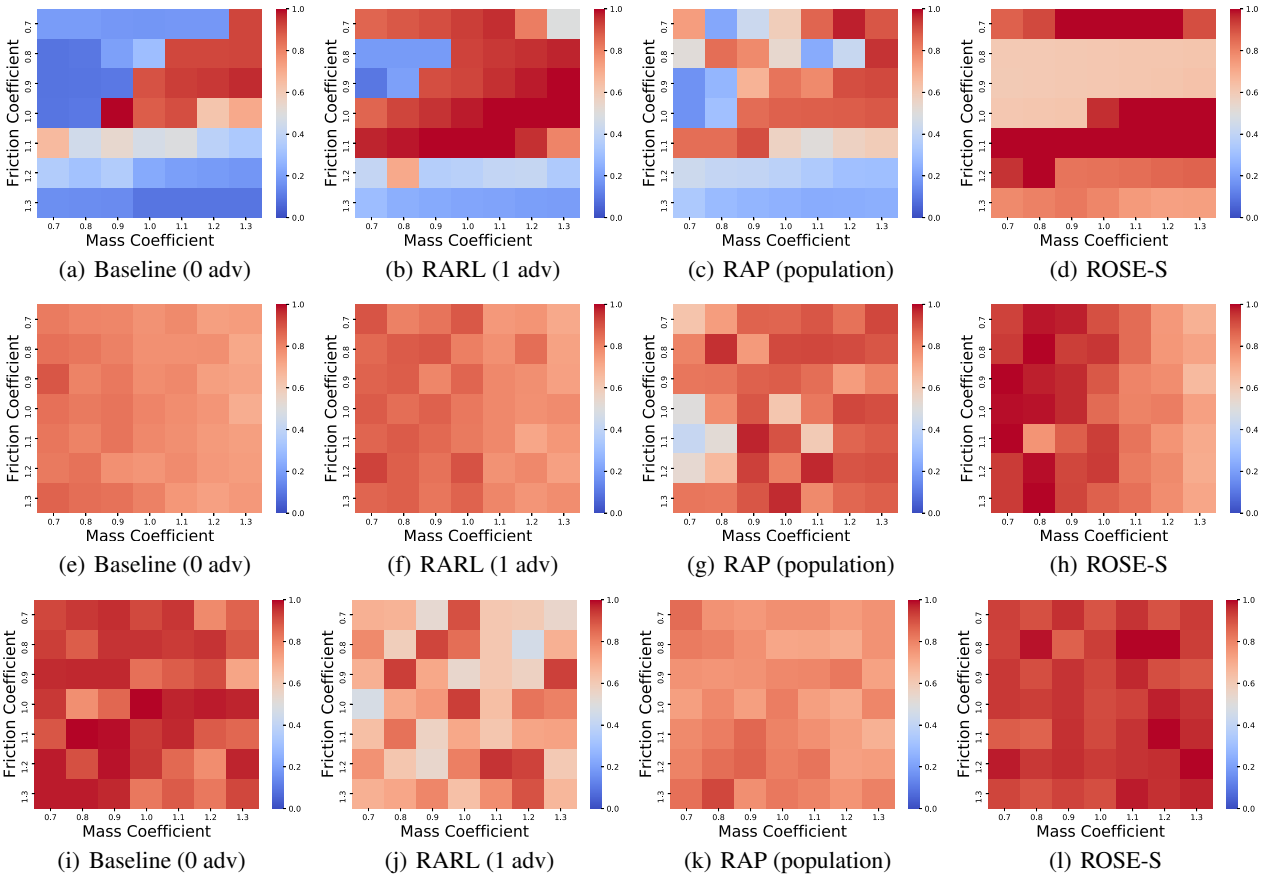
Table 4: Performance of ROSE and baselines under various disturbances for **PPO** (mean$\pm$ standard deviation).

Method	**Baseline (0 adv)**	**RARL (1 adv)**	**RAP (population adv)**	**ROSE**
Hopper (No disturbance)	0.89$\pm$0.009	**0.97$\pm$0.003**	0.87$\pm$0.003	**0.97$\pm$0.33**
Hopper(Action noise)	0.72$\pm$0.07	**0.94$\pm$0.002**	0.53$\pm$0.2	0.88$\pm$0.001
Hopper (Worst Adversary)	0.65$\pm$0.04	**0.88$\pm$0.009**	0.68$\pm$0.17	0.87$\pm$0.24
Half-Cheetah (No disturbance)	0.89$\pm$0.04	0.91$\pm$0.04	0.89$\pm$0.08	**0.92$\pm$0.08**
Half-Cheetah(Action noise)	**0.91$\pm$0.03**	0.89$\pm$0.10	0.53$\pm$0.43	**0.91$\pm$0.03**
Half-Cheetah (Worst Adversary)	0.21$\pm$0.24	0.24$\pm$0.04	0.28$\pm$0.39	**0.51$\pm$0.43**
Walker2d (No disturbance)	0.94$\pm$0.32	0.91$\pm$0.33	0.90$\pm$0.10	**0.98$\pm$0.09**
Walker2d (Action noise)	0.93$\pm$0.29	0.86$\pm$0.35	**0.99$\pm$0.14**	0.98$\pm$0.03
Walker2d (Worst Adversary)	0.30$\pm$0.13	0.51$\pm$0.16	0.53$\pm$0.24	**0.71$\pm$0.37**

worst adversary in the trained adversarial ensemble of ROSE-all. The selection of the worst adversary follows the same process as described in Section 5. As can be seen in Figure 5, ROSE-all cannot survive from its own adversary, i.e., the adversary that it has encountered during training.

C ABLATION STUDIES ON THE RL ALGORITHM IMPLEMENTING ROSE-BASED METHODS

In Section 5, we adopt TRPO as our core baseline and consider different adversarial algorithms built on top of TRPO, where the learning curves of all robust RL approaches via adversarial training are shown in Figure 6. Both the number of iterations required to converge and the variance remained consistent, even with an increasing number of adversaries. This demonstrates that, although our method is more complex and delivers stronger performance, it does not introduce additional training difficulties. Here we mainly conduct the experiments on the Hopper, Half-Cheetah, and Walker2d tasks using Proximal Policy Optimization (PPO) (Schulman et al., 2017b) and show the robustness comparison with varying test conditions in Figure 7 and with various disturbances in Table 4. We also extend our method (ROSE-S) to an off-policy version using DDPG (Lillicrap et al., 2019), demonstrating better performance consistently in Table 5. Our ROSE performs better against other baselines using PPO and DDPG, indicating that our approach is not limited to a specific RL policy optimization method.



(a) Baseline (0 adv) (b) RARL (1 adv) (c) RAP (population) (d) ROSE-S

(e) Baseline (0 adv) (f) RARL (1 adv) (g) RAP (population) (h) ROSE-S

(i) Baseline (0 adv) (j) RARL (1 adv) (k) RAP (population) (l) ROSE-S

Figure 7: Average normalized return across 10 seeds tested via different mass coefficients for **PPO** on the x-axis and friction coefficients on the y-axis. High reward has red color; low reward has blue color. 1^{st} row: Hopper, 2^{nd} row: Half-Cheetah, 3^{rd}: Walker2d.

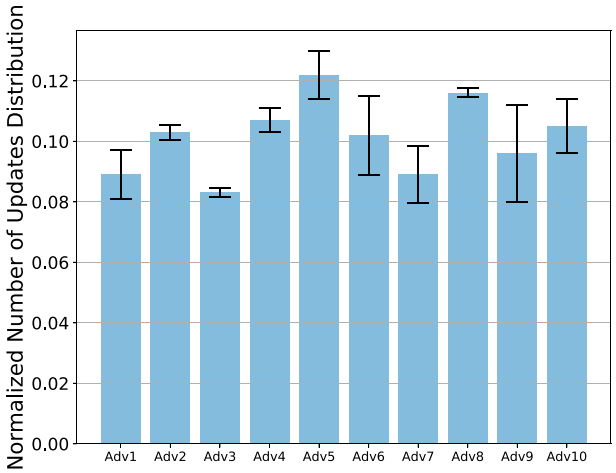


Figure 8: Number of updates for the adversaries in Ant environment with N=10 and k=3.

D EXPERIMENTAL DETAILS

We conduct experiments to evaluate the robustness of our proposed methods (ROSE and ROSE-S) on several MuJoCo environments. All of our experiments were run on Nvidia RTX A5000 with 24GB RAM. Our implementations are partly based on the codes published by *rllab* (Duan et al., 2016). Since the core contribution of this work is to introduce structured

Table 5: Performance of ROSE and baselines under various disturbances using **DDPG** with Ant environments (mean± standard deviation).

Method	**Baseline (0 adv)**	**RARL (1 adv)**	**RAP (population adv)**	**ROSE-S**
Ant (No disturbance)	0.80±0.12	0.84±0.06	0.86±0.09	**0.89±0.10**
Ant (Action noise)	0.58±0.19	0.61±0.18	**0.63±0.12**	**0.63±0.15**
Ant (Worst Adversary)	0.20±0.07	0.26±0.09	0.28±0.15	**0.35±0.14**

Table 6: The swept hyper-parameters for TRPO

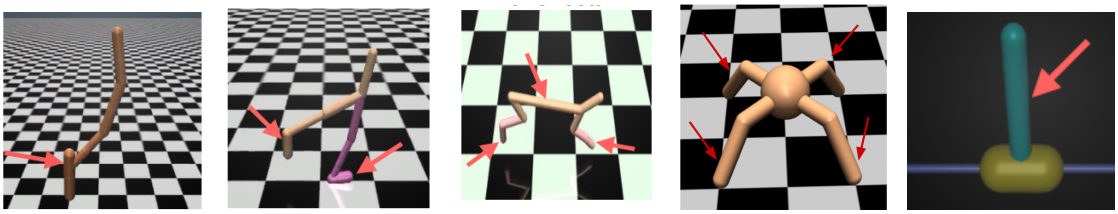
Hyper-parameter	Values
Learning Rate	$\{3\times10^{-3}, 10^{-3}, 3\times10^{-4}, 10^{-4}, 3\times10^{-5}, 10^{-5}\}$
KL Loss Threshold	$\{0.01, 0.02, 0.03\}$
Discount Factor (γ)	$\{0.8, 0.85, 0.9, 0.95, 0.99, 0.995\}$
GAE Parameter (λ)	$\{0.9, 0.93, 0.95, 0.97, 1.0\}$

Table 7: The swept hyper-parameters for PPO.

Hyper-parameter	Values
Learning Rate	$\{3\times10^{-3}, 10^{-3}, 3\times10^{-4}, 10^{-4}, 3\times10^{-5}, 10^{-5}\}$
Clipping Range	$\{0.1, 0.2, 0.3\}$
Discount Factor (γ)	$\{0.8, 0.85, 0.9, 0.95, 0.99, 0.995\}$
GAE Parameter (λ)	$\{0.9, 0.93, 0.95, 0.97, 1.0\}$
Value Function Coefficient	$\{0.5, 1.0\}$
Entropy Coefficient	$\{0, 0.01\}$

Table 8: The swept hyper-parameters for DDPG.

Hyper-parameter	Values
Actor Learning Rate	$\{3\times10^{-3}, 10^{-3}, 3\times10^{-4}, 10^{-4}, 3\times10^{-5}, 10^{-5}\}$
Critic Learning Rate	$\{3\times10^{-3}, 10^{-3}, 3\times10^{-4}, 10^{-4}, 3\times10^{-5}, 10^{-5}\}$
Discount Factor (γ)	$\{0.8, 0.85, 0.9, 0.95, 0.99, 0.995\}$
Soft Update Coefficient	$\{5\times10^{-3}, 5\times10^{-4}, 5\times10^{-5}\}$



(a) Hopper (b) Walker2d (c) Half-Cheetah (d) Ant (e) Pendulum

Figure 9: Illustrations of the environments evaluated in our experiments.

adversarial ensembles to improve robustness against the standard paradigm with a single adversary, we mainly implement TRPO following the RL algorithm used in RARL (Pinto et al., 2017). In addition, we also report the additional experiment results with PPO and DDPG in Section C. We used grid search to select those hyper-parameters and we list the details of swept hyper-parameters specifically for all three RL algorithms in Table 6, 7, and 8. We consider three fully connected layers with 256 neurons for each for all experiments. For those hyper-parameters that are not listed, we adopt the default values in *rllab*.

We train both ROSE-based methods and the baselines for 100 iterations on InvertedPendulum and for 500 iterations on the other tasks. 10 adversarial candidates were considered in RAP and ROSE-S, and ROSE was trained with a set of samples (m = 10). We selected the worst-k adversaries for updating in each iteration with $k = 3$, which is decided by the ablation studies in Table 3.