Towards Robust and Safe Reinforcement Learning with Benign Off-policy Data

Zuxin Liu [*1] **Zijian Guo** [*1] **Zhepeng Cen** [1] **Huan Zhang** [1] **Yihang Yao** [1] **Hanjiang Hu** [1] **Ding Zhao** [1]

Abstract

Previous work demonstrates that the optimal safe reinforcement learning policy in a noise-free environment is vulnerable and could be unsafe under observational attacks. While adversarial training effectively improves robustness and safety, collecting samples by attacking the behavior agent online could be expensive or prohibitively dangerous in many applications. We propose the robuSt vAriational ofF-policy lEaRning (SAFER) approach, which only requires benign training data without attacking the agent. SAFER obtains an optimal non-parametric variational policy distribution via convex optimization and then uses it to improve the parameterized policy robustly via supervised learning. The two-stage policy optimization facilitates robust training, and extensive experiments on multiple robot platforms show the efficiency of SAFER in learning a robust and safe policy: achieving the same reward with much fewer constraint violations during training than on-policy baselines.

1. Introduction

Deep reinforcement learning (RL) has witnessed great success in solving challenging problems (Mnih et al., 2013; Liu et al., 2020a; Jumper et al., 2021). Meanwhile, the potential risks of safety and robustness (Dulac-Arnold et al., 2021; Moos et al., 2022) arise when deploying deep RL in real-world applications. Safe RL has been gaining increasing attention recently, which aims to tackle the safety issue by learning a constraint-satisfaction policy (Garcıa and Fernández, 2015). Safe RL approaches explicitly separate the constraint violation signals and the rewards of task performance, showing advantages to satisfy the safety requirement (Ray et al., 2019; Brunke et al., 2021; Gu et al., 2022). However, recent studies find that *the learned safe RL policies are vulnerable to adversarial attacks*: a small perturbation in the observation space could induce dangerous behaviors of the agent and cause a dramatic drop in the safety performance (Liu et al., 2022c). Therefore, even a well-trained safe RL policy in the noise-free simulation environment may not be truly safe for real-world applications due to the commonly existing sensing noises. Thus, improving its robustness against observational perturbations is equally important to learning a constraint satisfaction policy.

Adversarial training is a commonly-used technique in improving policy robustness and task performance in standard RL setting (Pinto et al., 2017; Gleave et al., 2019). Recent work extends adversarial training to the safe RL domain and suggests that it can also greatly strengthen the safety performance under properly selected attackers (Liu et al., 2022c), where the proposed `ADV-PPOL` method collects corrupted interaction data for model training in an on-policy fashion. This type of method effectively evaluates the safety performance under strong adversarial attacks to improve the robustness. However, it is expensive to perform adversarial training and collect corrupted interaction data online in many applications since the dangerous behaviors induced by attacks could be dangerous or prohibitively unethical (Ibarz et al., 2021). For example, attacking a self-driving vehicle in the physical world by corrupting the vision system can lead to catastrophic accidents (Eykholt et al., 2018; Kong et al., 2020), while benign datasets are much easier to obtain for training. Therefore, investigating a more efficient and safer robust training algorithm for safe RL is an important but challenging problem.

This paper aims to develop a robust safe RL directly from **benign data**, i.e. the natural training samples without attacks. Conversely, **corrupted** data is defined as the rollout trajectories under attacks. Motivated by the Expectation-Maximization (EM) style approaches in RL (Abdolmaleki et al., 2018b; Levine, 2018; Liu et al., 2022b), we convert the robust safe RL problem to a convex optimization phase (E-step) and a supervised learning phase (M-step). Particularly, we propose the robuSt vAriational ofF-policy lEaRning (`SAFER`) method to improve the robustness against adversarial state perturbations. SAFER robustifies the policy in the M-step since it is a supervised learning problem, and thus adversarial training techniques can be effectively

[*]Equal contribution [1]Carnegie Mellon University, PA, USA. Correspondence to: Zuxin Liu <zuxinl@cmu.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

Table 1: Training features and performance comparison.

	Training features		Final performance	
	No attack to the behavior agent	Reuse off-policy training data	Maintain safety under attacks	Maintain reward under attacks
SA-PPOL (Zhang et al., 2020a)	✓	✗	✗	✓
ADV-PPOL (Liu et al., 2022c)	✗	✗	✓	✓
CVPO (Liu et al., 2022b)	✓	✓	✗	✗
SAFER (ours)	✓	✓	✓	✓

applied. More importantly, it only requires benign and off-policy interaction data, which greatly improves training safety and efficiency. We highlight the difference between SAFER and other related training approaches in Table 1. We summarize our contributions as follows:

1. We study the problem of learning a robust and safe policy from benign and off-policy data, which is rarely discussed in the literature. We show that directly applying on-policy adversarial training techniques to the off-policy setting can hardly work well.

2. We propose the SAFER approach from the variational safe RL perspective, which converts the policy learning step into an easy-to-optimize supervised learning problem. The key insight is that adversarial training in the supervised learning phase can improve the robustness and does not require attacking the behavior agent.

3. Our experiment results show that SAFER is effective in learning a robust and safe policy, and it is also safer and more efficient during the learning process than the on-policy robust training baselines.

2. Related Work

Safe RL aims to learn a constraint-satisfaction policy by interacting with the environment (Xu et al., 2022). Domain knowledge of the safety constraint could be applied in an RL system to improve safety (Dalal et al., 2018; Alshiekh et al., 2018; Liu et al., 2020b; Luo and Ma, 2021; Chen et al., 2021; Mguni et al., 2021; Liu et al., 2022a). Another type of approach focuses on the constrained optimization perspective (Sootla et al., 2022; Yang et al., 2021; Flet-Berliac and Basu, 2022). The Lagrangian method is a commonly used technique to adapt a standard RL method to the safe RL setting (Bhatnagar and Lakshmanan, 2012; Chow et al., 2017; Stooke et al., 2020; As et al., 2022). Convex relaxation via low-order Taylor expansions is another widely used approach to solve safe RL (Achiam et al., 2017; Yu et al., 2019). (Zhang et al., 2020c) and (Yang et al., 2020) further propose an additional projection step to recover the updating policy to the safe set and improve the safety performance, while (Liu et al., 2022b) propose a variational inference method to overcome the degradation of safety performance from the approximation error.

Robust RL is another important aspect for developing trustworthy decision-making systems (Xu et al., 2022). Unlike the supervised learning task, robustness in RL has different definitions (Moos et al., 2022), including the robustness against action noises (Tessler et al., 2019), adversarial reward (Wang et al., 2020; Lin et al., 2020; Eysenbach and Levine, 2021), domain shift (Muratore et al., 2018; Huang et al., 2022), and dynamics uncertainty (Lim et al., 2013; Pinto et al., 2017). We focus on the robustness of a deep RL agent under state adversarial attacks (Zhang et al., 2020b; Huang et al., 2017). Utilizing the critics to guide the adversarial perturbation direction is shown to be effective in reducing the agent's performance (Kos and Song, 2017; Lin et al., 2017; Pattanaik et al., 2017). (Zhang et al., 2020a) propose a critic-free adversary by maximizing the KL divergence of the original policy and corrupted policy, which can achieve effective attacks in reducing the agent's reward. The most related work indicates that trained safe RL policies can be easily attacked and thus lead to constraint violations (Liu et al., 2022c).

3. Preliminary

3.1. CMDP and Safe RL

Safe RL is usually modeled under the Constrained Markov Decision Process (CMDP) framework (Altman, 1998). An infinite horizon CMDP $\mathcal{M}$ is defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, c, \gamma, \mu_0)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function, $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function, $\gamma \to [0, 1)$ is the discount factor, and $\mu_0 : \mathcal{S} \to [0, 1]$ is the initial state distribution. CMDP augments the MDP tuple with an additional element $c : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, C_m]$ to characterize the cost for violating the constraint, where C_m is the maximum cost. A safe RL problem is denoted as $\mathcal{M}_\Pi^\kappa := (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, c, \gamma, \mu_0, \Pi, \kappa)$, where Π is a locally Lipschitz continuous policy class, and $\kappa \to [0, +\infty)$ is a threshold for constraint violation cost. Let $\pi(a|s) \in \Pi$ denote the policy and $\tau = \{s_0, a_0, ..., \}$ denote the trajectory. We use shorthand $\mathbf{f}_t = \mathbf{f}(s_t, a_t, s_{t+1}), \mathbf{f} \in \{r, c\}$ for simplicity. The value function is $V_\mathbf{f}^\pi(\mu_0) = \mathbb{E}_{\tau\sim\pi, s_0\sim\mu_0}[\sum_{t=0}^\infty \gamma^t \mathbf{f}_t], \mathbf{f} \in \{r, c\}$, which is the expectation of discounted return under the policy π and the initial state distribution μ_0. We overload the notation $V_\mathbf{f}^\pi(s) = \mathbb{E}_{\tau\sim\pi, s_0=s}[\sum_{t=0}^\infty \gamma^t \mathbf{f}_t], \mathbf{f} \in \{r, c\}$ to denote the value with the initial state $s_0 = s$, and denote $Q_\mathbf{f}^\pi(s, a) = \mathbb{E}_{\tau\sim\pi, s_0=s, a_0=a}[\sum_{t=0}^\infty \gamma^t \mathbf{f}_t]$ as the state-action value function under the policy π. The objective of $\mathcal{M}_\Pi^\kappa$ is to find the policy that maximizes the reward while limiting the cost incurred from constraint violations to a threshold κ:

$$\pi^* = \arg\max_\pi V_r^\pi(\mu_0), \quad s.t. \quad V_c^\pi(\mu_0) \le \kappa. \tag{1}$$

We denote the **feasible** policy class as the set of policies that satisfies the constraint with threshold κ: $\Pi_\mathcal{M}^\kappa := \{\pi(a|s) : V_c^\pi(\mu_0) \le \kappa, \pi \in \Pi\}$; the **optimal** policy π^* as the policy with the highest reward return in the feasible

policy class: $\pi^* \in \Pi^\kappa_{\mathcal{M}}, \forall \pi \in \Pi^\kappa_{\mathcal{M}}, V_r^{\pi^*}(\mu_0) \geq V_r^\pi(\mu_0)$; the **tempting** policy class as the set of policies that have a higher reward return than the optimal policy: $\Pi^T_{\mathcal{M}} := \{\pi(a|s) : V_r^\pi(\mu_0) > V_r^{\pi^*}(\mu_0), \pi \in \Pi\}$. Note that a tempting policy is unsafe, and the existence of tempting policies induces challenges in safe RL (Liu et al., 2022c).

3.2. State Adversarial Safe RL

We study the observational robustness of safe RL under the state-adversarial safe RL framework (Zhang et al., 2020a;b). A deterministic adversary $\nu(s) : \mathcal{S} \to \mathcal{S}$ corrupts the state observation of the agent, aiming to increase the constraint violation rate. The corrupted state is denoted as $\tilde{s} := \nu(s)$, and the corrupted policy is denoted as $\pi \circ \nu := \pi(a|\tilde{s}) = \pi(a|\nu(s))$, where the state is first contaminated by ν and then processed by the policy π. Note that the adversary does not modify the true states in the original CMDP, but only the agent's input. We restrict the perturbed observation to be within a pre-defined perturbation set $B(s)$: $\forall s \in \mathcal{S}, \nu(s) \in B(s)$, such that the power of the adversary is limited and maintaining attacking stealthiness. Following convention (Madry et al., 2017), we define the perturbation set $B^\epsilon_p(s)$ as the ℓ_p-ball around the original state: $\forall s' \in B^\epsilon_p(s), \|s' - s\|_p \leq \epsilon$, where ϵ is the ball size. The observation adversary commonly exists in real-world applications, such as sensing noise or perception errors (Hu et al., 2022).

Different from standard RL, safe RL has to ensure constraint satisfaction, since the cost of violating constraints in many safety-critical applications can be unaffordable. Therefore, the primary goal for the adversary in safe RL is to increase the constraint violation cost. There are two strong adversarial attackers in prior work Liu et al. (2022c): `Maximum-Cost` (MC) and `Maximum-Reward` (MR). The MC attacker directly maximizes the cost return to obtain the perturbation: $\nu_{\text{MC}} = \arg\max_\nu V_c^{\pi\circ\nu}(\mu_0)$, while the MR attacker maximizes the reward return to make the policy be tempting: $\nu_{\text{MR}} = \arg\max_\nu V_r^{\pi\circ\nu}(\mu_0)$. Both attackers are shown to be effective in inducing unsafe behaviors of the safe RL agent, and the MR attacker is also stealthy in maintaining the task reward such that the agent can not be aware of attacks easily. Note that the minimizing-reward adversaries in standard RL can hardly work in decreasing safety performance.

4. Method

This section introduces the SAFER algorithm, which is built upon variational inference-based RL methods (Abdolmaleki et al., 2018b; Liu et al., 2022b), aiming to robustify the policy under observational perturbations. We first present how to formulate safe RL as a variational inference problem, and then introduce how to optimize the policy to improve its robustness against adversarial inputs.

4.1. Robust Safe RL as Inference

The classical view of safe RL aims to find the actions that could maximize task rewards while satisfying the constraints, while the probabilistic inference perspective seeks to answer "given future success in maximizing task rewards, what are the **feasible** actions most likely to have been taken?". Following the RL as inference literature (Levine, 2018), we consider an infinite discounted reward formulation and define the optimality variable of a state-action pair as O, which represents the event of maximizing the reward by choosing an action at a state. Then the likelihood of being optimal given a trajectory is proportional to the exponential of the discounted cumulative reward: $p(O = 1 \mid \tau) \propto \exp(\sum_t \gamma^t r_t/\alpha)$, where α is a temperature parameter. Denote the probability of a trajectory τ under the policy π as $p_\pi(\tau) = p(s_0)\prod_{t\geq 0} p(s_{t+1} \mid s_t, a_t)\pi(a_t \mid s_t)$, then the lower bound of the log-likelihood of optimality given the policy π is:

$$\begin{aligned}
&\log p_\pi(O=1) = \log \mathbb{E}_{\tau\sim q}[\frac{p(O=1\mid\tau)p_\pi(\tau)}{q(\tau)}] \\
&\geq \mathbb{E}_{\tau\sim q} \log \frac{p(O=1\mid\tau)p_\pi(\tau)}{q(\tau)} \\
&\propto \mathbb{E}_{\tau\sim q}[\sum_{t=0}^{\infty}\gamma^t r_t] - \alpha D_{\text{KL}}(q(\tau)\|p_\pi(\tau)) = \mathcal{J}(q,\pi)
\end{aligned} \tag{2}$$

where the inequality follows Jensen's inequality, $q(\tau)$ is an auxiliary trajectory-wise variational distribution. $\mathcal{J}(q,\pi)$ in equation (2) is the evidence lower bound (ELBO), which is an important quantity in Expectation-Maximization (EM). EM-based RL algorithms alternate to improve $\mathcal{J}(q,\pi)$ in terms of $q(\tau)$ and $p_\pi(\tau)$ such that the likelihood objective of optimality is increased (Abdolmaleki et al., 2018b;a). More specifically, the E-step optimizes $q(\tau)$ to maximize the discounted return within the trust region of the old policy, while the M-step aims to minimize the KL divergence between $p_\pi(\tau)$ and $q(\tau)$ by updating the parametrized policy in a supervised learning fashion. Off-policy deep RL techniques can be used during training, making the EM updating steps scalable and data efficient. By factorizing the variational distribution $q(\tau)$ the same way as $p_\pi(\tau)$: $q(\tau) = p(s_0)\prod_{t\geq 0} p(s_{t+1}|s_t,a_t)q(a_t|s_t)$ and cancelling the transitions, we have the following ELBO over the state-conditioned action distribution $q(a|s)$:

$$\mathcal{J}(q,\theta) = \mathbb{E}_{\rho_q}\Big[\sum_{t=0}^{\infty}\gamma^t r_t - \alpha D_{\text{KL}}(q\|\pi_\theta)\Big] + \log p(\theta) \tag{3}$$

where $\rho_q(s)$ is the stationary state distribution induced by $q(\cdot|s)$ and ρ_0, θ denotes the parameters for policy π, and $p(\theta)$ is a prior distribution over the parameters. Note we overload q by using it both in $q(a|s)$ and $q(\tau)$. In safe RL, an additional constraint on the variational distribution is required to ensure safety. It is done by limiting the

choices of the variational distribution $q(\cdot|s)$ in a feasible distribution family $\Pi_{\mathcal{Q}}^{\kappa}$ with threshold κ: $\Pi_{\mathcal{Q}}^{\kappa} := \{q(a|s) : \mathbb{E}_{\tau\sim q}[\sum_{t=0}^{\infty}\gamma^t c_t] \leq \kappa, a \in \mathcal{A}, s \in \mathcal{S}\}$, which is a set of all the state-conditioned action distributions that satisfy the safety constraint. Optimizing the factorized lower bound $\mathcal{J}(q,\theta)$ w.r.t q within the feasible distribution family and the policy parameter θ iteratively via EM yields the CVPO safe RL method (Liu et al., 2022b). Note that our approach is built upon CVPO framework with the same E-step but a different M-step. We will introduce each step as follows.

4.2. Constrained E-step

The E-step step aims to find the optimal variational distribution $q \in \Pi_{\mathcal{Q}}^{\kappa}$ that maximizes the reward return while satisfying the safety constraint. At the i-th iteration, We can write the ELBO objective w.r.t q as a constrained optimization problem (see Appendix A.5 for details):

$$\begin{aligned} \max_{q} \quad & \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q(\cdot|s)}\big[Q_r^{\pi_{\theta_i}}(s,a)\big]\Big], \\ s.t. \quad & \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q(\cdot|s)}\big[Q_c^{\pi_{\theta_i}}(s,a)\big]\Big] \leq \kappa; \\ & \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q(\cdot|s)}\big[D_{\mathrm{KL}}(q\|\pi_{\theta_i})\big]\Big] \leq \delta \end{aligned} \tag{4}$$

where $\rho_q(s)$ is the stationary state distribution induced by $q(a|s)$ and ρ_0, which can be approximated by the replay buffer. We use $Q_{\mathbf{f}}^{\pi_{\theta_i}}(s,a) = \mathbb{E}_{\tau\sim\pi_{\theta_i},s_0=s,a_0=a}\Big[\sum_{t=0}^{\infty}\gamma^t\mathbf{f}_t\Big], \mathbf{f} \in \{r,c\}$ denote the reward and cost state-action value function. The two constraints aim to ensure the optimized variational distribution is within the feasible set $\Pi_{\mathcal{Q}}^{\kappa}$ and the trust region of the old policy, respectively.

Note that the objective function and the constraints are all convex w.r.t the decision variable q, so we adopt a nonparametric form of the variational distribution. Particularly, we use K samples in the action space to represent $q(\cdot|s)$ for continuous action space. Therefore, solving the E-step could be viewed as a convex optimization problem with finite decision variables. The choice of a nonparametric form of q in the E-step formulation gives a good property: we could obtain the optimal (and in most cases unique) solution in an analytical form (5) after solving a convex dual problem, as shown in Proposition 1 (Liu et al., 2022b):

Proposition 1. *Suppose the problem (4) has a feasible solution, then the optimal distribution $q_i^*(\cdot|s)$ has the form:*

$$q_i^*(a|s) = \frac{\pi_{\theta_i}(a|s)}{Z(s)}\exp\left(\frac{Q_r^{\theta_i}(s,a) - \lambda^* Q_c^{\theta_i}(s,a)}{\eta^*}\right) \tag{5}$$

where $Z(s)$ is a constant normalizer to ensure q^* is a valid distribution, and the dual variables η^* and λ^* are the solutions to the following convex optimization problem:

$$\begin{aligned} \min_{\lambda\geq 0,\eta\geq 0} \quad & g(\eta,\lambda) = \lambda\kappa + \eta\delta + \\ & \eta\mathbb{E}_{\rho_q}\left[\log\mathbb{E}_{\pi_{\theta_i}}\left[\exp\left(\frac{Q_r^{\theta_i}(s,a) - \lambda Q_c^{\theta_i}(s,a)}{\eta}\right)\right]\right] \end{aligned} \tag{6}$$

The proof is in Appendix A.6. Eq. (5) indicates that the optimal q is re-weighted based on the old policy π_{θ_i}, where the weights are controlled by $Q_r^{\theta_i}(s,a), Q_c^{\theta_i}(s,a), \eta, \lambda$. Note that the $Q_r^{\theta_i}(s,a)$ and $Q_c^{\theta_i}(s,a)$ are constants since they could be viewed as the evaluation of the state-action pair (s,a). On the one hand, the action probability density is high if it corresponds to a high task reward return and a low safety cost return, where the weight between them is balanced by λ. On the other hand, the dual variable η serves as a temperature to prevent the action distribution from collapsing to a sharp one, because a low-variance policy distribution discourages the agent to explore new actions. The solution of η is related to the KL-constraint threshold δ, which makes sense since we limit the updating policy within a trust region. One exciting property is that the dual problem (6) is strongly convex with mild assumptions, which guarantees the optimality and uniqueness of the solution and improves the optimization efficiency. The details can be found in (Liu et al., 2022b).

In summary, during the E-step, we obtain the optimal variational distribution that 1) maximizes the task rewards, 2) belongs to the feasible distribution family $\Pi_{\mathcal{Q}}^{\kappa}$, and 3) stays within the trust region of the old policy. It has a closed-form solution regarding two dual variables, which can be efficiently solved by convex optimization.

4.3. Robust M-step

The optimal nonparametric distributions q in E-step cannot cover the full state space, so we need a parametrized policy, such as a neural network, to fit q and then achieve generalization beyond the state-action samples used for training, which yields the **vanilla M-step**. This step improves the ELBO w.r.t the policy parameter θ_i, where i is the training iteration index. By removing the terms that are irrelevant to θ in Eq. (3) and using the Gaussian prior $\theta \sim \mathcal{N}(\theta_i, \frac{F_{\theta_i}}{\alpha\beta})$, we obtain the following optimization problem (Abdolmaleki et al., 2018a;b):

$$\begin{aligned} \max_{\theta} \quad & \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q_i^*(\cdot|s)}\big[\log\pi_\theta(a|s)\big]\Big] \\ s.t. \quad & \mathbb{E}_{\rho_q}\big[D_{\mathrm{KL}}(\pi_{\theta_i}(a|s)\|\pi_\theta(a|s))\big] \leq \xi. \end{aligned} \tag{7}$$

The detailed derivation is presented in Appendix A.7. Solving Eq. (7) is essentially a constrained supervised learning problem, where the objective is a maximum likelihood estimation loss, and the constraint is the KL divergence between the updated policy and the old policy. Intuitively, the

parametrized policy is optimized to approximate the optimal nonparametric variational distribution, but the updating step is limited within the trust region of the old policy. The KL constraint is necessary for regularizing the policy improvement and improving the training stability, because the value estimations could be inaccurate when the policy is out of the trust region. This constraint is commonly used in the standard RL literature (Schulman et al., 2015; 2017).

The vanilla M-step formulation doesn't consider the policy robustness against adversarial inputs, while previous works suggest that neural networks are vulnerable to adversarial attacks (Machado et al., 2021). The sensitivity of the observation space may lead the safe RL agent to perform dangerous behaviors and violate safety constraints. Therefore, we propose an additional **adversarial training loss** combined with the vanilla loss in Eq. (7) to improve the policy robustness by optimizing toward the worst-case perturbations:

$$\begin{aligned} \max_{\theta} \quad & \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q_i^*(\cdot|s)}\big[\log \pi_\theta(a|\nu(s))\big]\Big] \\ s.t. \quad & \mathbb{E}_{\rho_q}\big[D_{\mathrm{KL}}(\pi_{\theta_i}(a|s)\|\pi_\theta(a|\nu(s)))\big] \le \tilde{\xi}, \end{aligned} \tag{8}$$

where $\nu(s)$ is the corrupted state from the adversary. Optimizing the objective function in Eq. (8) could be viewed as training a model with an augmented adversarial dataset in supervised learning literature (Volpi et al., 2018; Shorten and Khoshgoftaar, 2019), while the constraint aims to smooth the policy to improve the robustness (Madry et al., 2018; Zhang et al., 2019; 2020a). Note that the robust M-step in Eq. (8) is optimized together with the vanilla M-step loss in Eq. (7), see Appendix Eq. (62) for details.

We could interpret this additional adversarial training loss from two perspectives: 1) smoothing the policy by constraining the KL divergence between the corrupted state-conditional action distribution and the benign one, and 2) the smoothing direction should be towards the optimal variational distribution of the benign state. We propose to use the `Maximum-Cost` (MC) adversary ν_{MC} since it directly outputs the worst-case perturbations that can lead to the maximum future constraint violations, as we introduced in Sec. 3.2. The MC adversary is of the form $\nu_{\mathrm{MC}} = \arg\max_\nu V_c^{\pi\circ\nu}(\mu_0)$, which could be solved by the Q-value-based formulation in practice: $\nu_{\mathrm{MC}}(s) = \arg\max_{\tilde{s}\in B_p^\epsilon(s)} \mathbb{E}_{\tilde{a}\sim\pi(a|\tilde{s})}\left[Q_c^\pi(s,\tilde{a})\right]$. With the strong adversary in the robust M-step, SAFER enjoys many good theoretical properties, as we introduce in the next section.

4.4. Theoretical Analysis

The **vanilla M-step** in Eq. (7) together with the adversarial training in Eq. (8) yields the **robust M-step**. To better understand how does the robust M-step improve safety and robustness when compared to the vanilla M-step, we derive their worst-case cost value bounds under observational perturbations, respectively. Denote $\mathcal{S}_c$ as the set of unsafe states that have non-zero cost, p_s as the maximum probability of entering unsafe states from state s: $p_s = \max_a \sum_{s'\in\mathcal{S}_c} p(s'|s,a)$, and $A_c^\pi(s,a) = Q_c^\pi(s,a) - V_c^\pi(s)$ as the cost advantage function of π. For simplicity, we use shorthand $\pi_i = \pi_{\theta_i}$ and use $\pi_{i+1}^{(V)}, \pi_{i+1}^{(R)}$ to denote updated policies after vanilla M-step or robust M-step. Then given **any** adversary $\nu(s)$ with an ℓ_p-ball $B_p^\epsilon(s)$ perturbation set, the constraint violation cost values for them are upper bounded by the following theorems.

Theorem 1 (Cost bound of vanilla M-step under attacks)**.** *Suppose the parameterized policy π is locally L-Lipschitz continuous: $D_{TV}[\pi(\cdot|s')\|\pi(\cdot|s)] \le L\,\|s'-s\|_p$, and the policy π_i is feasible (i.e., $\pi_i \in \Pi_{\mathcal{M}}^\kappa$), then we have:*

$$\begin{aligned} V_c^{\pi_{i+1}^{(V)}\circ\nu}(\mu_0) \le \kappa + 2C_m \Bigg(& \frac{d_{i+1}^{(V)} + L\epsilon}{1-\gamma} + \\ & \frac{\sqrt{2\xi}\gamma d_{i+1}^{(V)} + 4\gamma L^2\epsilon^2}{(1-\gamma)^2}\Bigg)\left(\max_s p_s + \frac{\gamma}{1-\gamma}\right), \end{aligned} \tag{9}$$

where $d_{i+1}^{(V)} = \max_s D_{TV}[\pi_{i+1}^{(V)}(\cdot|s)\|\pi_i(\cdot|s)]$ denotes the maximum TV distance between the policies before and after vanilla M-step, and C_m is the maximum one-step cost.

Theorem 2 (Cost bound of robust M-step under attacks)**.** *Suppose π_i is feasible, then the safety performance of policy after robust M-step under attacks is bounded by:*

$$\begin{aligned} V_c^{\pi_{i+1}^{(R)}\circ\nu}(\mu_0) \le \kappa + 2C_m & \left(\frac{d_{i+1}^{(R)}}{1-\gamma} + \frac{\sqrt{2\tilde{\xi}}\gamma d_{i+1}^{(R)}}{(1-\gamma)^2}\right)\cdot \\ & \left(\max_s p_s + \frac{\gamma}{1-\gamma}\right), \end{aligned} \tag{10}$$

where $d_{i+1}^{(R)} = \max_s D_{TV}[\pi_{i+1}^{(R)}(\cdot|\nu(s))\|\pi_i(\cdot|s)]$ denotes the maximum TV distance between the corrupted policies after robust M-step and benign policy before M-step, and C_m is the maximum one-step cost.

The proofs are in Appendix A.1 & A.2. We can find that the robust M-step provides a tighter upper bound than the vanilla M-step in general, since the Lipschitz continuity L of π^V after the vanilla M-step could be an unbounded value, while the robust M-step explicitly restricts the smoothness of π^R by the KL divergence threshold $\tilde{\xi}$, which significantly improves safety under strong attacks.

Although Theorem 2 is mainly derived based on the adversarial training step in Eq.(8), we argue that the vanilla M-step in Eq. (7) is of equal importance in training, because the corrupted policy $\pi\circ\nu$ can correspond to multiple benign policies π and thus lead to the convergence issue. We define it formally as follows (proof is in Appendix A.3).

Proposition 2. *Given a fixed adversary ν, suppose there exists $s_1, s_2 \in \mathcal{S}$ such that $\nu(s_1) = \nu(s_2)$, then there exists two natural policies $\pi_1, \pi_2 \in \Pi$ such that $\pi_1 \circ \nu \triangleq \pi_2 \circ \nu$.*

The analysis is based on a fixed adversary hypothesis, while in practice, we use stronger and policy-dependent adversaries such as the MC attacker. We can still prove that there are multiple policies sharing the same corrupted policy under MC (or MR) attack under mild conditions. The precise statement and proofs are in Appendix A.4. Since the corrupted policy does not always correspond to a unique benign policy, the adversarial training step in Eq. (8) alone can hardly ensure the training convergence. Furthermore, since we robustify the policy without attacking the behavior agents, the additional adversarial training step can contaminate the learning progress in noise-free environments. Therefore, ensuring training stability with benign data is necessary. Luckily, since our method is derived from the EM-based algorithm, therefore, the training robustness can be achieved by incorporating the vanilla M-step in Eq. (7). More details regarding this property can be found in Proposition 3 in (Liu et al., 2022b).

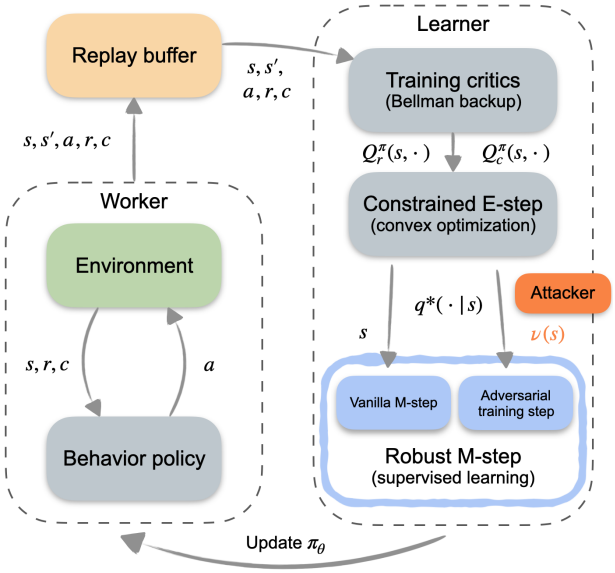


Figure 1: Figure illustration of SAFER.

4.5. Practical Implementation

Fig. 1 shows the training pipeline of SAFER. Different from other adversarial training methods in the literature, SAFER only requires benign data for training and could be implemented in an off-policy fashion, which greatly improves training safety and efficiency. Algo. 1 highlights the key steps of training the policy. Practically, we approximate the stationary state distribution ρ_q by the samples from the replay buffer. More implementation details and training tricks are available in Appendix B.1.

Algorithm 1 SAFER Algorithm at the i-th Iteration

Rollout **benign** trajectories by π_{θ_i}
for each policy optimization iteration **do**
  Sample N transitions from the replay buffer
  Update $Q_r^{\pi_{\theta_i}}, Q_c^{\pi_{\theta_i}}$ by the Bellman equation.
  ▷ *Constrained E-step begins*
  Compute dual variables η^*, λ^* by solving (6)
  Compute the variational distribution for each state $\{q^*(\cdot|s_n); n = 1, ..., N\}$ by Eq. (5)
  ▷ *Robust M-step begins*
  for each M-step iteration **do**
    Get adversarial states $\{\nu_{\text{MC}}(s_n); n = 1, ..., N\}$
    Optimize π_{θ_i} one-step by solving (7) and (8)
  end for
end for

5. Experiment

We consider two tasks (Run and Circle) and four robots (Ball, Car, Drone, and Ant) which have been used in many previous works as the testing ground (Achiam et al., 2017; Chow et al., 2019). The simulation environments are from a publicly available benchmark (Gronauer, 2022). For the Run task, the agents are rewarded for running fast between two boundaries and are given constraint violation cost if they run across the boundaries or exceed an agent-specific velocity threshold. For the Circle task, the agents are rewarded for running in a circle but are constrained within a safe region that is smaller than the radius of the target circle. We name the tasks as `Ball-Circle`, `Car-Circle`, `Drone-Run`, and `Ant-Run`.

On-policy baselines. We use the adversarial training algorithm **ADV-PPOL** proposed by (Liu et al., 2022c) as the major on-policy baseline, which collects corrupted trajectories by the MC attacker to train the base PPO-Lagrangian (PPOL) agent. We use a robust training algorithm that is effective in standard RL **SA-PPOL** as another baseline (Zhang et al., 2020a). We also extend it by changing the MAD attacker to a stronger MC attacker in the safe RL setting, which yields the **SA-PPOL(MC)** baseline.

Off-policy baselines. Since SAFER is closely related to the EM-based safe RL algorithm CVPO (Liu et al., 2022b), we use it as a basic baseline and name it as **CVPO-vanilla**. We adopt its variant **CVPO-random** that is trained under random noise as another baseline. In addition, we directly apply the same online adversarial training techniques with the MC attacker in ADV-PPOL to the off-policy setting, which yields the **ADV-CVPO** baseline.

Metrics. We compare the methods in terms of episodic reward (the higher, the better) and episodic constraint violation cost (the lower, the better). We also compare the sample efficiency of utilizing each constraint violation.

Table 2: Evaluation results of natural performance (no attack) and under attacks. Each value is averaged over 50 episodes and 5 seeds. We shadow two lowest-cost agents under each attacker column and break ties based on rewards, excluding the failing agents, whose natural rewards are less than 10% of CVPO-vanilla's and are marked with $\star$.

Env	Method		Natural		MC		MR		Average	
			Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost
Car-Circle $\epsilon = 0.05$	On-policy	SA-PPOL	440.79	0.25	275.01	71.62	393.56	96.4	369.79	56.09
		SA-PPOL(MC)	439.44	0.35	348.21	91.88	375.59	56.22	387.75	49.48
		ADV-PPOL	300.0	0.0	338.79	0.28	281.64	0.47	306.81	0.25
	Off-policy	CVPO-vanilla	297.2	0.38	244.4	57.21	287.4	38.75	276.33	32.11
		CVPO-random	170.33	0.0	166.5	7.51	214.61	2.3	183.81	3.27
		$\star$ADV-CVPO	0.42	0.0	0.48	0.0	0.57	0.0	0.49	0.0
		SAFER	258.63	0.0	313.61	0.1	333.29	0.32	301.84	0.14
Drone-Run $\epsilon = 0.025$	On-policy	SA-PPOL	338.07	0.0	-72.72	59.27	258.35	69.97	174.57	43.08
		SA-PPOL(MC)	183.9	0.0	206.77	5.1	214.32	5.25	201.66	3.45
		ADV-PPOL	265.84	0.0	298.7	1.5	264.37	0.55	276.3	0.68
	Off-policy	CVPO-vanilla	347.17	0.0	353.62	63.52	387.82	71.28	362.87	44.93
		CVPO-random	284.98	0.0	300.0	40.63	337.03	72.32	307.33	37.65
		ADV-CVPO	222.7	38.0	309.14	63.35	285.16	56.75	272.33	52.7
		SAFER	193.26	0.0	210.99	0.63	220.17	0.68	208.14	0.44
Ant-Run $\epsilon = 0.025$	On-Policy	SA-PPOL	699.78	1.47	692.34	62.2	714.38	103.63	702.17	55.77
		SA-PPOL(MC)	547.23	1.03	575.58	25.65	584.42	28.35	569.07	18.34
		ADV-PPOL	608.02	0.0	668.25	0.55	672.79	1.32	649.69	0.62
	Off-policy	CVPO-vanilla	686.59	0.85	668.53	86.03	728.37	164.65	694.5	83.84
		CVPO-random	682.28	1.17	672.27	100.5	747.65	173.45	700.73	91.71
		ADV-CVPO	334.31	79.08	302.15	75.02	330.5	68.3	322.32	74.13
		SAFER	496.11	0.17	514.8	0.82	555.81	1.5	522.24	0.83

5.1. Main Results and Analysis

The evaluation results are shown in Table 2, where **Natural** represents the performance without noise. We shadow the two safest agents with the lowest cost values except for the failure agents (marked with $\star$) whose rewards are less than 10% of the CVPO-vanilla method. The complete results with more attackers are deferred in Appendix B.4. We summarize the findings as follows.

First, we can observe that the vanilla EM-based safe RL method CVPO-vanilla and its variant CVPO-random (adding random noises) are vulnerable to adversarial attacks, although they can attain near zero natural cost in a noise-free environment. **The poor safety performance of these approaches under attacks indicates the necessity of studying their robustness**, which is rarely discussed in the safe RL literature. The on-policy robust training algorithm SA-PPOL that works well in maintaining the reward can hardly ensure safety under strong attacks, even if it is trained with the MC attacker. The only baseline that performs well in safety, robustness, and task performance is the on-policy adversarial training method ADV-PPOL. However, **the successful on-policy adversarial training techniques in ADV-PPOL do not work in the off-policy setting**, as the ADV-CVPO method is not safe under adversarial attackers and even performs poorly in noise-free environments. We also conducted an ablation study by training the agents with corrupted data obtained by sampling benign data from the reply buffer. However, the agents have similar unsafe behavior as ADV-CVPO and are not robust against adversarial attackers. More details of the results and discussions about the failures are in Appendix B.4.

Second, we can clearly see that **SAFER learns a safe and robust policy under adversarial attacks**, as it can achieve comparable performance to the ADV-PPOL method and consistently outperforms other baselines in safety and robustness with the lowest cost while maintaining the task reward. However, ADV-PPOL requires on-policy corrupted trajectories for training, while SAFER can robustify the policy through benign and off-policy data that are much easier to obtain in real-world applications.

Finally, **SAFER is more sample efficient during learning.** Fig. 2 demonstrates the efficacy of utilizing each cost, i.e., how much task rewards the agent can obtain given a budget of constraint violations. Since the off-policy baselines cannot provide a safe and robust policy, we only compare SAFER with on-policy baselines. The x-axis is the cumulative cost during the training, and the y-axis is the maximum episodic rewards achieved from the start of training. The curve that is closer to the upper left is better because fewer costs are required to achieve high rewards. Note that the x-axis is of the log scale. We can clearly see that SAFER outperforms on-policy baselines with a large mar-

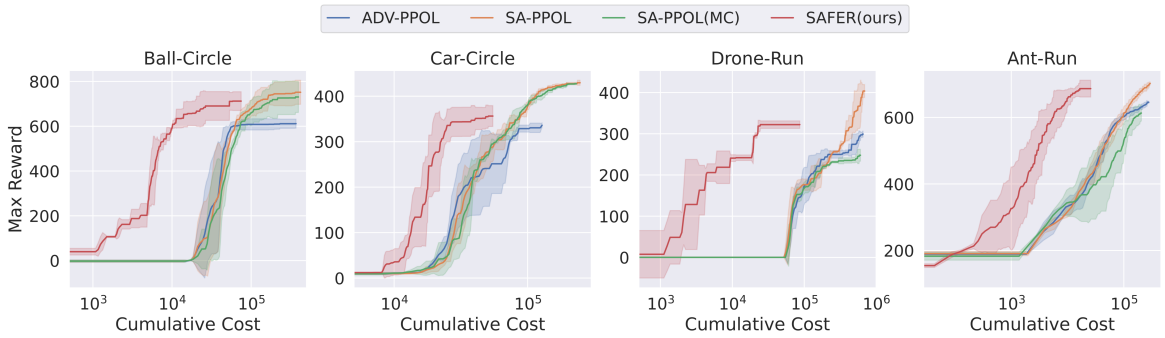


Figure 2: Reward versus cumulative cost (log-scale).

gin among all tasks: it uses fewer cumulative constraint violations to achieve the same task reward. The results also validate our hypothesis that **using benign and off-policy data to robustify the policy is safer and more efficient during training than using on-policy corrupted samples**. Therefore, we can conclude that the proposed SAFER algorithm is capable of learning a robust and safe policy under strong attacks, as well as improving training safety and efficiency compared with existing baselines.

5.2. Ablation experiment

To study the influence of the vanilla M-step (VM) in Eq. (7), the KL constraint and the maximum-likelihood (MLE) style loss in the adversarial training step Eq. (8), we conduct an ablation study by removing each component from the full SAFER algorithm. Table 3 shows the experiment results, where we only present the average performance due to the page limit. The detailed results are presented in Appendix B.4. We can clearly see that the agent fails to learn if we remove the vanilla M-step in training, which is because the adversarial training alone can not ensure the agent's performance in the natural environment and thus corrupt the learning process, as we introduced in Proposition 2. We can also observe significant safety performance (cost) degradation if we remove the KL constraint and task performance (reward) drop if we remove the MLE loss in the adversarial training step. Therefore, **both the vanilla M-step and the adversarial training step are necessary and important components for SAFER**.

We also study the relationship between the safety performance of SAFER w.r.t the constraint threshold $\tilde{\xi}$ in the adversarial training step. The results are available in Appendix B.4. A summary is that the KL constraint threshold $\tilde{\xi}$ directly affects the safety performance under strong attacks: smaller thresholds can usually achieve better safety performance, which validates the worst-case cost value bound in Theorem 2. A downside is that improving safety under attacks is usually at the cost of sacrificing reward in natural environments, so we should carefully balance the trade-off between robustness and task performance.

Table 3: Ablation study of removing the vanilla M-step (VM), the KL constraint, and the MLE loss.

		Ball-Circle	Car-Circle	Ant-Run
without VM	Reward	165.19	11.36	45.91
	Cost	0.0	2.54	0.06
without KL	Reward	544.66	360.86	146.43
	Cost	5.77	23.53	1.58
without MLE	Reward	535.98	283.73	179.0
	Cost	1.23	1.99	0.52
Full SAFER	Reward	632.39	301.84	522.24
	Cost	0.82	0.14	0.83

6. Conclusion

We propose the SAFER method that only requires benign and off-policy data to train a robust and safe policy under observational perturbations. We analyze the theoretical properties of each component of the SAFER algorithm and conduct comprehensive experiments and ablation studies to validate our arguments and claims. The results show that the SAFER agent effectively maintains safety under adversarial attacks. More importantly, it is much safer and more efficient during training than baseline robust training methods by using fewer costs to achieve the same reward.

One limitation of this work is that the decoupled E-step and M-step require more computation and are thus with slower training speed than policy-gradient-based approaches. In addition, the SAFER algorithm is only evaluated with simple safe RL tasks due to the limitation of off-policy safe RL algorithms: accurately estimating the Q value function of sparse cost signals for long-horizon tasks is challenging. Therefore, studying whether these off-policy methods can still perform well in more difficult tasks such as SafetyGym environments (Ray et al., 2019) leaves future work. The potential negative societal impact includes the misuse of the adversarial attacking method in real systems. However, we hope our findings about the potential risks of deploying safe RL methods can inspire more interdisciplinary research in this direction, because both safety and robustness are non-negligible factors in safety-critical applications.

ACKNOWLEDGEMENTS

We gratefully acknowledge support from the National Science Foundation under grant CAREER CNS-2047454. Equally, we extend our sincerest appreciation to the reviewers for their invaluable suggestions.

References

Abdolmaleki, A., Springenberg, J. T., Degrave, J., Bohez, S., Tassa, Y., Belov, D., Heess, N., and Riedmiller, M. (2018a). Relative entropy regularized policy iteration. *arXiv preprint arXiv:1812.02256*.

Abdolmaleki, A., Springenberg, J. T., Tassa, Y., Munos, R., Heess, N., and Riedmiller, M. (2018b). Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*.

Achiam, J., Held, D., Tamar, A., and Abbeel, P. (2017). Constrained policy optimization. In *International Conference on Machine Learning*, pages 22–31. PMLR.

Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., and Topcu, U. (2018). Safe reinforcement learning via shielding. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Altman, E. (1998). Constrained markov decision processes with total cost criteria: Lagrangian approach and dual linear program. *Mathematical methods of operations research*, 48(3):387–417.

As, Y., Usmanova, I., Curi, S., and Krause, A. (2022). Constrained policy optimization via bayesian world models. *arXiv preprint arXiv:2201.09802*.

Bhatnagar, S. and Lakshmanan, K. (2012). An online actor–critic algorithm with function approximation for constrained markov decision processes. *Journal of Optimization Theory and Applications*, 153(3):688–708.

Brunke, L., Greeff, M., Hall, A. W., Yuan, Z., Zhou, S., Panerati, J., and Schoellig, A. P. (2021). Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5.

Chen, B., Liu, Z., Zhu, J., Xu, M., Ding, W., Li, L., and Zhao, D. (2021). Context-aware safe reinforcement learning for non-stationary environments. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10689–10695. IEEE.

Chow, Y., Ghavamzadeh, M., Janson, L., and Pavone, M. (2017). Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120.

Chow, Y., Nachum, O., Faust, A., Duenez-Guzman, E., and Ghavamzadeh, M. (2019). Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1901.10031*.

Dalal, G., Dvijotham, K., Vecerik, M., Hester, T., Paduraru, C., and Tassa, Y. (2018). Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*.

Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., and Hester, T. (2021). Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468.

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D. (2018). Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1625–1634.

Eysenbach, B. and Levine, S. (2021). Maximum entropy rl (provably) solves some robust rl problems. *arXiv preprint arXiv:2103.06257*.

Flet-Berliac, Y. and Basu, D. (2022). Saac: Safe reinforcement learning as an adversarial game of actor-critics. *arXiv preprint arXiv:2204.09424*.

Garcıa, J. and Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480.

Gleave, A., Dennis, M., Wild, C., Kant, N., Levine, S., and Russell, S. (2019). Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615*.

Gronauer, S. (2022). Bullet-safety-gym: Aframework for constrained reinforcement learning.

Gu, S., Yang, L., Du, Y., Chen, G., Walter, F., Wang, J., Yang, Y., and Knoll, A. (2022). A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*.

Hu, H., Liu, Z., Li, L., Zhu, J., and Zhao, D. (2022). Robustness certification of visual perception models via camera motion smoothing. In *6th Annual Conference on Robot Learning*.

Huang, P., Xu, M., Fang, F., and Zhao, D. (2022). Robust reinforcement learning as a stackelberg game via adaptively-regularized adversarial training. *arXiv preprint arXiv:2202.09514*.

Huang, S., Papernot, N., Goodfellow, I., Duan, Y., and Abbeel, P. (2017). Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*.

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.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589.

Kong, Z., Guo, J., Li, A., and Liu, C. (2020). Physgan: Generating physical-world-resilient adversarial examples for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14254–14263.

Kos, J. and Song, D. (2017). Delving into adversarial attacks on deep policies. *arXiv preprint arXiv:1705.06452*.

Levine, S. (2018). Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*.

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

Lim, S. H., Xu, H., and Mannor, S. (2013). Reinforcement learning in robust markov decision processes. *Advances in Neural Information Processing Systems*, 26.

Lin, Y.-C., Hong, Z.-W., Liao, Y.-H., Shih, M.-L., Liu, M.-Y., and Sun, M. (2017). Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748*.

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, P., Tateo, D., Ammar, H. B., and Peters, J. (2022a). Robot reinforcement learning on the constraint manifold. In *Conference on Robot Learning*, pages 1357–1366. PMLR.

Liu, Z., Cen, Z., Isenbaev, V., Liu, W., Wu, S., Li, B., and Zhao, D. (2022b). Constrained variational policy optimization for safe reinforcement learning. In *International Conference on Machine Learning*, pages 13644–13668. PMLR.

Liu, Z., Chen, B., Zhou, H., Koushik, G., Hebert, M., and Zhao, D. (2020a). Mapper: Multi-agent path planning with evolutionary reinforcement learning in mixed dynamic environments. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11748–11754. IEEE.

Liu, Z., Guo, Z., Cen, Z., Zhang, H., Tan, J., Li, B., and Zhao, D. (2022c). On the robustness of safe reinforcement learning under observational perturbations. *arXiv preprint arXiv:2205.14691*.

Liu, Z., Zhou, H., Chen, B., Zhong, S., Hebert, M., and Zhao, D. (2020b). Constrained model-based reinforcement learning with robust cross-entropy method. *arXiv preprint arXiv:2010.07968*.

Luo, Y. and Ma, T. (2021). Learning barrier certificates: Towards safe reinforcement learning with zero training-time violations. *Advances in Neural Information Processing Systems*, 34.

Machado, G. R., Silva, E., and Goldschmidt, R. R. (2021). Adversarial machine learning in image classification: A survey toward the defender's perspective. *ACM Computing Surveys (CSUR)*, 55(1):1–38.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*.

Mguni, D., Islam, U., Sun, Y., Zhang, X., Jennings, J., Sootla, A., Yu, C., Wang, Z., Wang, J., and Yang, Y. (2021). Desta: A framework for safe reinforcement learning with markov games of intervention. *arXiv preprint arXiv:2110.14468*.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

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.

Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. G. (2016). Safe and efficient off-policy reinforcement learning. *arXiv preprint arXiv:1606.02647*.

Muratore, F., Treede, F., Gienger, M., and Peters, J. (2018). Domain randomization for simulation-based policy optimization with transferability assessment. In *Conference on Robot Learning*, pages 700–713. PMLR.

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

Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. (2017). Robust adversarial reinforcement learning. In *International Conference on Machine Learning*, pages 2817–2826. PMLR.

Ray, A., Achiam, J., and Amodei, D. (2019). Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 7.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR.

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

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48.

Sootla, A., Cowen-Rivers, A. I., Jafferjee, T., Wang, Z., Mguni, D. H., Wang, J., and Ammar, H. (2022). Sauté rl: Almost surely safe reinforcement learning using state augmentation. In *International Conference on Machine Learning*, pages 20423–20443. PMLR.

Stooke, A., Achiam, J., and Abbeel, P. (2020). Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pages 9133–9143. 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.

Volpi, R., Namkoong, H., Sener, O., Duchi, J. C., Murino, V., and Savarese, S. (2018). Generalizing to unseen domains via adversarial data augmentation. *Advances in neural information processing systems*, 31.

Wang, J., Liu, Y., and Li, B. (2020). Reinforcement learning with perturbed rewards. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6202–6209.

Xu, M., Liu, Z., Huang, P., Ding, W., Cen, Z., Li, B., and Zhao, D. (2022). Trustworthy reinforcement learning against intrinsic vulnerabilities: Robustness, safety, and generalizability. *arXiv preprint arXiv:2209.08025*.

Yang, Q., Simão, T. D., Tindemans, S. H., and Spaan, M. T. (2021). Wcsac: Worst-case soft actor critic for safety-constrained reinforcement learning. In *AAAI*, pages 10639–10646.

Yang, T.-Y., Rosca, J., Narasimhan, K., and Ramadge, P. J. (2020). Projection-based constrained policy optimization. *arXiv preprint arXiv:2010.03152*.

Yu, M., Yang, Z., Kolar, M., and Wang, Z. (2019). Convergent policy optimization for safe reinforcement learning. *arXiv preprint arXiv:1910.12156*.

Zhang, H., Chen, H., Xiao, C., Li, B., Liu, M., Boning, D., and Hsieh, C.-J. (2020a). Robust deep reinforcement learning against adversarial perturbations on state observations. *Advances in Neural Information Processing Systems*, 33:21024–21037.

Zhang, H., Chen, H., Xiao, C., Li, B., Liu, M., Boning, D., and Hsieh, C.-J. (2020b). Robust deep reinforcement learning against adversarial perturbations on state observations. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21024–21037. Curran Associates, Inc.

Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., and Jordan, M. (2019). Theoretically principled trade-off between robustness and accuracy. In *International conference on machine learning*, pages 7472–7482. PMLR.

Zhang, Y., Vuong, Q., and Ross, K. (2020c). First order constrained optimization in policy space. *Advances in Neural Information Processing Systems*.

Table of Contents

A PROOFS AND DISCUSSIONS **12**

A.1 Proof of Theorem 1 - Worst-case Cost Bound for Vanilla M-step Under Attacks . . . 12

A.2 Proof of Theorem 2 - Worst-case Cost Bound for Robust M-step Under Attacks . . . 13

A.3 Proof of Proposition 2 - Multi-mapping From Corrupted Policy to Benign Policy Given a Fixed Adversary 14

A.4 Precise Statement and Proofs of Multi-mapping Under MC or MR Adversary . . . 14

A.5 Derivation of Equation 4 - E-step Optimization Objective . . . 15

A.6 Proof of Proposition 1 - Optimal Variational Distribution . . . 15

A.7 Derivation of Equation 7 - M-step Optimization Objective . . . 16

B EXPERIMENT DETAILS AND MORE RESULTS **18**

B.1 SAFER Implementation Details and Training Tricks . . . 18

B.2 Baseline Implementation Details . . . 19

B.3 Experiment Setting and Hyper-parameters . . . 22

B.4 Additional Experiment Results . . . 23

A. PROOFS AND DISCUSSIONS

A.1. Proof of Theorem 1 - Worst-case Cost Bound for Vanilla M-step Under Attacks

Remind that we denote $\mathcal{S}_c$ as the set of unsafe states that have non-zero cost: $\mathcal{S}_c := \{s' \in \mathcal{S} : c(s,a,s') > 0\}$, p_s as the maximum probability of entering unsafe states from state s, i.e., $p_s = \max_a \sum_{s' \in \mathcal{S}_c} p(s'|s,a)$; and $A_c^\pi(s,a) = Q_c^\pi(s,a) - V_c^\pi(s)$ represents the cost advantage function of policy π. We first introduce two lemmas for following proofs.

Lemma 1 (Achiam et al. (2017) Corollary 2)**.** *For any policies π, π', the following bound holds:*

$$V_c^{\pi'}(\mu_0) - V_c^\pi(\mu_0) \leq \frac{1}{1-\gamma}\mathbb{E}_{s\sim d_\pi, a\sim\pi'}\left[A_c^\pi(s,a) + \frac{2\gamma\alpha_c^{\pi'}}{1-\gamma}D_{TV}[\pi'(\cdot|s)\|\pi(\cdot|s)]\right], \tag{11}$$

where $\alpha_c^{\pi'} = \max_s |\mathbb{E}_{a\sim\pi'} A_c^\pi(s,a)|$.

Lemma 2. *Given any policy π, π', the advantage function of corrupted policy $\pi \circ \nu$ is bounded by*

$$\mathbb{E}_{a\sim\pi'} A_c^\pi(s,a) \leq 2D_{TV}[\pi'(\cdot|s)\|\pi(\cdot|s)] \cdot \left[c(s,a) + \gamma \max_{s'} V_c^\pi(s')\right]. \tag{12}$$

Proof.

$$\begin{aligned}
\mathbb{E}_{a\sim\pi'} A_c^\pi(s,a,s') &= \mathbb{E}_{a\sim\pi', s\sim p(\cdot|s,a)}[c(s,a,s') + \gamma V_c^\pi(s') - V_c^\pi(s)] && (13)\\
&= \mathbb{E}_{a\sim\pi', s'\sim p(\cdot|s,a)}[c(s,a,s') + \gamma V_c^\pi(s')] - V_c^\pi(s) && (14)\\
&= \mathbb{E}_{a\sim\pi', s\sim p(\cdot|s,a)}[c(s,a,s') + \gamma V_c^\pi(s')] - \mathbb{E}_{a\sim\pi, s'\sim p(\cdot|s,a)}[c(s,a,s') + \gamma V_c^\pi(s')] && (15)\\
&= \sum_a (\pi'(a|s) - \pi(a|s))\mathbb{E}_{s'\sim p(\cdot|s,a)}[c(s,a,s') + \gamma V_c^\pi(s')] && (16)\\
&\leq \sum_a |\pi'(a|s) - \pi(a|s)| \cdot \max_{a,s'}[p_s c(s,a,s') + \gamma V_c^\pi(s')] && (17)\\
&= 2D_{TV}[\pi'(\cdot|s)\|\pi(\cdot|s)] \cdot \left[\max_{a,s'} p_s c(s,a,s') + \gamma \max_{s'} V_c^\pi(s')\right] && (18)
\end{aligned}$$

In practice, since the maximum one step cost $\max c(s,a,s') = C_m, V_c^\pi(s) \leq \frac{C_m}{1-\gamma}$, we further have

$$\begin{aligned}
\max_s |\mathbb{E}_{a\sim\pi'} A_c^\pi(s,a)| &\leq 2\max_s D_{TV}[\pi'(\cdot|s)\|\pi(\cdot|s)] \cdot \left[\max_s p_s C_m + \gamma\frac{C_m}{1-\gamma}\right] && (19)\\
&= 2\max_s D_{TV}[\pi'(\cdot|s)\|\pi(\cdot|s)] \cdot \left(\max_s p_s + \frac{\gamma}{1-\gamma}\right) C_m. && (20)
\end{aligned}$$

□

Now we first consider the cost bound of **benign** policy after vanilla M step $V_c^{\pi_{i+1}^{(V)}}$. By applying Lemma 1, we have

$$V_c^{\pi_{i+1}^{(V)}}(\mu_0) \leq V_c^{\pi_i}(\mu_0) + \left(\frac{1}{1-\gamma} + \frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{s\sim d_\pi} D_{TV}[\pi_{i+1}^{(V)}(\cdot|s)\|\pi_i(\cdot|s)]\right)\alpha_c^{\pi_{i+1}^{(V)}} \tag{21}$$

$$\leq V_c^{\pi_i}(\mu_0) + \left(\frac{1}{1-\gamma} + \frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{s\sim d_\pi}\sqrt{\frac{1}{2}D_{KL}[\pi_{i+1}^{(V)}(\cdot|s)\|\pi_i(\cdot|s)]}\right)\alpha_c^{\pi_{i+1}^{(V)}}, \text{ by } D_{TV} \leq \sqrt{\frac{D_{KL}}{2}}, \tag{22}$$

$$\leq V_c^{\pi_i}(\mu_0) + \left(\frac{1}{1-\gamma} + \frac{2\gamma}{(1-\gamma)^2}\sqrt{\frac{1}{2}\mathbb{E}_{s\sim d_\pi} D_{KL}[\pi_{i+1}^{(V)}(\cdot|s)\|\pi_i(\cdot|s)]}\right)\alpha_c^{\pi_{i+1}^{(V)}}, \tag{23}$$

$$\leq V_c^{\pi_i}(\mu_0) + \left(\frac{1}{1-\gamma} + \frac{2\gamma}{(1-\gamma)^2}\sqrt{\frac{1}{2}\xi}\right)\alpha_c^{\pi_{i+1}^{(V)}}, \text{ by constraint of vanilla M-step,} \tag{24}$$

$$= \kappa + \left(\frac{1}{1-\gamma} + \frac{\sqrt{2\xi}\gamma}{(1-\gamma)^2}\right)\alpha_c^{\pi_{i+1}^{(V)}}, \ \pi_i \text{ is feasible,} \tag{25}$$

where $\alpha_c^{\pi_{i+1}^{(V)}} = \max_s |\mathbb{E}_{a\sim\pi_{i+1}^{(V)}} A_c^{\pi_i}(s,a)|$.

Then by Lemma 2, we have

$$V_c^{\pi_{i+1}^{(V)}}(\mu_0) \leq \kappa + \left(\frac{2d_{i+1}^{(V)}}{1-\gamma} + \frac{2\sqrt{2\xi}\gamma d_{i+1}^{(V)}}{(1-\gamma)^2}\right)\left(\max_s p_s + \frac{\gamma}{1-\gamma}\right)C_m, \tag{26}$$

where $d_{i+1}^{(V)} = \max_s D_{TV}[\pi_{i+1}^{(V)}(\cdot|s)\|\pi_i(\cdot|s)]$ denotes the maximum TV distance between the policies before and after M-step.

Second, we consider the cost bound of **corrupted** policy after vanilla M step $V_c^{\pi_{i+1}^{(V)}\circ\nu}$. Since we assume the L-Lipschitz continuity of policy π and constrain perturbation within a ℓ_p-ball, apply the Theorem 3 in (Liu et al., 2022c) and we have

$$V_c^{\pi_{i+1}^{(V)}\circ\nu}(\mu_0) - V_c^{\pi_{i+1}^{(V)}}(\mu_0) \leq 2L\epsilon C_m\left(\frac{1}{1-\gamma} + \frac{4\gamma L\epsilon}{(1-\gamma)^2}\right)\left(\max_s p_s + \frac{\gamma}{1-\gamma}\right). \tag{27}$$

Therefore, combine Eq.(26) & (27) and we have the cost bound of corrupted policy:

$$V_c^{\pi_{i+1}^{(V)}\circ\nu}(\mu_0) \leq \kappa + 2C_m\left(\frac{d_{i+1}^{(V)} + L\epsilon}{1-\gamma} + \frac{\sqrt{2\xi}\gamma d_{i+1}^{(V)} + 4\gamma L^2\epsilon^2}{(1-\gamma)^2}\right)\left(\max_s p_s + \frac{\gamma}{1-\gamma}\right). \tag{28}$$

A.2. Proof of Theorem 2 - Worst-case Cost Bound for Robust M-step Under Attacks

In robust M-step, we explicitly limit the discrepancy between corrupted policy $\pi_{i+1}^{(R)} \circ \nu$ and the last benign policy π_i. Therefore we can obtain safety performance under attacks by Lemma 1 & 2 directly:

$$V_c^{\pi_{i+1}^{(R)}\circ\nu}(\mu_0) \leq \kappa + \left(\frac{2d_{i+1}^{(R)}}{1-\gamma} + \frac{2\sqrt{2\tilde{\xi}}\gamma d_{i+1}^{(R)}}{(1-\gamma)^2}\right)\left(\max_s p_s + \frac{\gamma}{1-\gamma}\right)C_m, \tag{29}$$

where $d_{i+1}^{(R)} = \max_s D_{TV}[\pi_{i+1}^{(R)}(\cdot|\nu(s))\|\pi_i(\cdot|s)]$ is the maximum TV distance between the corrupted policy after robust M-step and benign policy before M-step.

We can find that the worse-case safety performance of the corrupted policy $\pi_{i+1}^{(R)} \circ \nu$ under the strongest attacks is similar to the benign policy $\pi_{i+1}^{(V)}$ in the natural environment, which shows the advantage of the robust M-step update.

A.3. Proof of Proposition 2 - Multi-mapping From Corrupted Policy to Benign Policy Given a Fixed Adversary

Recall that Proposition 2 said given a fixed adversary ν and suppose there exists $s_1, s_2 \in \mathcal{S}$ such that $\nu(s_1) = \nu(s_2)$, then there exists two natural policies $\pi_1, \pi_2 \in \Pi$ such that $\pi_1 \circ \nu \triangleq \pi_2 \circ \nu$. We provide the proof as follows.

Proof. We can construct two policies $\pi_1, \pi_2 \in \Pi$ such that they are the same in all states except s_1, s_2. Formally, we have:

$$\pi_1(\cdot|s) \triangleq \pi_2(\cdot|s), \quad \forall s \in \mathcal{S} \setminus \{s_1, s_2\}; \quad \pi_1(\cdot|s_1) \neq \pi_2(\cdot|s_1) \quad \text{or} \quad \pi_1(\cdot|s_2) \neq \pi_2(\cdot|s_2). \tag{30}$$

Then π_1, π_2 are two different benign policies. In addition, we can obtain:

$$\pi_1(\cdot|s) \circ \nu = \pi_1(\cdot|\nu(s)) \triangleq \pi_2(\cdot|\nu(s)) = \pi_2(\cdot|s) \circ \nu, \quad \forall s \in \mathcal{S} \setminus \{s_1, s_2\}. \tag{31}$$

Based on our assumption, $\nu(s_1) = \nu(s_2)$, we have:

$$\pi_1(\cdot|s) \circ \nu \triangleq \pi_2(\cdot|s) \circ \nu, \quad \forall s \in \{s_1, s_2\}. \tag{32}$$

Therefore, the two corrupted policies are identical for every state: $\pi_1(\cdot|s) \circ \nu \triangleq \pi_2(\cdot|s) \circ \nu, \quad \forall s \in \mathcal{S}$. □

A.4. Precise Statement and Proofs of Multi-mapping Under MC or MR Adversary

Proposition 2 considers the case with a fixed deterministic adversary. However, in practical training for SAFER, we adopt a stronger MC adversary that is dependent on the policy π as well as its value function. Therefore, we aim to study whether the multiple mapping phenomenon still holds for policy-dependent adversaries.

We denote $\mathcal{V}^\epsilon$ be the set of adversaries that distort all states to the corrupted states within ϵ-size ℓ_p-ball around original states, i.e., $\mathcal{V}^\epsilon = \{\nu : \mathcal{S} \to \mathcal{S} | \forall s, \nu(s) \in B_p^\epsilon(s)\}$, then we have following formal theorem.

Theorem 3. *Let ν be the MC (or MR) adversary for policy π within $\mathcal{V}^\epsilon$. Suppose for a given π, its MC (or MR) adversary ν within $\mathcal{V}^\epsilon$ is also the MC (or MR) adversary within $\mathcal{V}^{(1+\Delta)\epsilon}$ for any $\Delta > 0$, then there exists at least one policy $\tilde{\pi} \neq \pi$ and its corresponding MC (or MR) adversary $\tilde{\nu}$ within $\mathcal{V}^\epsilon$, such that they share the same corrupted policy under MC (or MR) attack, i.e., $\pi(a|\nu(s)) = \tilde{\pi}(a|\tilde{\nu}(s)), \forall s, a$.*

Proof. We will take maximum-reward (MR) adversary as an example and all the proofs can be extended to minimum-cost (MC) adversary as well.

According to the assumption, given the policy π, the MR adversary within $\mathcal{V}^\epsilon$ satisfies

$$\nu_{\text{MR}} \in \mathcal{V}^\epsilon, \quad \nu_{\text{MR}} = \operatorname*{arg\,max}_{\nu \in \mathcal{V}^{(1+\Delta)\epsilon}} V_r^{\pi \circ \nu}(\mu_0) \tag{33}$$

Since $\Delta > 0$, then there exists a ξ s.t. $\xi^p + \epsilon^p < (1+\Delta)^p \epsilon^p$.

First, we will prove that for any perturbation functions $\mu_1 \in \mathcal{V}^\xi, \mu_2 \in \mathcal{V}^\epsilon$, the composition function $\mu_1 \circ \mu_2(s) = \mu_1(\mu_2(s))$ satisfies that $\mu_1 \circ \mu_2 \in \mathcal{V}^{(1+\Delta)\epsilon}$. For any state s, since $\mu_1 \in \mathcal{V}^\xi, \mu_2 \in \mathcal{V}^\epsilon$, then we have $\|\mu_1(s) - s\|_p \leq \xi, \|\mu_2(s) - s\|_p \leq \epsilon$. Therefore,

$$\|\mu_1(\mu_2(s)) - \mu_2(s)\|_p \leq \xi \tag{34}$$

$$\Rightarrow \|\mu_1(\mu_2(s)) - s\|_p \leq \left(\|\mu_1(\mu_2(s)) - \mu_2(s)\|_p^p + \|\mu_2(s) - s\|_p^p\right)^{1/p} \tag{35}$$

$$\leq (\xi^p + \epsilon^p)^{1/p} < (1+\Delta)^p \epsilon^p, \tag{36}$$

and thus $\mu_1 \circ \mu_2 \in \mathcal{V}^{(1+\Delta)\epsilon}$. Furthermore, we can find that there exists at least one pair (μ_1^*, μ_2^*) s.t. $\mu_1^* \circ \mu_2^* = \nu_{\text{MR}}$.

Second, let $\tilde{\nu}_{\text{MR}}$ be the MR adversary for $\tilde{\pi} := \pi \circ \mu_1^*$ within $\mathcal{V}^\epsilon$ and we will prove $\mu_2^* = \tilde{\nu}_{\text{MR}}$. According to the definition, we have

$$V_r^{\pi \circ \mu_{\text{MR}}}(\mu_0) = V_r^{\pi \circ \mu_1^* \circ \mu_2^*}(\mu_0) \leq \max_{\mu_2 \in \mathcal{V}^\epsilon} V_r^{\pi \circ \mu_1^* \circ \mu_2}(\mu_0) = V_r^{\pi \circ \mu_1^* \circ \tilde{\nu}_{\text{MR}}}(\mu_0). \tag{37}$$

Meanwhile, since $\mu_1 \circ \mu_2 \in \mathcal{V}^{(1+\Delta)\epsilon}$, we have

$$V_r^{\pi\circ\mu_1^*\circ\tilde{\nu}_{\text{MR}}}(\mu_0) \leq \max_{\mu_1\in\mathcal{V}^{\xi},\mu_2\in\mathcal{V}^{\epsilon}} V_r^{\pi\circ\mu_1\circ\mu_2}(\mu_0) \leq \max_{\nu\in\mathcal{V}^{(1+\Delta)\epsilon}} V_r^{\pi\circ\nu}(\mu_0) = V_r^{\pi\circ\nu_{\text{MR}}}(\mu_0). \tag{38}$$

Therefore, combine the above two equations together and we can obtain

$$V_r^{\pi\circ\nu_{\text{MR}}}(\mu_0) \leq V_r^{\pi\circ\mu_1^*\circ\tilde{\nu}_{\text{MR}}}(\mu_0) \leq V_r^{\pi\circ\nu_{\text{MR}}}(\mu_0). \tag{39}$$

The equality holds if and only if $\tilde{\nu}_{\text{MR}} = \mu_2^*$, i.e., $\nu_{\text{MR}} = \mu_1^* \circ \tilde{\nu}_{\text{MR}}$. Therefore, there exists a policy $\tilde{\pi} \neq \pi$ s.t. they share the same corrupted policy $\pi \circ \nu_{\text{MR}} = \tilde{\pi} \circ \tilde{\nu}_{\text{MR}}$ under their corresponding MR attacks. □

Theorem 3 together with Proposition 2 show the shrinkage from benign policy space to corrupted policy space, which makes it difficult to restore the benign policy if we only consider updating corrupted policy in the robust M-step. Therefore, it is significant to incorporate vanilla M-step to solve the optimal benign policy given its corrupted policy, which is also validated by our experiments.

A.5. Derivation of Equation 4 - E-step Optimization Objective

The E-step at the i-th iteration aims to find the optimal variational distribution $q \in \Pi_{\mathcal{M}}^{\kappa}$ that maximizes the ELBO while satisfying the safety constraint, which can be formulated as:

$$\begin{aligned} \max_q \quad & \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q(\cdot|s)}\big[Q_r^{\pi_{\theta_i}}(s,a)\big] - \alpha D_{\text{KL}}(q\|\pi_{\theta_i})\Big] + \log p(\theta), \\ s.t. \quad & \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q(\cdot|s)}\big[Q_c^{\pi_{\theta_i}}(s,a)\big]\Big] \leq \kappa \end{aligned} \tag{40}$$

Solving the E-step (40) could be regarded as a KL-regularized constrained optimization problem. However, since the expected reward return term $\mathbb{E}_{q(\cdot|s)}\big[Q_r^{\pi_{\theta_i}}(s,a)\big]$ could be on an arbitrary scale, it is hard to choose a proper penalty coefficient α of the KL regularizer for different CMDP settings. Therefore, we impose a hard constraint δ on the KL divergence between the non-parametric distribution $q(a|s)$ that to be optimized and the parametrized policy $\pi_{\theta_i}(a|s)$: $\mathbb{E}_{\rho_q}\Big[D_{\text{KL}}(q(a|s)\|\pi_{\theta_i})\Big] \leq \delta$. In addition, the last term $\log p(\theta)$ in the object function is a constant to q, which can be omitted in the optimization process. Then the optimization problem (40) yields to (4).

A.6. Proof of Proposition 1 - Optimal Variational Distribution

Since $q(a|s)$ is a distribution function, there indeed exists another constraint for $q(a|s)$: $\int q(a|s)da = 1, \forall s \sim \rho_q$. With the definition of expectation, the objective in E-step can be re-written as:

$$\begin{aligned} \max_q \quad & \int \rho_q(s) \int q(a|s) Q_r^{\pi_{\theta_i}}(s,a) da ds, \\ s.t. \quad & \int \rho_q(s) \int q(a|s) Q_c^{\pi_{\theta_i}}(s,a) da ds \leq \kappa; \\ & \int \rho_q(s) \int q(a|s) \log \frac{q(a|s)}{\pi_{\theta_i}(a|s)} da ds \leq \delta; \\ & \int q(a|s) da = 1, \quad \forall s \sim \rho_q, \end{aligned} \tag{41}$$

where $\rho_q(s)$ is the stationary state distribution induced by $q(a|s)$ and ρ_0. Then we prove the optimal variational distribution analytical form and its dual function in Proposition 1.

Proof. Since the objective is linear and all constraints are convex (note that KL is convex) w.r.t q, this constrained optimization problem is convex. Then we obtain the equivalent dual problem:

$$\min_{\lambda,\eta} \max_q L(q,\lambda,\eta), \tag{42}$$

where λ, η are the Lagrange multipliers for the constraints, and L is the equivalent Lagrangian function:

$$L(q, \lambda, \eta, \gamma) = \int \rho_q(s) \int q(a|s) Q_r^{\pi_{\theta_i}}(s,a) dads \tag{43}$$

$$+ \lambda \left(\kappa - \int \rho_q(s) \int q(a|s) Q_c^{\pi_{\theta_i}}(s,a) dads \right) \tag{44}$$

$$+ \eta \left(\delta - \int \rho_q(s) \int q(a|s) \log \frac{q(a|s)}{\pi_{\theta_i}(a|s)} dads \right) \tag{45}$$

$$+ \gamma \left(1 - \int \rho_q(s) \int q(a|s) dads \right) \tag{46}$$

Take the derivative of Lagrangian function w.r.t q:

$$\frac{\partial L}{\partial q} = Q_r^{\pi_{\theta_i}}(s,a) - \lambda Q_c^{\pi_{\theta_i}}(s,a) - \eta - \gamma - \eta \log \frac{q(a|s)}{\pi_{\theta_i}(a|s)}. \tag{47}$$

Let Eq. (47) be zero, we have the form of the optimal q distribution:

$$q^*(a|s) = \pi_{\theta_i}(a|s) \exp\left(\frac{Q_r^{\pi_{\theta_i}}(s,a) - \lambda Q_c^{\pi_{\theta_i}}(s,a)}{\eta} \right) \exp\left(-\frac{\eta + \gamma}{\eta} \right), \tag{48}$$

where $\exp\left(-\frac{\eta+\gamma}{\eta}\right)$ could be viewed as a normalizer for $q(a|s)$ since it is a constant that is independent of q. Thus, we obtain the following form of the normalizer by integrating the optimal q:

$$\exp\left(\frac{\eta + \gamma}{\eta} \right) = \int \pi_{\theta_i}(a|s) \exp\left(\frac{Q_r^{\pi_{\theta_i}}(s,a) - \lambda Q_c^{\pi_{\theta_i}}(s,a)}{\eta} \right) da, \tag{49}$$

$$\frac{\eta + \gamma}{\eta} = \log \int \pi_{\theta_i}(a|s) \exp\left(\frac{Q_r^{\pi_{\theta_i}}(s,a) - \lambda Q_c^{\pi_{\theta_i}}(s,a)}{\eta} \right) da. \tag{50}$$

Take the optimal q distribution in Equation (48) and $\frac{\eta+\gamma}{\eta}$ in Equation (50) back to the Lagrangian function (46), we can find that most of the terms are cancelled out, and obtain the dual function $g(\eta, \lambda)$,

$$g(\eta, \lambda) = \lambda\kappa + \eta\delta + \eta \int \rho_q(s) \log \int \pi_{\theta_i}(a|s) \exp\left(\frac{Q_r^{\pi_{\theta_i}}(s,a) - \lambda Q_c^{\pi_{\theta_i}}(s,a)}{\eta} \right) dads. \tag{51}$$

The optimal dual variables are calculated by

$$\eta^*, \lambda^* = \arg\min_{\eta,\lambda} g(\eta, \lambda). \tag{52}$$

□

Note that the dual function g is a convex function, and is strongly convex in many cases, see Appendix A.3 in (Liu et al., 2022b) for details.

A.7. Derivation of Equation 7 - M-step Optimization Objective

As shown in section 4.3, given the optimal variational distribution q_i^* from the E-step, the M-step objective is:

$$\theta_{i+1} = \arg\max_{\theta} \mathbb{E}_{\rho_q}\left[\alpha \mathbb{E}_{q_i^*(\cdot|s)}\left[\log \pi_\theta(a|s) \right] \right] + \log p(\theta) \tag{53}$$

which is a Maximum A-Posteriori (MAP) problem (Abdolmaleki et al., 2018b;a). Consider a Gaussian prior around the old policy parameter θ_i, we have

$$\theta \sim \mathcal{N}(\theta_i, \frac{F_{\theta_i}}{\alpha\beta}) \tag{54}$$

where F_{θ_i} is the Fisher information matrix and β is a positive constant. With the Gaussian prior, the objective (53) becomes

$$\begin{aligned}\theta_{i+1} &= \arg\max_{\theta} \alpha \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q_i^*(\cdot|s)}\big[\log \pi_\theta(a|s)\big]\Big] - \alpha\beta(\theta-\theta_i)^T F_{\theta_i}^{-1}(\theta-\theta_i) \\ &= \arg\max_{\theta} \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q_i^*(\cdot|s)}\big[\log \pi_\theta(a|s)\big]\Big] - \beta(\theta-\theta_i)^T F_{\theta_i}^{-1}(\theta-\theta_i)\end{aligned} \tag{55}$$

where we could observe that $(\theta-\theta_i)^T F_{\theta_i}^{-1}(\theta-\theta_i)$ is the second order Taylor expansion of $\mathbb{E}_{\rho_q}\big[D_{\mathrm{KL}}(\pi_{\theta_i}(a|s)\|\pi_\theta(a|s))\big]$. Thus, we could generalize the above objective to the KL-regularized one:

$$\max_{\theta} \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q_i^*(\cdot|s)}\big[\log \pi_\theta(a|s)\big] - \beta D_{\mathrm{KL}}(\pi_{\theta_i}\|\pi_\theta)\Big]. \tag{56}$$

Similar to the E-step, we could convert the soft KL regularizer to a hard KL constraint:

$$\begin{aligned}\max_{\theta} \quad & \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q_i^*(\cdot|s)}\big[\log \pi_\theta(a|s)\big]\Big] \\ s.t. \quad & \mathbb{E}_{\rho_q}\big[D_{\mathrm{KL}}(\pi_{\theta_i}(a|s)\|\pi_\theta(a|s))\big] \le \xi.\end{aligned} \tag{57}$$

B. EXPERIMENT DETAILS AND MORE RESULTS

B.1. SAFER Implementation Details and Training Tricks

As introduced in the method section, SAFER consists of a constrained E-step and a robust M-step. The E-step step aims to find the optimal variational distribution that maximizes the reward return while satisfying the safety constraint. This step can be written as a constrained optimization problem, which has a closed-form solution and can be efficiently solved by convex optimization. The robust M-step has two components: a vanilla M-step that aims to fit the variational distribution obtained from the E-step using a parametrized policy, such as neural networks (NNs), and then generalize beyond the state-action samples used for training; and an adversarial training step that aims to improve the policy robustness by optimizing toward the worst-case perturbations. The complete data-flow of SAFER is shown in Figure. 1.

Due to the page limit, we omit the implementation details of SAFER in the main content. We will present the full algorithm and some implementation tricks in this section. Without otherwise statements, the critics' and policies' parametrization is assumed to be neural networks (NNs), while we believe other parametrization forms should also work in practice.

Critics update. Denote ϕ_r as the parameters for the task reward critic Q_r, and ϕ_c as the parameters for the constraint violation cost critic Q_c. Similar to many other off-policy algorithms (Lillicrap et al., 2015), we use a target network for each critic and the polyak smoothing trick to stabilize the training. Other off-policy critic's training methods, such as Re-trace (Munos et al., 2016), could also be easily incorporated with the SAFER training framework. Denote ϕ'_r as the parameters for the **target** reward critic Q'_r, and ϕ'_c as the parameters for the **target** cost critic Q'_c. Define $\mathcal{D}$ as the replay buffer and (s, a, s', r, c) as the state, action, next state, reward, and cost respectively. The critics are updated by minimizing the following mean-squared Bellman error (MSBE):

$$L(\phi_r) = \mathbb{E}_{(s,a,s',r,c)\sim\mathcal{D}}\Big[\,(Q_r(s,a) - (r + \gamma\mathbb{E}_{a'\sim\pi}[Q'_r(s',a')]))^2\,\Big] \tag{58}$$

$$L(\phi_c) = \mathbb{E}_{(s,a,s',r,c)\sim\mathcal{D}}\Big[\,(Q_c(s,a) - (c + \gamma\mathbb{E}_{a'\sim\pi}[Q'_c(s',a')]))^2\,\Big]. \tag{59}$$

Denote α_c as the critics' learning rate, we have the following updating equations:

$$\phi_r \leftarrow \phi_r - \alpha_c\nabla_{\phi_r}L(\phi_r), \quad \phi_c \leftarrow \phi_c - \alpha_c\nabla_{\phi_c}L(\phi_c). \tag{60}$$

We use the polyak averaging trick to update the critics with a weight parameter $\rho \in (0, 1)$:

$$\phi'_r = \rho\phi'_r + (1-\rho)\phi_r \quad \phi'_c = \rho\phi'_c + (1-\rho)\phi_c. \tag{61}$$

Robust M-step constrained supervised learning. Recall that the robust M-step has a vanilla M-step (Eq. (7)) and an adversarial training step (Eq. (8)), which could be written as the Lagrangian function with the Lagrangian multipliers $\beta, \tilde{\beta}$:

$$\begin{aligned} L(\theta,\beta,\tilde{\beta}) =& \mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q_i^*(\cdot|s)}\big[\log\pi_\theta(a|s)\big] + \beta\Big(\xi - D_{\mathrm{KL}}\big(\pi_{\theta_i}(a|s)\|\pi_\theta(a|s)\big)\Big)\Big] \\ &+\mathbb{E}_{\rho_q}\Big[\mathbb{E}_{q_i^*(\cdot|s)}\big[\log\pi_\theta(a|\nu(s))\big] + \tilde{\beta}\Big(\tilde{\xi} - D_{\mathrm{KL}}\big(\pi_{\theta_i}(a|s)\|\pi_\theta(a|\nu(s))\big)\Big)\Big]. \end{aligned} \tag{62}$$

In practical implementation, we can also replace the adversarial KL term w.r.t the corrupted data $D_{\mathrm{KL}}\big(\pi_{\theta_i}(a|s)\|\pi_\theta(a|\nu(s)\big)$ to $D_{\mathrm{KL}}\big(\pi_\theta(a|s)\|\pi_\theta(a|\nu(s)\big)$, but stop the gradient from the benign data path $\pi_\theta(a|s)$. Namely, we only compute the gradient from the corrupted state path $\pi_\theta(a|\nu(s))$ to pursue training stability.

By performing the gradient descend ascend algorithm over the dual variables $\beta, \tilde{\beta}$ and the policy parameters θ in Eq. (62) iteratively yields the KL-constrained policy improvement in a constrained supervised learning fashion:

$$\max_{\theta}\ \min_{\beta>0,\tilde{\beta}>0}\quad L(\theta,\beta,\tilde{\beta}). \tag{63}$$

More specifically, denote $\alpha_\beta, \alpha_{\tilde{\beta}}, \alpha_\theta$ as the learning rate for $\beta, \tilde{\beta}, \theta$ respectively, we have the following updating equations:

$$\beta \leftarrow \beta - \alpha_\beta\frac{\partial L(\theta,\beta,\tilde{\beta})}{\partial\beta} = \beta - \alpha_\beta\left(\xi - D_{\mathrm{KL}}\Big(\pi_{\theta_i}(a|s)\|\pi_\theta(a|s)\Big)\right) \tag{64}$$

$$\tilde{\beta} \leftarrow \tilde{\beta} - \alpha_{\tilde{\beta}}\frac{\partial L(\theta,\beta,\tilde{\beta})}{\partial\tilde{\beta}} = \tilde{\beta} - \alpha_{\tilde{\beta}}\left(\tilde{\xi} - D_{\mathrm{KL}}\Big(\pi_{\theta_i}(a|s)\|\pi_\theta(a|\nu(s))\Big)\right) \tag{65}$$

$$\theta \leftarrow \theta - \alpha_\theta\frac{\partial L(\theta,\beta,\tilde{\beta})}{\partial\theta}. \tag{66}$$

Perturbation set B_p^ϵ range schedule. We gradually increase the perturbation range size from 0 to ϵ during training to make the training more stable (Zhang et al., 2020a). Suppose the adversarial training starting epoch is T_s the increasing epoch is T_i, then the perturbation range at epoch t is:

$$\epsilon_t = \min\{\epsilon, \epsilon \times \frac{\max\{0, t - T_s\}}{T_i}\}. \tag{67}$$

Constraint threshold transform in the off-policy setting. Note that for off-policy methods, we need to convert the episodic-wise constraint violation threshold to a state-wise threshold for the Q_c functions. Denote H as the episode length, the target cost limit for one episode is κ_H. Denote the discounting factor as γ. Then, if we assume that at each time step we have an equal probability to violate the constraint, the target constraint value κ for safety critic $Q_c^{\pi_\theta}$ could be approximated by (Ray et al., 2019):

$$\kappa = \kappa_H \times \frac{1 - \gamma^H}{H(1 - \gamma)}$$

The converted threshold κ will be used as one of the constraint thresholds in the E-step:

$$\int \pi(a|s) Q_c^{\pi_{\theta_i}}(s, a) \leq \kappa, \quad \forall s, a$$

With all the implementation tricks mentioned above, we present the full SAFER algorithm:

Algorithm 2 SAFER Algorithm

Input: rollouts number B, robust M-step iteration number M, batch size B, particle size K, discount factor γ, polyak weight ρ, critics learning rate α_c, policy learning rate α_θ, dual variables' learning rates $\alpha_\beta, \alpha_{\tilde{\beta}}$, thresholds $\xi, \tilde{\xi}$
Output: policy π_θ

Initialize policy parameters θ, θ', critics parameters $\phi_r, \phi_r', \phi_c, \phi_c'$ and replay buffer $\mathcal{D} = \{\}$
for each training iteration $t = 1, ..., T$ **do**
  Rollout B trajectories by π_θ from the environment $\mathcal{D} = \mathcal{D} \cup \{(s, a, s', r, c)\}$
  Sample N transitions $\{(s_n, a_n, s_{n+1}, r_n, c_n)_{n=1,...,N}\}$ from the replay buffer $\mathcal{D}$
  Update the perturbation range ϵ_t by Eq. (67).
  ▷ *Constrained E-step begins*
  Update reward critic by Eq. (58): $\phi_r \leftarrow \phi_r - \alpha_c \nabla_{\phi_r} L(\phi_r)$
  Update cost critic by Eq. (59): $\phi_c \leftarrow \phi_c - \alpha_c \nabla_{\phi_c} L(\phi_c)$
  for $n = 1, ..., N$ **do**
    Sample K actions $\{a_1, ..., a_K\}$ for s_n
    Compute $\{Q_r^{\theta_i}(s_n, a_k), Q_c^{\theta_i}(s_n, a_k); k = 1, ..., K\}$
  end for
  Compute optimal dual variables η^*, λ^* by solving the convex optimization problem (6)
  Compute the optimal variational distribution for each state $\{q^*(\cdot|s_n); n = 1, ..., N\}$ by Eq. (5)
  Normalize the variational distribution $\{q^*(\cdot|s_n); n = 1, ..., N\}$ for each state
  ▷ *Robust M-step begins*
  Compute the corrupted states $\{\nu_{\text{MC}}(s_1), ..., \nu_{\text{MC}}(s_N)\}$ with the perturbation range size ϵ_t.
  for Robust M-step iterations $m = 1, ..., M$ **do**
    Perform one gradient step for β via Eq. (64) and for $\tilde{\beta}$ via Eq. (65)
    Perform one gradient step for policy parameters via Eq. (66): $\theta \leftarrow \theta - \alpha_\theta \frac{\partial L(\theta, \beta, \tilde{\beta})}{\partial \theta}$
  end for
  Polyak averaging target networks by Eq. (61), and update the target policy network $\pi_{\theta_i} \leftarrow \pi_\theta$.
end for

B.2. Baseline Implementation Details

In this section, we first recap the observational adversarial attacker algorithms and then introduce the on-policy baselines and off-policy baselines, respectively.

B.2.1. Observational Adversarial Attackers

In Safe RL settings, observational perturbations weaken the robustness for the safety constraint violations. To introduce the adversary attackers, we first recall the setting of Safe RL under observational perturbations, and then show the attacker methods.

Safe RL under observational perturbations: In realistic scenarios, observational perturbations for safe RL agents could be the noise from the sensing system or the errors from the upstream perception system, which harms the safety. To

improve the robustness with respect to safety constraint satisfaction under such circumstance, a deterministic **observational** adversary $\nu(s) : \mathcal{S} \to \mathcal{S}$ which corrupts the state observation of the agent is implemented during the training process. We denote the corrupted state as $\tilde{s} := \nu(s)$ and the corrupted policy as $\pi \circ \nu := \pi(a|\tilde{s}) = \pi(a|\nu(s))$, as the state is first contaminated by ν and then used by the operator π. Note that the adversary does **not** modify the original CMDP and true states in the environment, but only the input of the agent.

MC and MR Attacker: In the experiments, MC (Maximum Cost) and MR (Maximum reward) are selected to be the attacker. The MC attacker directly maximizes the cost return to obtain the perturbation: $\nu_{\text{MC}} = \arg\max_\nu V_c^{\pi\circ\nu}(\mu_0)$, while the MR attacker maximizes the reward return to make the policy be tempting: $\nu_{\text{MR}} = \arg\max_\nu V_r^{\pi\circ\nu}(\mu_0)$. The details are shown as follows. We use the gradient of the state-action value function $Q(s, a)$ to provide the direction to update states adversarially in K steps ($Q = Q_c^\pi$ for MC, and $Q = Q_r^\pi$ for MR):

$$s^{k+1} = \text{Proj}[s^k - \eta \nabla_{s^k} Q(s^0, \pi(s^k))], k = 0, \ldots, K-1 \tag{68}$$

where Proj$[\cdot]$ is a projection to $B_p^\epsilon(s^0)$ (the perturbation set, i.e., the ℓ_p-ball around the original state), η is the learning rate, and s^0 is the state under attack. Note that we use the gradient of $Q(s^0, \pi(s^k))$ rather than $Q(s^k, \pi(s^k))$ to make the optimization more stable, since the Q function may not generalize well to unseen states in practice. The implementation of MC and MR attacker is shown in algorithm 3.

Algorithm 3 MC and MR attacker

Input: A policy π under attack, corresponding Q networks, initial state s^0, attack steps K, attacker learning rate η, perturbation range ϵ, two thresholds ϵ_Q and ϵ_s for early stopping
Output: An adversarial state $\tilde{s}$

```
for k = 1 to K do
    g^k = ∇_{s^{k-1}} Q(s_0, π(s^{k-1}))
    s^k ← Proj[s^{k-1} − η g^k]
    Compute δQ = |Q(s_0, π(s^k)) − Q(s_0, π(s^{k-1}))| and δs = |s^k − s^{k-1}|
    if δQ < ε_Q and δs < ε_s then
        break for early stopping
    end if
end for
```

B.2.2. On-Policy Baselines

PPO-Lagrangian algorithm. The objective of PPO (clipped) has the form (Schulman et al., 2017):

$$\ell_{ppo} = \min(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)} A^{\pi_{\theta_k}}(s, a), \text{clip}(\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}, 1 - \epsilon_{\text{clip}}, 1 + \epsilon_{\text{clip}}) A^{\pi_{\theta_k}}(s, a)) \tag{69}$$

We use PID Lagrangian (Stooke et al., 2020) that addresses the oscillation and overshoot problem in Lagrangian methods. The loss of the PPO-Lagrangian has the form:

$$\ell_{ppol} = \frac{1}{1+\lambda}(\ell_{ppo} - \lambda A_c^{\pi_{\theta_k}}(s, a)) \tag{70}$$

The Lagrangian multiplier λ is computed by applying feedback control to V_c^π and is determined by K_P, K_I, and K_D that need to be fine-tuned.

SA-PPOL. The SA-PPO-Lagrangian algorithm uses the KL robustness regularizer to robustify the training policy. Choosing different adversaries ν yields different baseline algorithms. The original SA-PPOL method adopts the MAD attacker as shown in Algo.4. We replace it with the MC attacker, which yields the SA-PPOL(MC) baseline.

Algorithm 4 SA-PPO-Lagrangian Algorithm

Input: rollouts T, policy optimization steps M, PPO-Lag loss function $\ell_{ppo}(s, \pi_\theta, r, c)$, adversary function $\nu(s)$
Output: policy π_θ

Initialize policy parameters and critics parameters
for each training iteration **do**
  Rollout T trajectories by π_θ from the environment $\{(s, a, s', r, c)\}_N$
  Compute adversary states $\tilde{s} = \nu(s)$ for the sampled trajectories
  ▷ *Update actors*
  for Optimization steps $m = 1, ..., M$ **do**
    Compute KL robustness regularizer $\tilde{L}_{KL} = D_{\mathrm{KL}}(\pi(s) \| \pi_\theta(\tilde{s}))$, no gradient from $\pi(s)$
    Compute PPO-Lag loss $\ell_{ppol}(s, \pi_\theta, r, c)$ by
    Combine them together with a weight β: $\ell = \ell_{ppol}(s, \pi_\theta, r, c) + \beta \tilde{\ell}_{KL}$
    Update actor $\theta \leftarrow \theta - \alpha \nabla_\theta \ell$
  end for
  ▷ *Update critics*
  Update value function based on samples $\{(s, a, s', r, c)\}_N$
end for

ADV-PPOL. The ADV-PPOL (Liu et al., 2022c) uses an on-policy adversarial training technique and does not use the KL robustness regularizer. This type of method effectively evaluates the safety performance under strong adversarial attacks to improve the robustness and we adopt this algorithm trained with MC attacker as an on-policy baseline. The full algorithm is shonw in Algo.5 and the diagram is shown in Figure. 3a.

Algorithm 5 ADV-PPOL Algorithm

Input: rollouts T, policy optimization steps M, PPO-Lag loss function $\ell_{ppol}(s, \pi_\theta, r, c)$, adversary function $\nu(s)$, policy parameter θ, critic parameter ϕ_r and ϕ_c, target critic parameter ϕ'_r and ϕ'_c
Output: policy π_θ

Initialize policy parameters and critics parameters
for each training iteration **do**
  Rollout T trajectories by $\pi_\theta \circ \nu$ from the environment $\{(\nu(s), \nu(a), \nu(s'), r, c)\}_N$
  ▷ *Update learner*
  for Optimization steps $m = 1, ..., M$ **do**
    ▷ *No KL regularizer!*
    Compute PPO-Lag loss $\ell_{ppol}(\tilde{s}, \pi_\theta, r, c)$ by Eq. (70)
    Update actor $\theta \leftarrow \theta - \alpha \nabla_\theta \ell_{ppo}$
  end for
  Update value function and action value functions based on samples $\{(s, a, s', r, c)\}_N$
end for

B.2.3. Off-Policy Baselines

CVPO-vanilla. Since SAFER is closely related to the EM-based safe RL algorithm CVPO (Liu et al., 2022b), we use it as a basic baseline and name it as CVPO-vanilla, which is shown in Algo.6.

CVPO-random. We adopt its variant CVPO-random which is trained under random noise as another baseline. More specifically, we simply add random noise when collecting the data.

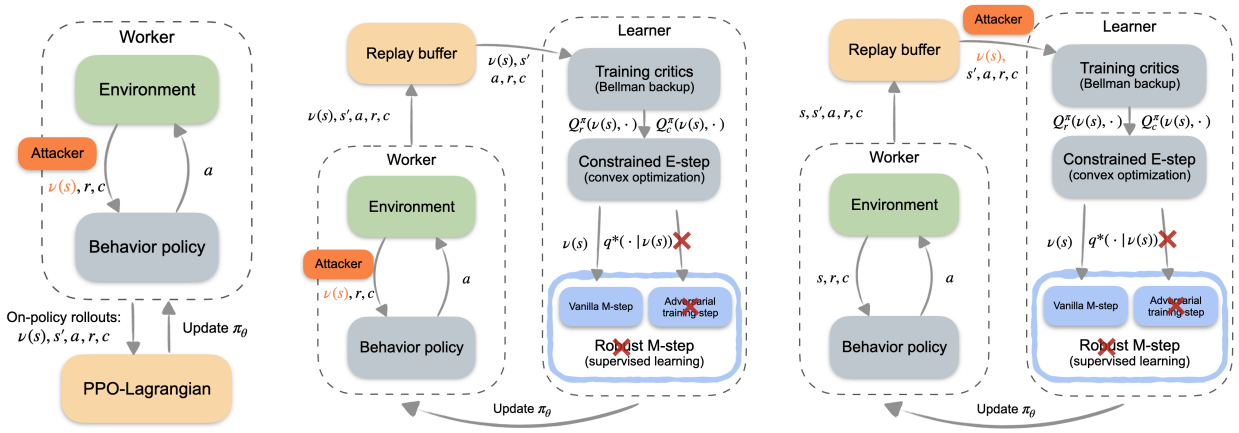
ADV-CVPO. To investigate the performance of directly applying the same online adversarial training techniques with the MC attacker in ADV-PPOL to the off-policy setting, we adopt the ADV-CVPO baseline, where the line 3 of Algo.6 is changed to $\mathcal{D} = \mathcal{D} \cup \{(\nu(s), a, s', r, c)\}$ where ν is the MC attacker and $a \sim \pi_{\theta_i}(\cdot|\nu(s))$. The algorithm diagram is shown in Fig. 3b.

ADV-EM-CVPO. We consider another intuitive and simple adversarial training method by attacking the sampled data from the replay buffer, which we name it as the ADV-EM-CVPO baseline, where the line 5 of Algo.6 is changed to $\{(\nu(s_n), a_n, s_{n+1}, r_n, c_n)_{n=1,...,N}\}$ where ν is the MC attacker. The data-flow diagram is shown in Fig. 3c. Due to the

Algorithm 6 CVPO Algorithm

Initialize the policy π_θ, reward value $Q_r^{\pi_\theta}$, cost value $Q_c^{\pi_\theta}$, and replay buffer $\mathcal{D} = \{\}$
for the i-th training epoch **do**
  Rollout trajectories by π_{θ_i} from the environment $\mathcal{D} = \mathcal{D} \cup \{(s, a, s', r, c)\}$
  for each policy optimization iteration **do**
    Sample N transitions from the replay buffer $\mathcal{D}$: $\{(s_n, a_n, s_{n+1}, r_n, c_n)_{n=1,\dots,N}\}$
    Update $Q_r^{\pi_{\theta_i}}, Q_c^{\pi_{\theta_i}}$ by the Bellman equation.
    $\triangleright$ *Constrained E-step begins*
    Compute dual variables η^*, λ^* by solving the convex optimization problem (6)
    Compute the variational distribution for each state $\{q^*(\cdot|s_n); n = 1, ..., N\}$ by Eq. (5)
    $\triangleright$ *M-step begins*
    for each M-step iteration **do**
      Update policy by solving Eq.(7)
    end for
  end for
end for

page limit, we omit the experiment results in the main content and leave them in B.4.



(a) ADV-PPOL (b) ADV-CVPO (c) ADV-EM-CVPO

Figure 3: Figure illustration of ADV-PPOL, ADV-CVPO, and ADV-EM-CVPO baselines.

B.3. Experiment Setting and Hyper-parameters

B.3.1. Experiment Descriptions

We use the Bullet safety gym (Gronauer, 2022) environments for this set of experiments. In the Run tasks, agents are rewarded for running fast between two safety boundaries and are given costs for violation constraints if they run across the boundaries or exceed an agent-specific velocity threshold. The reward and cost functions are defined as:

$$r(\boldsymbol{s_t}) = ||\boldsymbol{x_{t-1}} - \boldsymbol{g}||_2 - ||\boldsymbol{x}_t - \boldsymbol{g}||_2 + r_{robot}(s_t)$$
$$c(\boldsymbol{s_t}) = \mathbf{1}(|y| > y_{lim}) + \mathbf{1}(||\boldsymbol{v_t}||_2 > v_{lim})$$

where v_{lim} is the speed limit, y_{lim} specifies the safety region, $\boldsymbol{v_t} = [v_x, v_y]$ is the velocity of the agent at timestamp t, $\boldsymbol{g} = [g_x, g_y]$ is the position of a fictitious target, $\boldsymbol{x_t} = [x_t, y_t]$ is the position of the agent at timestamp t, and $r_{robot}(\boldsymbol{s_t})$ is the specific reward for different robot. For example, an ant robot will gain reward if its feet do not collide with each other. In the Circle tasks, the agents are rewarded for running in a circle in a clockwise direction but are constrained to stay

within a safe region that is smaller than the radius of the target circle. The reward and cost functions are defined as:

$$r(\boldsymbol{s_t}) = \frac{-y_t v_x + x_t v_y}{1 + |||\boldsymbol{x_t}||_2 - r|} + r_{robot}(\boldsymbol{s_t})$$
$$c(\boldsymbol{s_t}) = \mathbf{1}(|x| > x_{lim})$$

where r is the radius of the circle, and x_{lim} specifies the range of the safety region.

B.3.2. HYPER-PARAMETERS

For the on-policy baselines, we use Gaussian policies with mean vectors given as the outputs of neural networks, and with variances that are separate learnable parameters. The policy networks and Q networks for all experiments have two hidden layers of sizes (256, 256) with ReLU activation functions. We use a discount factor of $\gamma = 0.995$, a GAE-λ for estimating the regular advantages of $\lambda^{GAE} = 0.97$, a KL-divergence step size of $\delta_{KL} = 0.01$, a clipping coefficient of 0.02. The PID parameters for the Lagrange multiplier are: $K_p = 0.1$, $K_I = 0.003$, and $K_D = 0.001$. The learning rate of the adversarial attackers: MC and MR is 0.2. The optimization steps of MC and MR is 200. We choose larger perturbation range for the Car and Ball robots because they are simpler and easier to train. We use the same minibatch size, rollout length, cost limit, perturbation ϵ, and hyperparameters of the attackers for the off-policy baselines for fair comparison. The complete hyperparameters used in the experiments are shown in Table 4.

Table 4: Hyperparameters for on-policy baselines (left) and off-policy baselines (right).

Parameter	Ball-Circle	Car-Circle	Dron-Run	Ant-Run
training epoch	300	500	250	250
batch size	45000	45000	60000	60000
minibatch size	300	300	300	300
rollout length	200	300	100	200
cost limit	5	5	5	5
perturbation ϵ	0.05	0.05	0.025	0.025
actor optimization step	80	80	80	160
actor learning rate	0.0005	0.0003	0.0003	0.0005
critic learning rate	0.001	0.001	0.001	0.001

Parameter	Ball-Circle	Car-Circle	Drone-Run	Ant-Run
training epoch	900	700	700	600
batch size	50000	80000	20000	80000
particle size	32	32	32	64
M-step iterations	6	15	6	8
E-step KL threshold δ	0.05	0.1	0.01	0.1
VM-step KL threshold ξ	0.02	0.01	0.0008	0.05
KL threshold ξ in Eq.(8)	0.05	0.012	0.06	0.06
actor learning rate	0.002	0.002	0.001	0.001
critic learning rate	0.001	0.001	0.001	0.001

B.4. Additional Experiment Results

Table 5: Evaluation results of natural performance (no attack), under MC and MR attackers, and the average of them. Each value is reported as: mean ± standard deviation for 50 episodes and 5 seeds. We shadow two lowest-cost agents under each attacker column and break ties based on rewards, excluding the failing agents (natural rewards are less than 10% of CVPO-vanilla's) that are marked with $\star$.

Env	Method		Natural		MC		MR		Average	
			Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost
Ball-Circle $\epsilon = 0.05$	On-policy	SA-PPOL	705.52±71.79	2.82±3.98	688.24±30.12	48.48±12.14	760.64±51.59	55.83±6.14	718.13±46.74	35.71±4.58
		SA-PPOL(MC)	681.98±76.43	3.07±2.03	659.13±28.9	42.53±7.52	726.58±81.7	46.38±9.28	689.23±50.48	30.66±3.97
		ADV-PPOL	467.54±38.23	0.0±0.0	666.74±28.22	2.1±2.27	447.53±100.62	1.38±2.58	527.27±37.94	1.16±1.41
	Off-policy	CVPO-vanilla	622.47±103.38	0.0±0.0	556.08±102.2	20.91±13.31	709.5±65.01	44.82±13.69	629.35±85.73	21.91±8.37
		CVPO-random	548.16±46.93	0.0±0.0	546.75±22.05	7.06±6.3	647.34±52.21	29.99±16.05	580.75±34.29	12.35±5.88
		$\star$ADV-CVPO	4.35±6.14	0.0±0.0	14.39±16.6	0.0±0.0	13.94±23.12	0.11±0.65	10.89±11.32	0.04±0.22
		SAFER	610.03±29.19	0.0±0.0	625.93±26.26	0.47±1.37	661.22±38.52	1.99±3.51	632.39±27.19	0.82±1.41
Car-Circle $\epsilon = 0.05$	On-policy	SA-PPOL	440.79±9.81	0.25±1.1	275.01±95.31	71.62±55.24	393.56±65.58	96.4±27.27	369.79±45.12	56.09±19.62
		SA-PPOL(MC)	439.44±9.51	0.35±1.72	348.21±18.73	91.88±32.03	375.59±66.86	56.22±27.31	387.75±25.92	49.48±17.36
		ADV-PPOL	300.0±11.82	0.0±0.0	338.79±20.22	0.28±2.05	281.64±18.84	0.47±2.56	306.81±9.93	0.25±1.08
	Off-policy	CVPO-vanilla	297.2±118.44	0.38±1.21	244.4±87.79	57.21±27.06	287.4±80.68	38.75±24.44	276.33±80.76	32.11±12.54
		CVPO-random	170.33±15.11	0.0±0.0	166.5±64.51	7.51±12.6	214.61±57.07	2.3±6.23	183.81±40.2	3.27±4.9
		$\star$ADV-CVPO	0.42±2.32	0.0±0.0	0.48±2.16	0.0±0.0	0.57±2.05	0.0±0.0	0.49±1.37	0.0±0.0
		SAFER	258.63±18.72	0.0±0.0	313.61±14.81	0.1±0.89	333.29±13.21	0.32±2.13	301.84±12.13	0.14±0.76
Drone-Run $\epsilon = 0.025$	On-policy	SA-PPOL	338.07±3.49	0.0±0.0	-72.72±332.0	59.27±17.83	258.35±260.66	69.97±12.01	174.57±147.34	43.08±7.84
		SA-PPOL(MC)	183.9±12.3	0.0±0.0	206.77±6.84	5.1±0.44	214.32±10.04	5.25±0.54	201.66±6.53	3.45±0.26
		ADV-PPOL	265.84±7.13	0.0±0.0	298.7±19.23	1.5±2.31	264.37±35.82	0.55±1.15	276.3±17.89	0.68±1.05
	Off-policy	CVPO-vanilla	347.17±2.54	0.0±0.0	353.62±82.7	63.52±24.04	387.82±18.34	71.28±9.24	362.87±27.57	44.93±7.7
		CVPO-random	284.98±28.65	0.0±0.0	300.0±36.45	40.63±21.91	337.03±24.67	72.32±6.11	307.33±26.91	37.65±7.71
		ADV-CVPO	222.7±0.92	38.0±0.0	309.14±29.98	63.35±2.8	285.16±37.74	56.75±2.19	272.33±16.63	52.7±1.18
		SAFER	193.26±24.43	0.0±0.0	210.99±28.92	0.63±2.38	220.17±20.29	0.68±2.56	208.14±21.3	0.44±1.33
Ant-Run $\epsilon = 0.025$	On-Policy	SA-PPOL	699.78±7.11	1.47±1.23	692.34±8.51	62.2±13.2	714.38±11.66	103.63±14.42	702.17±3.89	55.77±7.29
		SA-PPOL(MC)	547.23±86.96	1.03±1.63	575.58±101.39	25.65±16.33	584.42±88.15	28.35±19.83	569.07±91.35	18.34±11.7
		ADV-PPOL	608.02±5.89	0.0±0.0	668.25±5.23	0.55±0.59	672.79±9.95	1.32±1.19	649.69±6.4	0.62±0.45
	Off-policy	CVPO-vanilla	686.59±15.04	0.85±1.05	668.53±31.97	86.03±16.86	728.37±77.04	164.65±24.01	694.5±28.86	83.84±9.5
		CVPO-random	682.28±15.78	1.17±1.11	672.27±71.41	100.5±26.87	747.65±22.68	173.45±12.65	700.73±25.92	91.71±9.02
		ADV-CVPO	334.31±136.22	79.08±60.5	302.15±160.75	75.02±60.35	330.5±106.58	68.3±54.03	322.32±112.81	74.13±52.35
		SAFER	496.11±60.67	0.17±0.45	514.8±59.42	0.82±1.19	555.81±15.01	1.5±1.72	522.24±32.35	0.83±0.67

Complete evaluation results. The experiment results of trained safe RL policies in all tasks are shown in Table 5. The natural column is the same one in Table 2. Video demos can be found in our website: https://sites.google.com/view/safer-rl/home.

The performance of applying on-policy adversarial training tricks in the off-policy setting. The performance of ADV-CVPO demonstrates that the successful on-policy adversarial training techniques in ADV-PPOL do not work in the off-policy setting. We adopt an additional baseline ADV-EM-CVPO as introduced in Appendix B.2.3 and Fig. 3c for a complete comparison. We first collect benign data (s, a, s', r, c) and store them in the data buffer. After they are sampled from the data buffer we use MC attacker to obtain $(\nu(s), a, s', r, c)$, which are used to train the agents. From Table 6, we can observe that the agents have poor task and safety performance, which means that simply applying the on-policy adversarial training techniques to the benign data from the reply buffer also does not work in the off-policy setting.

Table 6: Ablation study of ADV-EM-CVPO: training CVPO-vanilla by attacking the benign data from replay buffer. Each value is reported as: mean ± standard deviation for 50 episodes. The last row is the average performance of SAFER (the same as Table 2).

		Ball-Circle	Car-Circle	Drone-Run	Ant-Run
Natural	Reward	296.18±83.46	19.06±17.77	259.56±14.98	503.31±43.58
	Cost	17.41±12.23	17.3±53.58	22.15±1.65	13.35±21.69
MC	Reward	307.12±107.76	19.85±19.45	299.59±14.4	500.01±60.62
	Cost	28.6±20.39	7.74±31.04	52.3±9.25	26.83±37.32
MR	Reward	353.38±85.67	22.75±21.5	306.4±13.72	541.88±51.4
	Cost	33.02±20.43	17.01±51.36	47.25±5.64	8.45±14.51
Average	Reward	318.89±86.73	20.55±13.79	288.51±9.65	515.07±43.77
	Cost	26.35±16.08	14.02±26.97	40.57±3.57	16.21±22.53
SAFER Average	Reward	632.39±27.19	301.84±12.13	193.26±24.43	522.24±32.35
	Cost	0.82±1.41	0.14±0.76	0.0±0.0	0.83±0.67

The role of KL constraint threshold in the adversarial training step. Table 7 shows the agents' performance under different KL constraint threshold $\tilde{\xi}$ in Eq.(8). The worst case cost is the maximum cost under MC and MR attackers and the worst case reward is the corresponding reward to the worst case cost. We can observe that as the KL constraint threshold increases, the cost also increases, which validates Theorem 2 about the cost bound of robust M-step under attacks.

Table 7: Ablation study of KL regularizer in robust M-step. Evaluation results of the worst-case cost and reward under MC and MR attacker. Each value is reported as: mean ± standard deviation for 50 episodes and 5 seeds.

Ball-Circle $\epsilon = 0.05$	$\tilde{\xi}$	0.03	0.05	0.07	0.09	0.11
	Reward	537.27±23.78	552.51±113.15	638.43±90.95	567.85±69.97	695.89±5.44
	Cost	1.7±2.77	1.72±2.55	2.53±2.36	3.82±5.82	5.35±1.53
Car-Circle $\epsilon = 0.05$	$\tilde{\xi}$	0.01	0.02	0.03	0.05	0.09
	Reward	244.77±44.51	379.82±16.64	357.11±20.29	328.98±13.46	332.82±12.3
	Cost	0.0±0.0	0.65±2.52	2.9±6.69	4.97±5.52	5.76±11.5
Ant-Run $\epsilon = 0.025$	$\tilde{\xi}$	0.05	0.07	0.09	0.11	0.12
	Reward	563.61±4.1	572.12±18.55	584.53±18.29	531.37±45.7	585.7±26.67
	Cost	0.15±0.36	4.0±3.83	6.42±2.5	7.63±5.16	12.02±10.36

The influence of each component in the robust M-step. Recall that the robust M-step consists of a vanilla M-step and an adversarial training step whose objective is the Maximum Likelihood Estimation (MLE) loss, and the constraint is the KL divergence. We investigate their effect by removing each component from the full SAFER algorithm. The complete results are shown in Table 8. We can clearly see that the agent fails to learn if we remove the vanilla M-step in training, which is because the adversarial training alone can not ensure the agent's performance in the natural environment and thus corrupt the learning process, as we introduced in Proposition 2. We can also observe significant safety performance (cost) degradation if we remove the KL constraint and task performance (reward) drop if we remove the MLE loss in the adversarial training step. We can conclude that the vanilla M-step is the most important part of the robust M-step, with KL being more important regarding safety performance and KLE being more important when it comes to task performance. Therefore, each component in the robust M-step is necessary for SAFER, and ignoring any one of them has a negative effect on the overall performance.

Table 8: Ablation study of removing the vanilla M-step (VM), the KL constraint, and the MLE loss. Evaluation results of natural performance (no attack), under MC and MR attackers, and the average of them. Each value is reported as: mean ± standard deviation for 50 episodes and 5 seeds. We mark the failing agents whose naturla rewards are less than 10% of CVPO-vanilla with $\star$.

Env	Method	Natural		MC		MR		Average	
		Reward	Cost	Reward	Cost	Reward	Cost	Reward	Cost
Ball-Circle $\epsilon = 0.05$	without VM	136.91±99.81	0.0±0.0	160.28±112.68	0.0±0.0	198.37±93.17	0.0±0.0	165.19±88.9	0.0±0.0
	without KL	507.61±76.75	3.13±24.07	524.19±98.64	7.45±33.6	602.17±64.48	6.73±8.32	544.66±45.96	5.77±14.59
	without MLE	488.13±67.06	0.0±0.0	536.5±34.62	0.55±1.2	583.31±43.64	3.15±3.79	535.98±44.05	1.23±1.31
	Full SAFER	610.03±29.19	0.0±0.0	625.93±26.26	0.47±1.37	661.22±38.52	1.99±3.51	632.39±27.19	0.82±1.41
Car-Circle $\epsilon = 0.05$	$\star$without VM	9.06±11.57	0.0±0.0	10.98±12.27	5.37±30.37	14.06±15.15	2.25±10.93	11.36±10.75	2.54±10.63
	without KL	373.56±42.94	0.0±0.0	313.82±31.71	48.43±40.51	395.2±35.62	22.15±22.76	360.86±31.67	23.53±19.9
	without MLE	244.5±27.6	0.0±0.0	283.61±19.49	3.0±6.32	323.07±19.23	2.97±4.48	283.73±17.94	1.99±2.41
	Full SAFER	258.63±18.72	0.0±0.0	313.61±14.81	0.1±0.89	333.29±13.21	0.32±2.13	301.84±12.13	0.14±0.76
Drone-Run $\epsilon = 0.025$	$\star$without VM	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
	without KL	75.96±54.96	0.0±0.0	98.85±70.89	0.0±0.0	100.19±71.58	0.0±0.0	91.67±65.57	0.0±0.0
	without MLE	157.96±27.7	0.0±0.0	178.41±32.07	0.52±1.55	180.33±29.99	1.02±3.22	172.24±27.76	0.51±1.19
	Full SAFER	193.26±24.43	0.0±0.0	210.99±28.92	0.63±2.38	220.17±20.29	0.68±2.56	208.14±21.3	0.44±1.33
Ant-Run $\epsilon = 0.025$	$\star$without VM	37.64±23.5	0.0±0.0	48.12±21.47	0.18±0.63	51.98±22.32	0.0±0.0	45.91±18.71	0.06±0.21
	without KL	107.44±186.98	1.78±7.86	173.11±127.05	0.63±2.66	158.74±156.49	2.32±6.69	146.43±107.16	1.58±3.53
	without MLE	175.11±50.11	0.42±0.86	178.46±49.64	0.38±0.83	183.44±48.68	0.75±1.39	179.0±45.3	0.52±0.7
	Full SAFER	496.11±60.67	0.17±0.45	514.8±59.42	0.82±1.19	555.81±15.01	1.5±1.72	522.24±32.35	0.83±0.67