5-2024

Regret-based defense in adversarial reinforcement learning

Roman BELAIRE

Pradeep VARAKANTHAM

Thanh Hong NGUYEN

David LO
Singapore Management University, davidlo@smu.edu.sg

Citation

BELAIRE, Roman; VARAKANTHAM, Pradeep; NGUYEN, Thanh Hong; and LO, David. Regret-based defense in adversarial reinforcement learning. (2024). *AAMAS '24: Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, Auckland, New Zealand, May 6-10*. 2633-2640.

Available at: https://ink.library.smu.edu.sg/sis_research/9243

Regret-based Defense in Adversarial Reinforcement Learning

AAAI Track

Roman Belaire
Singapore Management University
Singapore
rbelaire.2021@phdcs.smu.edu.sg

Pradeep Varakantham
Singapore Management University
Singapore
pradeepv@smu.edu.sg

Thanh Nguyen
University of Oregon
Eugene, OR, United States
thanhhng@cs.uoregon.edu

David Lo
Singapore Management University
Singapore
davidlo@smu.edu.sg

ABSTRACT

Deep Reinforcement Learning (DRL) policies are vulnerable to adversarial noise in observations, which can have disastrous consequences in safety-critical environments. For instance, a self-driving car receiving adversarially perturbed sensory observations about traffic signs (e.g., a stop sign physically altered to be perceived as a speed limit sign) can be fatal. Leading existing approaches for making RL algorithms robust to an observation-perturbing adversary have focused on (a) regularization approaches that make expected value objectives robust by adding adversarial loss terms; or (b) employing "maximin" (i.e., maximizing the minimum value) notions of robustness. While regularization approaches are adept at reducing the probability of successful attacks, their performance drops significantly when an attack is successful. On the other hand, maximin objectives, while robust, can be extremely conservative. To this end, we focus on optimizing a well-studied robustness objective, namely regret. To ensure the solutions provided are not too conservative, we optimize an approximation of regret using three different methods. We demonstrate that our methods outperform existing best approaches for adversarial RL problems across a variety of standard benchmarks from literature.

KEYWORDS

Robust Reinforcement Learning; Adversarial Robustness; Regret

ACM Reference Format:
Roman Belaire, Pradeep Varakantham, Thanh Nguyen, and David Lo. 2024. Regret-based Defense in Adversarial Reinforcement Learning: AAAI Track. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 8 pages.

1 INTRODUCTION

Harnessing the power of Deep Neural Networks in DRL [22] allows RL models to achieve outstanding results on complex and even safety-critical tasks, such as self-driving [16, 32]. However, DNN performance is known to be vulnerable to attacks on input, consequently impacting DRL models which rely on them [10, 25, 33]. In such perturbations, adversaries only alter the observations received by an RL agent and not the underlying state or the dynamics (transition function) of the environment. Even with limited perturbations, high-risk tasks such as self-driving yield opportunities for significant harm to property and loss of life. One such example is presented by [7], in which a stop sign is both digitally and physically altered to attack an object recognition model.

Although the effectiveness of known adversarial examples can be mitigated by adversarial training (supervised training against adversarial examples) [10, 11, 25, 34], this does not guarantee the ability to generalize to unseen adversaries. Additionally, it has been shown that naive adversarial training in RL leads to unstable training and lowered agent performance, if effective at all [38]. Thus, we need algorithms that are not tailored to specific adversarial perturbations but are inherently robust. Rather than develop a policy that is value-optimal for as many known examples as possible, we want to determine what behavior and states involve risk and reduce them across the horizon. To achieve this, maximin methods operate to maximize the minimum reward of a policy [8, 19], which can be robust but often trades unperturbed solution quality to improve the lower bound. Regularization methods construct adversarial loss terms to ensure actions remain unchanged across similar inputs [19, 24, 38], reducing the probability of a successful adversarial attack. However, as we empirically show later, they remain vulnerable when attacks are successful.

To these ends, we provide a regret-based adversarial defense approach that aims to reduce the impact of a successful attack without being overly conservative.

Through our contributions, we:

- Formally define regret in settings where an observation-perturbing adversary is present. Broadly, regret is the difference of value achieved in the absence versus in the presence of an observation-perturbing adversary.
- Derive an approximation of regret, named Cumulative Contradictory Expected Regret (CCER), which is amenable to scalable optimization due to satisfying the optimal sub-structure property. We provide a value iteration approach to minimize CCER, named RAD-DRN (Regret-based Adversarial Defense with Deep Regret Networks). Using CCER, our approach can balance robustness and nominal solution quality.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

- Provide a policy gradient approach to minimize CCER, named RAD-PPO. We derive the policy gradient w.r.t the regret measure and utilize it in the PPO framework.
- Provide a Cognitive Hierarchy Theory based approach named RAD-CHT that generates potential adversarial policies and computes a regret-based robust response, to demonstrate an application of CCER in adversary-reactive frameworks.
- Finally, we provide detailed experimental results on multiple benchmark problems (MuJoCo, Atari, and Highway) to demonstrate the utility of our approaches against several leading approaches in adversarial RL. Like previous approaches, we demonstrate the performance of our approach against strong greedy attacks (e.g., PGD). Unlike existing work, we also show effectiveness against multi-step attack strategies that are directly computed against victim policy.

Algo.	Num. Atks	% Success	Score
RADIAL	6	5%	4.16
WocaR	6	12%	6.63
RAD-DRN	6	40%	9.9
RAD-PPO	7	10%	18
RAD-CHT	8	60%	20.1

Table 1: The impact and frequency of successful attacks on an example set of the Strategically Timed Attack [20] on *highway-fast-v0*. The Num. Atks column shows the number of attempted attacks; the % Success column indicates the rate at which an attempted attack changed the selected action. When unperturbed, all methods reach approximately 22 points.

2 RELATED WORK

Adversarial Attacks in RL. Deep RL has shown to be vulnerable to attacks on its input, whether from methods with storied success against DNNs such as an FGSM attack [11, 12], tailored attacks against the value function [17, 33], or adversarial behavior learned by an opposing policy [8, 10, 24, 38]. We compile attacks on RL loosely into two groups of learned adversarial policies: observation poisonings [10, 20, 33] and direct ego-state disruptions [26, 27]. Each category has white-box counterparts that leverage the victim's network gradients to generate attacks [8, 11, 12, 24]. While previous methods focus on robustness against one or the other, we demonstrate that the proposed methods are comparably robust to both categories of attacks.

Adversarial Training. In this area, adversarial examples are found or generated and integrated into the set of training inputs [2, 9, 21, 30, 31, 37]. For a comprehensive review, we refer readers to [3]. In RL, research efforts have demonstrated the viability of training RL agents against adversarial examples [4, 10, 14, 26, 35]. Naively training RL agents against known adversaries is a sufficient defense against known attacks; however, new or more general adversaries remain effective [10, 15] and therefore we focus on proactively robust defense methods instead of reactive (react to known adversaries) defense methods.

Robustness through Regularization. Regularization approaches [8, 24, 38] take vanilla value-optimized policies and robustify them to minimize the loss due to adversarial perturbations. These approaches utilize certifiable robustness bounds computed for neural networks when evaluating adversarial loss and ensure the probability of success of an attack is reduced using these lower bounds. *Despite lowering the likelihood of a successful attack, an attack that does break through will still be effective (as shown in Table 1).* Note that for RADIAL (a regularization approach), even though the success percentage of attacks is the least, it has the largest drop in performance.

Regret Optimization in MDPs. Measuring and optimizing a regret value to improve the robustness has been studied previously in uncertain Markov Decision Processes (MDPs)[1, 28]. In RL, [13] established Advantage-Like Regret Minimization (ARM) as a policy gradient solution for agents robust to partially observable environments. While the properties and optimization of regret are in general well-studied, *current applications in RL focus on utilizing regret as a robustness tool against natural environment variance; to the best of our knowledge, this is the first application of regret to defend against strategic adversarial perturbations.*

Environment	Algorithm	Unperturbed	Random Pert.
Hopper	PA-ATLA-PPO	3449	1564
	WocaR-PPO	3136	3242
	RAD-PPO	3473	3415
HalfCheetah	PA-ATLA-PPO	6289	3414
	WocaR-PPO	3993	4128
	RAD-PPO	4426	4387

Table 2: Adversary-specific robustness methods (forms of adversarial retraining) are robust only to the adversarial behavior they specifically train against. PA-ATLA-PPO is the SoTA adversarial retraining method, and WocaR-PPO is the SoTA adversary-agnostic method. RAD-PPO is our best-performing adversary-agnostic approach. Note that the adversary-specific models perform poorly against even random adversarial perturbations, which are weaker than other attacks. Attacks use perturbation radius $\epsilon = 0.15$.

Adversary-Agnostic Approaches Unlike adversary-specific robustness training, the methods we term as "adversary-agnostic" do not interact with a perturbed MDP during training. While the various forms of adversarial retraining do have merit, they often take longer to train (needing to train both victim and adversary policies). PA-ATLA-PPO [34], a SoTA adversarial retraining technique, reports needing 2 million training frames for MuJoCo-Halfcheetah. For comparison, both our proposed RAD-PPO and WocaR-PPO [19], another SoTA adversary-agnostic method, require less than 40% of the training frames. Furthermore, adversary-specific methods may have subpar performance against novel adversaries, in addition to well-known drawbacks such as catastrophic forgetting. [38] demonstrates how naive adversarial retraining is in general a poor solution; we further demonstrate in Table 2 that even advanced retraining frameworks are not as generally robust.

3 RL WITH ADVERSARIAL OBSERVATIONS

In problems of interest, we have an RL agent whose state/observation and action space are S and A respectively. There is an underlying transition function $T : S \times A \times S \to [0, 1]$ and reward model $R : S \times A \to \mathbb{R}$ which are not known *a priori* to the RL agent. The behavior of the agent is governed by a policy $\pi : S \to A$ that maps states to actions. The typical objective for an RL agent is to learn a policy π^* that maximizes its expected value without knowing the underlying transition and reward model *a priori*. This can be formulated as the following optimization problem:

$$\pi^* \in \arg\max_\pi V^\pi(s_0)$$

where $V^\pi(s_0)$ is the expected accumulated value the RL agent obtains starting from state s_0 for executing policy π. The policy computed is not robust in the presence of an adversary who can strategically alter the observation received by the agent at any time step. This is because the agent could be executing the wrong action for the underlying state.[1]

For the rest of this paper, we will use z_t to represent the observed (possibly perturbed) state and s_t to refer to the true underlying state at step t. Formally, on taking action a_t in state s_t, the environment transitions to a new state s_{t+1}. However, the adversary can alter the observation received by the RL agent to another state z_{t+1} instead of s_{t+1} to reduce the expected value of the RL agent, where $z_{t+1} \in N(s_{t+1})$ — a set of neighbors of s_{t+1}, defined as follows:

$$N(s_t) = \{z_t : \|z_t - s_t\|_\infty \leq \epsilon\}$$

Often, it is not possible to discern if the observation is perturbed or not. For example, given a highway speed limit sign, a perturbation showing a different speed limit may be undetectable for an agent and hence would be within the neighborhood. However, this is within a reasonable constraint; perturbing the speed limit sign to be a stop sign may be easy to heuristically distinguish and hence would not be part of the neighborhood.

We now can represent the *observation-altering adversarial policy* as $\mu : S \to S$. The notion of neighborhoods translates to adversarial policies μ such that $\mu(s) \in N(s)$. When the adversarial policy is deterministic and bijective, for a given observed state z, we will employ $\mu^{-1}(z)$ to retrieve the underlying unperturbed state $s \in N(z)$. In the following, we focus on computing policies that are inherently robust, even without knowing the adversary policy.

4 REGRET-BASED ADVERSARIAL DEFENSE (RAD)

Our approach to computing inherently robust policies is based on minimizing the maximum *regret* the agent receives for taking the wrong action assuming there was a perturbation in states. We first introduce the notions of regret and max regret for the RL agent to play a certain policy π. The regret-based robust policy is then formulated accordingly.[2]

[1]It should be noted that an adversary is assumed to alter only the agent observations and not the model dynamics or the real states.

[2]Our regret and minimax regret policy are defined w.r.t deterministic policies π and μ, for the sake of representation. Extending these definitions for stochastic policies is straightforward.

Definition 1 (Regret). *Given an adversary policy μ, the regret at each observed state z_t for the RL agent to play a policy π is defined as follows:*

$$\delta^{\pi,\mu}(z_t) = V^\pi(z_t) - V^{\pi,\mu}(z_t) \tag{1}$$

Intuitively, it is the difference between the expected value $V^\pi(\cdot)$ (assuming no adversary) and the value $V^{\pi,\mu}(\cdot)$ (assuming states are being perturbed by the adversary policy μ) while the RL agent takes actions according to a policy π. In particular, the value for an agent with a policy π in the absence of an adversary, $V^\pi(\cdot)$, is given by:

$$V^\pi(z_t) = R(z_t, \pi(z_t)) + \gamma \mathbb{E}_{z_{t+1}}\left[V^\pi(z_{t+1})\right] \tag{2}$$

where $z_{t+1} \sim T(\cdot|z_t, \pi(z_t))$. Note that when the adversary is not present, the observed states $z_t = s_t$ and $z_{t+1} = s_{t+1}$.

On the other hand, the $V^{\pi,\mu}(.)$ function for an agent with policy π in the presence of such an adversary μ is given by:

$$V^{\pi,\mu}(z_t) = R(s_t, \pi(z_t)) + \gamma \mathbb{E}_{z_{t+1}}\left[V^{\pi,\mu}(z_{t+1})\right] \tag{3}$$

where $s_t = \mu^{-1}(z_t)$, $z_{t+1} = \mu(s_{t+1})$ with $s_{t+1} \sim T(\cdot|s_t, \pi(z_t))$. Note that our regret is defined based on observed states, as the agent will only have access to observed states (and not the true states) when making test-time decisions.

Definition 2 (Minimax Regret Policy). *The regret-based robust policy for the RL agent is the policy that minimizes the maximum regret at the initial observed state z_0 over all possible adversary policies, formulated as follows:*

$$\pi^* \in \arg\min_\pi \max_\mu \delta^{\pi,\mu}(z_0) \tag{4}$$

Intuitively, a minimax regret policy minimizes the maximum regret by avoiding actions that have high variance across neighboring states. Unfortunately, there are multiple issues with optimizing minimax regret. First, optimizing regret requires iterating over different adversary policies, which can be infinite depending on the state and action spaces. If we assume specific types of adversaries, robustness may not extrapolate to other adversary policies. Second, the minimax regret expression does not exhibit optimal substructure property, thus rendering value iteration approaches (e.g., Q-learning, DQN) theoretically invalid. Third, deriving policy gradients w.r.t. regret is computationally challenging, requiring combinatorial perturbed trajectory simulations. As a result, employing policy gradient approaches to optimize regret is also infeasible.

We address the above issues by using an approximate notion of regret referred to as Cumulative Contradictory Expected Regret (CCER) and then proposing two types of approaches (adversary agnostic[3] and adversary dependent):

- Adversary agnostic approach for optimizing approximate regret: CCER has multiple useful properties: (i) It satisfies the optimal substructure property, thereby allowing for usage of a DQN type approach; and (ii) It is possible to compute a policy gradient with regards to CCER, thereby allowing for a policy gradient approach.
- Adversary dependent approach for optimizing approximate regret: We utilize an iterative best-response approach based on Cognitive Hierarchical Theory (CHT) to ensure defense

[3]Agnostic methods do not receive perturbations while training.

against a distribution of "good" adversarial policies. This approach is referred to as RAD-CHT.

4.1 Regret Approximation: CCER

We introduce a new notion of approximate regret, referred to as CCER. CCER accumulates contradictory regret at each epoch; we refer to this regret as contradictory as the observed state's optimal action may contradict the true state's action. The key intuition is that we accumulate the regret (maximum difference in reward) in each time step regardless of whether the state is perturbed or not perturbed.

For the underlying transition function, T, and reward model, R, CCER w.r.t a policy π is defined as follows:

$$\delta^{\pi}_{\text{CCER}}(z_t) = R(z_t, \pi(z_t)) - \min_{s_t \in N(z_t)} R(s_t, \pi(z_t)) + \gamma \mathbb{E}_{z_{t+1}}\left[\delta^{\pi}_{\text{CCER}}(z_{t+1})\right] \quad (5)$$

where $z_{t+1} \sim T(\cdot|z_t, \pi(z_t))$. Our goal is to compute a policy that minimizes CCER, i.e.,

$$\pi^{\perp} \in \arg\min_{\pi} \ \delta^{\pi}_{\text{CCER}}(z_0)$$

This is a useful objective, as CCER accumulates the myopic regrets at each time step and we minimize this overall accumulation. Importantly, CCER has the optimal substructure property, i.e., the optimal solution for a sub-problem (from step t to horizon H) is also part of the optimal solution for the overall problem (from step 0 to horizon H).

Proposition 1 (Optimal Substructure property). *At time step t, the CCER corresponding to a policy, π at z_t, i.e., $\delta^{\pi}_{\text{CCER}}(z_t)$ is minimum if it includes the CCER minimizing policy from $t+1$, i.e.,$\pi^{\perp}_{[t+1,H]}$ from $t+1$. Formally,*

$$\delta^{\left\langle \pi_t, \pi^{\perp}_{[t+1,H]} \right\rangle}_{\text{CCER}}(z_t) \leq \delta^{\langle \pi_t, \pi_{[t+1,H]} \rangle}_{\text{CCER}}(z_t)$$

All proofs are in Appendix A. Similar to the *state* and *state-action* value function, $V(s)$ and $Q(s,a)$, we also have state regret, $\delta^{\pi}_{\text{CCER}}(z_t)$ and state-action regret, $\delta^{\pi}_{\text{CCER}}(z_t, a_t)$. Specific definitions are provided in the following sections.

4.2 Approach 1: RAD-DRN

We first provide a mechanism similar to DQN [23] to compute a robust policy that minimizes CCER irrespective of any adversary. Briefly, we accumulate the regret over time steps (rather than reward) and act on the minimum of this estimate (rather than the maximum, as a DQN would). Intuitively, minimizing CCER ensures the obtained policy avoids taking actions that have high variance, therefore avoiding volatile states that have high regret. In our adversarial setting, this roughly corresponds to behavior that avoids states where false observations have q-values vastly different from the underlying state. In a driving scenario, for example, this could mean giving vehicles and obstacles enough berth to account for errors in distance sensing. Let us denote by:

$$\delta^{\pi}_{\text{CCER}}(z,a) = R(z,a) - \min_{s \in N(z)} R(s,a) + \gamma \mathbb{E}_{z'}\left[\min_{a'} \delta^{\pi}_{\text{CCER}}(z',a')\right]$$

as the *q-value* associated with CCER in our setting. We provide the pseudocode of our algorithm, named RAD-DRN, employing a neural network to predict $\delta_{\text{CCER}}(z,a)$ in Alg. 1. This network's parameter w is trained based on minimizing the loss between the predicted and observed δ_{CCER} values.

Algorithm 1: RAD-DRN

```
Initialize replay memory D to capacity N;
Initialize q-regret δ_CCER with random weights w;
Initialize target q-regret δ̂_CCER with weights w⁻ = w;
for episode = 1 → M do
    Get initial state z_0;
    for t = 0 → H do
        With prob. ϵ, select a random action a_t;
        Else, select a_t ∈ argmin_a δ_CCER(z_t, a, w);
        Execute action a_t, get observed state z_{t+1};
        Observe regret r_t = R(z_t, a_t) − min_{s_t ∈ N(z_t)} R(s_t, a);
        Store transition (z_t, a_t, z_{t+1}, r_t) → D;
        Sample mini-batch (z_i, a_i, z_{i+1}, r_i) ~ D;
        for each (z_i, a_i, z_{i+1}, r_i) in mini-batch do
            Set target y_i =
              { r_i, if episode terminates at step i+1
              { r_i + γ min_{a'} δ̂_CCER(z_{i+1}, a'; w⁻), otherwise
            Perform a gradient descent to update w based on
              loss: [y_i − δ_CCER(z_i, a_i, w)]²;
    Every K steps reset w⁻ = w;
```

Given the conservative nature of regret, RAD-DRN can provide degraded nominal (unperturbed) performance, especially in settings with very few high-variance states. We propose a heuristic optimization trick to reduce unnecessary conservatism in RAD-DRN. We consider using weighted combinations of value and CCER estimates to determine the utility of employing each policy at a given observation. Intuitively, we want to apply a value-maximizing policy in low-variance state neighborhoods and a regret-minimizing policy in high-variance neighborhoods; as such, the utility of using a regret-minimizing policy in a high-regret neighborhood will, in turn, be high. Specifically, we train a DQN to maximize the utility function:

$$Util(z,\pi) = V^{\pi}(z) - \beta \delta^{\pi}_{\text{CCER}}(z)$$

where β is a caution-weight constant decimal. This final step requires a minimal amount of training (500 episodes) and interacts directly with the non-adversarial environment. We constrain the available actions of the utility-maximizing policy to either be the value-optimal action $\text{argmax}_a Q^{\pi}(z,a)$ or the CCER-optimal action $\text{argmin}_a \delta^{\pi}_{\text{CCER}}(z,a)$, avoiding actions that are sub-optimal to both measures. These two optimal DQN and RAD-DRN policies are obtained in advance before training the above utility-maximizing policy.

4.3 Approach 2: RAD-PPO

Approach 1 provides a regret iteration approach (along the lines of value iteration in Deep Q Learning). In this section, we provide a policy gradient approach that minimizes CCER. Proximal Policy

Optimization (PPO), and any policy gradient (PG) method, can be extended with contradictory regret. The key contribution is defining/deriving the policy gradient based on the long-term CCER of a policy:

$$\delta^{\pi}_{\text{CCER}}(z) = \sum_a \pi(z,a)\delta^{\pi}_{\text{CCER}}(z,a)$$

Proposition 2. *The gradient of a CCER-minimizing policy π_θ parameterized by θ can be computed as follows:*

$$\frac{\partial \delta^{\pi_\theta}_{\text{CCER}}(z)}{\partial \theta} = \sum_s P(z|\pi_\theta)\sum_a \frac{\partial \pi_\theta(z,a)}{\partial \theta}\delta^{\pi_\theta}_{\text{CCER}}(z,a) \tag{6}$$

where $P(z|\pi_\theta)$ is the stationary distribution w.r.t π_θ.

This can also be rewritten along similar lines as the traditional policy gradient using an expectation (instead of $P(z|\pi_\theta)$):

$$\nabla_\theta \delta^{\pi_\theta}_{\text{CCER}}(z_0) = \mathbb{E}_{\pi_\theta}\left[\delta^{\pi_\theta}_{\text{CCER}}(z,a)\nabla_\theta \log \pi_\theta(a \mid z)\right]$$

CCER-based Advantage. Furthermore, we can replace $\delta^{\pi_\theta}_{\text{CCER}}(z,a)$ on the right side of the gradient with a CCER advantage function defined as follows:

$$A^{\text{CCER}}(z_t,a) = -[\delta^{\pi_\theta}_{\text{CCER}}(z_t,a) - \delta^{\pi_\theta}_{\text{CCER}}(z_t)] \tag{7}$$

The positive value of $|A^{\text{CCER}}|$ can be understood as the increase in regret associated with increasing $\pi_\theta(a|z_t)$, as the value of the selected action relative to others will be lower when a perturbation occurs. Thus, we invert the sign so that regret and A^{CCER} are inversely related, i.e. selecting the action with the highest A^{CCER} will minimize regret.

4.4 Approach 3: RAD-CHT

The previous RAD-DRN and RAD-PPO approaches are agnostic to specific adversarial policies. Here, we provide a reactive framework that identifies strong adversarial policies and computes a best-response (minimum CCER) policy against them. This approach builds on the well-known behavior model in game theory and economics referred to as Cognitive Hierarchical Theory (CHT) [6]. In CHT, players' strategic reasoning is organized into multiple levels; players at each level assume others are playing at a lower level (i.e., are less strategic).

Our approach, named RAD-CHT, is an iterative algorithm that proceeds as follows:

- Iteration 0 (Level 0): Both the RL agent and the adversary play completely at random, with the random policies for agent and adversary given by $\pi^{(0)}$ and $\mu^{(0)}$ respectively.
- Iteration 1 (Level 1): The RL agent assumes a level 0 adversary and wants to find a new policy $\pi^{(1)}$ that minimizes the regret given $\mu^{(0)}$, formulated as follows:

$$\pi^{(1)} = \arg\min_\pi \delta^{\pi,\mu^{[(0)]}}_{\text{CCER}}(z_0)$$
$$\delta^{\pi,\mu^{[(0)]}}_{\text{CCER}}(z_t) = R(z_t,\pi(z_t)) - R(s_t,\pi(z_t)) + \gamma\mathbb{E}_{z_{t+1}}\left[\delta^{\pi,\mu^{[(0)]}}_{\text{CCER}}(z_{t+1})\right]$$

  where s_t is the true state given the observed state is z_t and the attack policy is $\mu^{(0)}$, i.e., $\mu^{(0)}(s_t) = z_t$. On the other hand, the level-1 adversary assumes the RL agent is at level 0 and attempts to find a perturbation policy that minimizes the agent's expected return:

$$\mu^{(1)} \in \arg\min_\mu V^{\pi^{(0)},\mu}(\cdot)$$

- Iteration $k > 1$ (Level k): The agent assumes the adversary can be at any level below k. Like in CHT, the adversary policy is assumed to be drawn from a Poisson distribution over the levels $0, 1, \cdots, k-1$ with a mean $k-1$. The RL agent then finds a new policy $\pi^{(k)}$ that minimizes the regret formulated as the following:

$$\pi^{(k)} \in \arg\min_\pi \delta^{\pi,\mu^{[(k-1)]}}_{\text{CCER}}(z_0)$$
$$\delta^{\pi,\mu^{[(k-1)]}}_{\text{CCER}}(z_t) = R(z_t,\pi(z_t)) - \sum_{i=0}^{k-1} P^{(k)}(i)R(s^{(i)}_t,\pi(z_t)) + \gamma\mathbb{E}_{z_{t+1}}\left[\delta^{\pi,\mu^{[(k-1)]}}_{\text{CCER}}(z_{t+1})\right]$$

  where $P^{(k)}(i) \propto \frac{\lambda^i e^{-\lambda}}{i!}$ (aka. Poisson distribution) is the probability the adversary is at level i with $0 \le i \le k-1$. The notation $[(k-1)]$ represents the range $(0),\ldots,(k-1)$. In addition, the true state is $s^{(i)}_t$, the observed state is z_t and the attack policy is $\mu^{(i)}$, i.e., $\mu^{(i)}(s^{(i)}_t) = z_t$.
  Similarly, the adversary at this level assumes the RL agent is at a level below k, following the Poisson distribution. The adversary then optimizes the perturbation policy as follows:

$$\mu^{(k)} \in \text{argmin}_\mu \sum_{i=0}^{k-1} P^{(k)}(i)V^{\pi^{(i)},\mu}(\cdot)$$

At each iteration, the CCER minimization problem given the adversary distribution exhibits optimal substructure. Therefore, we can adapt Approach 1 and 2 to compute the CCER-minimizing policy given the adversarial distribution.

On the adversary's side, we provide the policy gradient given the victim policies at previous levels. Let's denote

$$V_{\mu_\theta}(\cdot) = \sum_{i=0}^{k-1} P^{(k)}(i)V^{\pi^{(i)},\mu_\theta}(\cdot)$$

where θ is the parameter of the adversary policy μ.

Proposition 3. *The gradient of the objective function of the adversary $V_{\mu_\theta}(\cdot)$ w.r.t θ can be computed as follows:*

$$\nabla_\theta V_{\mu_\theta} = \mathbb{E}_{i\sim P^{(k)}(i),\tau\sim(\mu_\theta,\pi^{(i)},T)}\left[R(\tau)\sum_t \nabla_\theta \log \mu_\theta(z_t|s_t)\right]$$

where $\tau = (s_0,z_0,a_0,s_1,z_1,a_1,\cdots)$ with $z_t \sim \mu_\theta(s_t)$ is the perturbed state created by the adversary policy μ_θ. In addition, the reward of a trajectory τ is defined as follows:

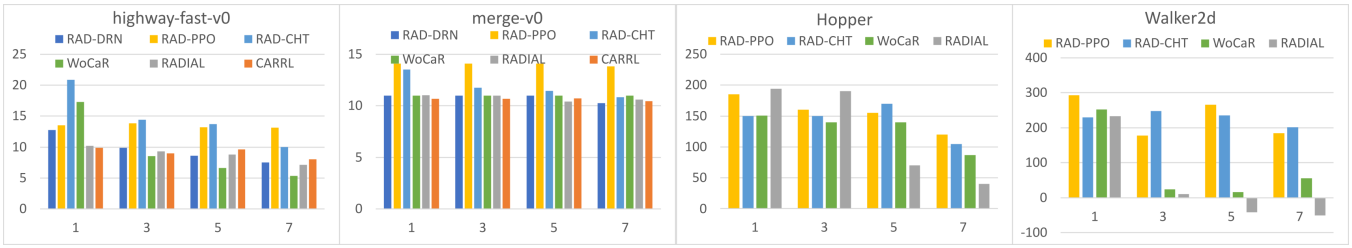
$$R(\tau) = \sum_t \gamma^t R(s_t,a_t)$$

Based on Proposition 3, finding an optimal adversarial policy at each level can be handled by a standard policy optimization algorithm such as PPO.

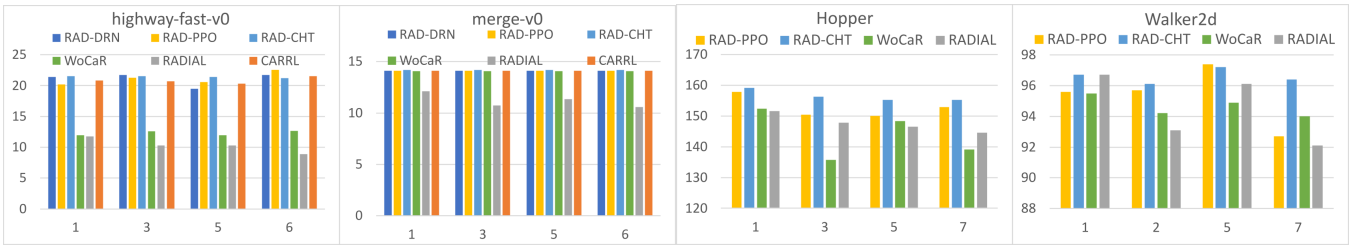
5 EXPERIMENTS

We provide empirical evidence to answer key questions:

- How do RAD approaches (RAD-DRN, RAD-PPO, RAD-CHT) compare against leading methods for Adversarially Robust RL on well-known baselines from MuJoCo, Atari, and Highway libraries?



(a) Score vs. Percentile Perturbation Frequency with Strategically Timed Attack



(b) Score vs. Percentile Perturbation Magnitude with Critical Point Attack

Figure 1: The performance of robust RL methods against strategic adversaries. The y-axis represents the score and the x-axis represents the intensity of the attack.

- For multi-step strategic attacks, how do RAD methods compare to leading defenses?
- Does RAD mitigate the value/robustness trade-off present in maximin methods?
- How does the performance of our approaches degrade as the intensity of attacks (i.e., # of attacks) is increased?

5.1 Experimental Setup

We evaluate our proposed methods on the commonly used Atari and MuJoCo domains [5, 36], and a suite of discrete-action self-driving tasks [18]. For the driving environments, each observation feature of the agent corresponds to the kinematic properties (coordinates, velocities, and angular headings) of the ego-vehicle and nearby vehicles. The task for each environment is to maximize the time spent in the fast lane while avoiding collisions. The road configurations for the *highway-fast-v0, merge-v0, roundabout-v0, and intersection-v0* tasks are, in order: a straight multi-lane highway, a two-lane highway with a merging on-ramp, a multi-lane roundabout, and a two-lane intersection. We use a standard training setup seen in [19, 24], detailed in Appendix C.

We compare RAD-DRN, RAD-PPO, and RAD-CHT to the following baselines: vanilla DQN [22] and PPO [29]; a simple but robust minimax method, CARRL [8]; a leading regularization approach, RADIAL [24]; and the current SoTA defense, WocaR [19]. We test all methods against both trained policy adversaries and gradient attacks, as well as the Strategically-Timed attack and Critical-Point attack, two leading multi-step strategic attacks [20, 33]. For RADIAL and WocaR, we use the best-performing methods for each environment (their proposed DQNs for discrete actions, and PPOs for MuJoCo).

Table 3: Results on Highway. Each row shows the mean scores of each RL method against different attacks. Further tasks are shown in Appendix B.

Algorithm	Unperturbed	WC Policy	PGD, $\epsilon = \frac{3}{255}$
highway-fast-v0			
DQN	24.91±20.27	3.68±35.41	15.71±13
PPO	22.8±5.42	13.63±19.85	15.21±16.1
CARRL	24.4±1.10	4.86±15.4	12.43±3.4
RADIAL	28.55±0.01	2.42±1.3	14.97±3.1
WocaR	21.49±0.01	6.15±0.3	6.19±0.4
RAD-DRN	24.85±0.01	**22.65±0.02**	18.8±24.6
RAD-PPO	21.01±1.23	20.59±4.10	20.02±0.01
RAD-CHT	21.83 ± 0.35	21.1 ± 0.24	**21.48 ± 1.8**

5.2 Worst Case Policy Attack

Each Worst-Case Policy (WC) adversary is a Q-learning agent that minimizes the overall return of the targeted agents by learning perturbed observations from a neighborhood distribution of surrounding states $N(s_t)$. Note that $N(s_t)$ is not necessarily equal to the training neighborhood that DRN samples; in our experiments, we increase the attack neighborhood radius by twenty percent (approximately). During the training of the adversary, it is permitted to perturb the victim's observations at every step. Each tested method has a corresponding WC Policy attacker. In the Highway environments, the perturbations correspond to a small shift in the position of a nearby vehicle; in the Atari domains, the input tensor is shifted up or down several pixels. For MuJoCo, we instead apply the Maximal Action Difference attack with $\epsilon = 0.15$ [38].

Table 4: Results on MuJoCo.

Algorithm	Unperturbed	MAD ϵ=0.15	PGD ϵ=$\frac{5}{255}$
Hopper			
PPO	2741 ± 104	970±19	36±156
RADIAL	3737±75	2401±13	3070±31
WocaR	3136±463	1510 ± 519	2647 ±310
RAD-PPO	3473±23	2783±325	**3110±30**
RAD-CHT	3506±377	**2910± 699**	3055 ±152
HalfCheetah			
PPO	5566 ± 12	1483±20	-27±1308
RADIAL	4724±76	4008±450	3911±129
WocaR	3993±152	3530±458	3475±610
RAD-PPO	4426±54	**4240±4**	**4022±851**
RAD-CHT	4230±140	4180±37	3934±486
Walker2d			
PPO	3635 ± 12	680±1570	730±262
RADIAL	5251±10	3895±128	3480±3.1
WocaR	4594±974	3928±1305	3944±508
RAD-PPO	4743±78	3922±426	**4136±639**
RAD-CHT	4790±61	**4228±539**	4009±516

Table 5: Results on Atari, with the same metrics as Table 3. Additional results in Appendix B.

Algorithm	Unperturbed	WC Policy	PGD ϵ=$\frac{5}{255}$
Pong			
PPO	21.0±0	-20.0± 0.07	-19.0±1.0
CARRL	13.0 ±1.2	11.0±0.010	6.0±1.2
RADIAL	21.0±0	11.0±2.9	**21.0± 0.01**
WocaR	21.0±0	18.7 ±0.10	20.0 ± 0.21
RAD-DRN	21.0±0	14.0 ± 0.04	14.0 ± 2.40
RAD-PPO	21.0±0	**20.1±1.0**	20.8±0.02
BankHeist			
PPO	1350±0.1	680±419	0±116
CARRL	849±0	830±32	790±110
RADIAL	1349±0	997±3	1130±6
WocaR	1220±0	1207±39	1154±94
RAD-DRN	1340±0	1170±42	1211±56
RAD-PPO	1340±0	**1301±8**	**1335±52**

5.3 Results

In each table, we report the mean return over 50 random seeds. The most robust score is shown in **boldface**.

Highway Domains: We report scores under PGD and WC attacks in Table 3. We observe that although the unperturbed performance of RAD- methods is lower than that of the vanilla solutions, all three approaches (RAD-DRN, RAD-PPO, and RAD-CHT) are extremely robust under different attacks and have higher performance than all other robust approaches in all of the Highway environments. We see that even though maximin methods (CARRL, WocaR) are clearly suited for some scenarios, RAD- methods still outperform them.

MuJoCo Domains: Table 4 reports the results on MuJoCo, playing three commonly tested environments seen in comparison literature [19, 24]. As the MuJoCo domains have continuous actions, we exclude value iteration-based methods (DQN, CARRL, and RAD-DRN). We directly integrated our proposed techniques into the implementation from [19]. Our assessment covers the MAD attack [38] with a testing $\epsilon = 0.15$, which marks the value at which we start observing deviations in the returns of the robustly trained agents. In addition, we subject the agents to PGD attacks with $\epsilon = \frac{5}{255}$.

Even in these problem settings, while RAD approaches do not provide the highest unperturbed performance (but are reasonably close), they (RAD-PPO and RAD-CHT) provide the best overall performance of all approaches.

Atari Domains: We report results on four Atari domains used in preceding works [19, 24] in Table 5. RAD-PPO outperforms other robust methods under nearly all attacks. RAD-DRN performs similarly to RAD-PPO, though faces some limitations in scenarios where rewards are sparse or mostly similar, such as in *Pong*. Regret measures the difference between outcomes, so the quality of training samples degrades when rewards are the same regardless of action. Interestingly, RAD-PPO does not suffer in this way, which we attribute to the superiority of the PPO framework over value iterative methods.

We exclude RAD-CHT from the Atari experiments, as the memory requirements to simultaneously hold models for all levels are outside the limitations of our hardware.

Strategic Attacks: Unlike previous works in robust RL, we also test our methods against attackers with a longer planning horizon than the above-mentioned greedy adversaries. In Figure 1, we test under a) the Strategically Timed Attack [20] and b) the Critical Point attack [33]. We observe that across all domains, regret defense outperforms other robustness methods. Particularly, regularization methods fail with increasing rates as the strength or frequency of the attacks increases, even as maximin methods (CARRL) retain some level of robustness. This is one of the main advantages of our proposed methods, as the resulting policies seek robust trajectories to occupy rather than robust single-step action distributions.

6 CONCLUSION

We show that regret can be used to increase the robustness of RL to adversarial observations, even against stronger or previously unseen attackers. We propose an approximation of regret, CCER, and demonstrate its usefulness in proactive value iterative and policy gradient methods, and reactive training under CHT. Our results on a wide variety of problems (more in Appendix B) show that regret-based defense significantly improves robustness against strong observation attacks from both greedy and strategic adversaries. Specifically, RAD-PPO performs the best on average across all (including Appendix B) of our experimental results.

ACKNOWLEDGMENTS

This research/project is supported by the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-RP-2020-017)

REFERENCES

[1] Asrar Ahmed, Pradeep Varakantham, Yossiri Adulyasak, and Patrick Jaillet. 2013. Regret based robust solutions for uncertain Markov decision processes.

[2] Maksym Andriushchenko and Nicolas Flammarion. 2020. Understanding and improving fast adversarial training. *Advances in Neural Information Processing Systems* 33 (2020), 16048–16059.

[3] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. 2021. Recent advances in adversarial training for adversarial robustness. *arXiv preprint arXiv:2102.01356* (2021).

[4] Xueying Bai, Jian Guan, and Hongning Wang. 2019. A model-based reinforcement learning with adversarial training for online recommendation. *Advances in Neural Information Processing Systems* 32 (2019).

[5] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. 2013. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research* 47 (jun 2013), 253–279. https://doi.org/10.1613/jair.3912

[6] Colin F Camerer, Teck-Hua Ho, and Juin-Kuan Chong. 2004. A cognitive hierarchy model of games. *The Quarterly Journal of Economics* 119, 3 (2004), 861–898.

[7] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Chau. 2018. Robust Physical Adversarial Attack on Faster R-CNN Object Detector. *CoRR* abs/1804.05810 (2018). arXiv:1804.05810 http://arxiv.org/abs/1804.05810

[8] Michael Everett, Björn Lütjens, and Jonathan P. How. 2020. Certified Adversarial Robustness for Deep Reinforcement Learning. *CoRR* abs/2004.06496 (2020). arXiv:2004.06496 https://arxiv.org/abs/2004.06496

[9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research* 17, 1 (2016), 2096–2030.

[10] Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. 2019. Adversarial Policies: Attacking Deep Reinforcement Learning. https://doi.org/10.48550/ARXIV.1905.10615

[11] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and Harnessing Adversarial Examples. https://doi.org/10.48550/ARXIV.1412.6572

[12] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. 2017. Adversarial Attacks on Neural Network Policies. https://doi.org/10.48550/ARXIV.1702.02284

[13] Peter Jin, Kurt Keutzer, and Sergey Levine. 2018. Regret minimization for partially observable deep reinforcement learning. , 2342–2351 pages.

[14] Parameswaran Kamalaruban, Yu-Ting Huang, Ya-Ping Hsieh, Paul Rolland, Cheng Shi, and Volkan Cevher. 2020. Robust reinforcement learning via adversarial training with langevin dynamics. *Advances in Neural Information Processing Systems* 33 (2020), 8127–8138.

[15] Daniel Kang, Yi Sun, Tom Brown, Dan Hendrycks, and Jacob Steinhardt. 2019. Transfer of adversarial robustness between perturbation types. *arXiv preprint arXiv:1905.01034* (2019).

[16] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. 2021. Deep reinforcement learning for autonomous driving: A survey.

[17] Jernej Kos and Dawn Song. 2017. Delving into adversarial attacks on deep policies. https://doi.org/10.48550/ARXIV.1705.06452

[18] Edouard Leurent. 2018. An Environment for Autonomous Driving Decision-Making. https://github.com/eleurent/highway-env.

[19] Yongyuan Liang, Yanchao Sun, Ruijie Zheng, and Furong Huang. 2022. Efficient adversarial training without attacking: Worst-case-aware robust reinforcement learning. *Advances in Neural Information Processing Systems* 35 (2022), 22547–22561.

[20] Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. 2017. Tactics of Adversarial Attack on Deep Reinforcement Learning Agents. *CoRR* abs/1703.06748 (2017). arXiv:1703.06748 http://arxiv.org/abs/1703.06748

[21] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards Deep Learning Models Resistant to Adversarial Attacks. https://doi.org/10.48550/ARXIV.1706.06083

[22] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. (2013). https://doi.org/10.48550/ARXIV.1312.5602

[23] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[24] Tuomas Oikarinen, Wang Zhang, Alexandre Megretski, Luca Daniel, and Tsui-Wei Weng. 2021. Robust Deep Reinforcement Learning through Adversarial Loss. (2021). https://openreview.net/forum?id=eaAM_bdW0Q

[25] Anay Pattanaik, Zhenyi Tang, Shuijing Liu, Gautham Bommannan, and Girish Chowdhary. 2017. Robust deep reinforcement learning with adversarial attacks.

[26] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. 2017. Robust adversarial reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2817–2826.

[27] Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. 2017. EPOpt: Learning Robust Neural Network Policies Using Model Ensembles. arXiv:1610.01283 [cs.LG]

[28] Marc Rigter, Bruno Lacerda, and Nick Hawes. 2021. Minimax Regret Optimisation for Robust Planning in Uncertain Markov Decision Processes. , 11930-11938 pages. https://doi.org/10.1609/aaai.v35i13.17417

[29] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. https://doi.org/10.48550/ARXIV.1707.06347

[30] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free! *Advances in Neural Information Processing Systems* 32 (2019).

[31] Ali Shafahi, Mahyar Najibi, Zheng Xu, John Dickerson, Larry S Davis, and Tom Goldstein. 2020. Universal adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5636–5643.

[32] Steven Spielberg, Aditya Tulsyan, Nathan P Lawrence, Philip D Loewen, and R Bhushan Gopaluni. 2019. Toward self-driving processes: A deep reinforcement learning approach to control. , e16689 pages.

[33] Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma, Yan Zheng, Kangjie Chen, and Yang Liu. 2020. Stealthy and Efficient Adversarial Attacks against Deep Reinforcement Learning. , 5883-5891 pages. https://doi.org/10.1609/aaai.v34i04.6047

[34] Yanchao Sun, Ruijie Zheng, Yongyuan Liang, and Furong Huang. 2023. Who Is the Strongest Enemy? Towards Optimal and Efficient Evasion Attacks in Deep RL. arXiv:2106.05087 [cs.LG]

[35] Kai Liang Tan, Yasaman Esfandiari, Xian Yeow Lee, Soumik Sarkar, et al. 2020. Robustifying reinforcement learning agents via action space adversarial training. In *2020 American control conference (ACC)*. IEEE, 3959–3964.

[36] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5026–5033. https://doi.org/10.1109/IROS.2012.6386109

[37] Eric Wong, Leslie Rice, and J Zico Kolter. 2020. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994* (2020).

[38] Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. 2020. Robust Deep Reinforcement Learning against Adversarial Perturbations on State Observations. https://doi.org/10.48550/ARXIV.2003.08938