Robust Deep Reinforcement Learning through Bootstrapped Opportunistic Curriculum

Junlin Wu [1] **Yevgeniy Vorobeychik** [1]

Abstract

Despite considerable advances in deep reinforcement learning, it has been shown to be highly vulnerable to adversarial perturbations to state observations. Recent efforts that have attempted to improve adversarial robustness of reinforcement learning can nevertheless tolerate only very small perturbations, and remain fragile as perturbation size increases. We propose *Bootstrapped Opportunistic Adversarial Curriculum Learning (BCL)*, a novel flexible adversarial curriculum learning framework for robust reinforcement learning. Our framework combines two ideas: conservatively bootstrapping each curriculum phase with highest quality solutions obtained from multiple runs of the previous phase, and opportunistically skipping forward in the curriculum. In our experiments we show that the proposed BCL framework enables dramatic improvements in robustness of learned policies to adversarial perturbations. The greatest improvement is for Pong, where our framework yields robustness to perturbations of up to 25/255; in contrast, the best existing approach can only tolerate adversarial noise up to 5/255. Our code is available at: https://github.com/jlwu002/BCL.

1. Introduction

Advances in reinforcement learning coupled with state of the art deep neural network-based representations have led to breakthroughs in a broad range of applications, including the AlphaZero general game-playing approach (Silver et al., 2018), autonomous driving (Kiran et al., 2021), navigation of stratospheric baloons (Bellemare et al., 2020), medical imaging (Zhou et al., 2021), and many others. However, a series of recent efforts demonstrated that policies learned by deep reinforcement learning (DRL) can be extremely fragile to small adversarial perturbations to input state observations (Lin et al., 2017; Sun et al., 2020; Wu et al., 2021; Zhang et al., 2021). Indeed, this echoes a broader pattern of fragility of neural network architectures to adversarial perturbation attacks (Athalye et al., 2018; Eykholt et al., 2018; Carlini & Wagner, 2017; Goodfellow et al., 2015; Szegedy et al., 2014; Vorobeychik & Kantarcioglu, 2018).

In turn, a series of efforts have emerged aiming to improve robustness of deep neural networks for supervised learning (Cai et al., 2018; Cohen et al., 2019; Raghunathan et al., 2018; Goodfellow et al., 2015; Madry et al., 2018; Vorobeychik & Kantarcioglu, 2018), as well as deep reinforcement learning (Oikarinen et al., 2021; Zhang et al., 2020; 2021). However, while variations of adversarial training have proved relatively successful at attaining robustness of deep neural networks in supervised settings, success has been more modest in reinforcement learning, where the best approaches can tolerate only very small-magnitude perturbations (e.g., up to 5/255 in Pong, which was achieved in Oikarinen et al. (2021) through RADIAL-A3C training).

We propose a novel curriculum learning framework, *Bootstrapped Opportunistic Adversarial Curriculum Learning (BCL)* to boost robustness of DRL. Our approach is inspired by recent successful curriculum learning approaches in adversarial supervised learning (Balaji et al., 2019; Cai et al., 2018; Sitawarin et al., 2021), but also differs substantively from these. In particular, both Balaji et al. (2019) and Cai et al. (2018) propose to construct a simple curriculum of increasing input difficulty; this is what we call *naive curriculum learning* below, and we show that it is not particularly effective in achieving robustness in DRL. Sitawarin et al. (2021) propose an adaptive curriculum by customizing difficulty to specific inputs as a function of attack success on each input. This idea, however, is not meaningful in DRL, where inputs are states and success is measured in terms of overall reward of a policy, rather than accuracy of predictions on individual inputs. Our approach is also inspired by recent success of curriculum learning approaches in reinforcement learning (Narvekar et al., 2017; 2020); however,

[1]Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO, USA. Correspondence to: Junlin Wu <junlin.wu@wustl.edu>, Yevgeniy Vorobeychik <yvorobeychik@wustl.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

ours is the first curriculum learning framework for *adversarial* reinforcement learning.

In the proposed BCL framework, we leverage two key ideas. First, we bootstrap each phase of the curriculum by ensuring that the result of the previous phase is successful, which we do by choosing the best result over multiple adversarial training runs. Second, we introduce adaptivity by opportunistically skipping forward in the curriculum if we find that the model learned in the current phase is already robust to the adversarial perturbations with higher magnitude.

We evaluate the efficacy of the proposed BCL framework in boosting robustness of DQN-style approaches with minimal reduction in nominal (non-adversarial) reward through extensive experiments on the Pong, Freeway, BankHeist, and RoadRunner OpenAI domains. In all cases, we show that BCL yields considerable improvements in robustness compared to the state of the art. In Pong, BCL-trained policies achieve near-flawless performance under adversarial perturbations of up to $\epsilon = 25/255$; in comparison, the state-of-the-art RADIAL-DQN performs poorly even with $\epsilon = 5/255$ (the reward under 30-step PGD attack is -17.7). In BankHeist, BCL training achieves an order-of-magnitude higher robustness for $\epsilon = 15/255$ than state of the art (SA-DQN), and for RoadRunner, it is several orders of magnitude better (RADIAL-DQN is state of the art).

In summary, we make the following contributions:

1. A novel flexible adversarial curriculum learning framework for reinforcement learning (BCL), in which bootstrapping each phase from multiple executions of previous phase plays a key role,
2. A novel opportunistic adaptive generation variant that opportunistically skips forward in the curriculum,
3. An approach that composes interval bound propagation and FGSM-based adversarial input generation as a part of adaptive curriculum generation, and
4. An extensive experimental evaluation using OpenAI Gym Atari games (main paper) and Procgen (Appendix A) that demonstrates significant improvement in robustness due to the proposed BCL framework.

2. Related Work

Robustness to adversarial perturbations has been a subject of considerable attention in machine learning broadly, although much of the focus, and the most significant progress, has been specifically in supervised learning (Athalye et al., 2018; Eykholt et al., 2018; Carlini & Wagner, 2017; Goodfellow et al., 2015; Szegedy et al., 2014; Vorobeychik & Kantarcioglu, 2018). In particular, in the supervised learning settings, adversarial training has emerged as a major paradigm for enhancing robustness (Cai et al., 2018; Cohen et al., 2019; Raghunathan et al., 2018; Goodfellow et al., 2015; Madry et al., 2018; Tong et al., 2019; Vorobeychik & Kantarcioglu, 2018; Wu et al., 2020).

Studies of adversarial state perturbations to policies learned using deep reinforcement learning are somewhat more recent (Behzadan & Munir, 2017; Kos & Song, 2017; Pattanaik et al., 2018; Wu et al., 2021), as are approaches for increasing robustness (Akkaya et al., 2019; Fortunato et al., 2017; Oikarinen et al., 2021; Pattanaik et al., 2018; Tobin et al., 2017; Zhang et al., 2020; 2021). Adversarial training techniques, using either lower or upper bounds on adversarial loss have been explored, but the efficacy of conventional adversarial training has been somewhat limited, with success restricted to weak FGSM attacks, or relatively small-size perturbations (Behzadan & Munir, 2017; Kos & Song, 2017; Pattanaik et al., 2018). A number of heuristic techniques, such as adding noise at training, have also been proposed (Akkaya et al., 2019; Fortunato et al., 2017; Tobin et al., 2017), but these are generally not as effective against strong attacks as those based on adversarial training. An orthogonal idea that attempts to introduce robustness directly at decision time is CARRL (Everett et al., 2021). However, their reliance on linear bounds makes it only suitable for low-dimensional settings (Weng et al., 2018). Among the most recent and most effective of approaches based on forms of adversarial training are RADIAL (Oikarinen et al., 2021) and SA-DQN (Zhang et al., 2020), and we compare to these directly. Finally, CROP is a recent approach for certifying robustness of deep reinforcement learning methods (Wu et al., 2022). However, CROP is not in itself a method for *improving* DRL robustness, either empirical or certified.

Our approach builds on prior work on the use of curriculum learning in adversarial settings (Balaji et al., 2019; Cai et al., 2018; Sitawarin et al., 2021), as well as curriculum learning in supervised (Bengio et al., 2009) and reinforcement (Narvekar et al., 2017; 2020) learning. However, as elaborated in the introduction, ours is the first *adversarial* curriculum learning framework in the reinforcement learning context with a particular attention to how to design a curriculum; prior approaches for adversarial curriculum learning either do not consider a curriculum design question, or are not applicable in reinforcement learning where efficacy depends on the process dynamics and cannot be evaluated independently for each input.

3. Preliminaries

In this section we introduce the basics of deep reinforcement learning (DRL), focusing primarily on Deep Q-learning that we leverage in the proposed BCL framework.

3.1. Deep Reinforcement Learning

Reinforcement learning models the world as a Markov Decision Process (MDP). An MDP is a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(s'|s,a)$ the (in our setting, unknown) transition function that determines the distribution of the next state s' given current state s and action a, and $R(s,a)$ the expected reward function obtained from taking action a in state s. Finally, $\gamma \in [0,1)$ is the temporal discount factor. Solving MDPs amounts to computing either the Q function, $Q(s,a)$, which is the maximum discounted sum of rewards that can be achieved starting in state s and taking an action a, or the value function $V(s) = \max_a Q(s,a)$. A solution to an MDP is a policy $\pi(s) \in \arg\max_a Q(s,a)$.

In *deep reinforcement learning (DRL)*, a key step is to approximate the value function, Q function, and/or policy using a deep neural network. Algorithms differ both in which of these they approximate, and the particular ways these are learned from experience. We focus on *Deep Q-Network (DQN)*, a class of approaches that learn a parametric representation of the Q function.

Specifically, DQN approximates the Q function using a deep neural network $Q(s,a;\theta)$ with parameters θ. A basic DQN learning algorithm learns $Q(s,a;\theta)$ by using the loss function

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,s',r)}\left[\left(r + \gamma \max_{a'} Q\left(s', a'; \theta\right) - Q(s,a;\theta)\right)^2\right].$$

We make use of several improvements on DQN: Double DQN (Van Hasselt et al., 2016) and Dueling DQN (Wang et al., 2016). Double DQN uses two Q-networks with Q_{target} for evaluation and Q_{actor} for training, with the loss function

$$\mathcal{L}\left(\theta_{\text{actor}}\right) = \mathbb{E}_{(s,a,s',r)}\left[\left(r + \gamma \max_{a'} Q_{\text{target}}(s', a'; \theta_{\text{target}}) - Q_{\text{actor}}(s,a;\theta_{\text{actor}})\right)^2\right]. \quad (1)$$

Dueling DQN is based on Double DQN and uses two estimators, one for state value function estimation (i.e., $V_Q(s)$), and one for the state-dependent action advantage function estimation $A_{Q(s,a)}$, with $Q(s,a) = V_Q(s) + A_{Q(s,a)}$.

3.2. Adversarial Deep RL

Adversarial Policy Perturbations In adversarial perturbation attacks on DRL, an adversary adds a perturbation δ to each observed state s constrained to be $\|\delta\|_p \leq \epsilon$ (for exogenously specified l_p norm and ϵ) so as to minimize expected discounted reward of the executed policy $\pi(s)$. We take $p = \infty$ here, as is common. If the policy is based on maximizing the learned Q function, as in DQN and its variants, the attack aims to perturb this function, indirectly affecting the policy, while if DRL is based on policy learning (e.g., actor-critic), with the policy itself represented by a neural network $\pi(s;\theta)$, the policy is attacked directly. Specifically, a common attack on DQN aims maximize $\mathcal{L}(\text{Softmax}(Q(s+\delta;\theta)), \pi(s))$ with respect to δ, where $\mathcal{L}$ is the cross-entropy loss, $Q(s)$ is the vector of Q values over all actions in state s. A PGD (projected gradient descent) attack (Madry et al., 2018) is then implemented with this loss function, which updates δ iteratively: $\delta_{k+1} \leftarrow \delta_k + \alpha \cdot \text{sign}(\nabla_\delta \mathcal{L}(Q(x+\delta_k;\theta), \pi(s)))$ over a fixed number of iterations, projecting to a nearest feasible state and clipping to ensure that $\|\delta\|_\infty \leq \epsilon$. In policy learning methods, a common loss function is instead $\mathcal{L}(\pi(s+\delta;\theta), \pi(s))$, with PGD attacks implemented just as above. An important special class of PGD is FGSM (fast gradient sign method) (Goodfellow et al., 2015), in which PGD is executed for only a single iteraration and $\alpha = \epsilon$.

Adversarial Training Deep RL is robust to adversarial policy perturbations with magnitude up to ϵ if attacks do not significantly reduce the discounted sum of rewards. Common approaches aimed at robust learning in general use some form of adversarial training, where after initially training the model in the regular manner, additional training phases either add adversarial perturbations to inputs that are used in further gradient updates (Madry et al., 2018), or take gradients of an upper bound on adversarial loss (Wong et al., 2018). A state-of-the-art form of adversarial training uses PGD attacks to generate adversarial perturbations (Madry et al., 2018). A recent alternative which is much more computationally efficient and equally efficacious uses FGSM with random initializations instead (Wong et al., 2020); henceforth, we term this variant RI-FGSM.

In robust DQN, a recent RADIAL-DQN approach (Oikarinen et al., 2021) on which we build defines the loss function as

$$\mathcal{L}_{\text{RADIAL}} = \kappa \mathcal{L}_{\text{standard}} + (1-\kappa)\mathcal{L}_{\text{adv}}, \quad (2)$$

where $\mathcal{L}_{\text{standard}}$ is defined in Equation (1), and

$$\mathcal{L}_{\text{adv}}\left(\theta_{\text{actor}}, \epsilon\right) = \mathbb{E}_{(s,a,s',r)}\left[\sum_y \mathcal{L}_y(s,a)\right],$$

with $\mathcal{L}_y(s,a) = \left(r + \gamma \max_{a'} Q_{\text{target}}\left(s', a'\right) - \tilde{Q}^{\epsilon}_{\text{actor}}(s,y)\right)^2$ when $y = a$ and $\mathcal{L}_y(s,a) = \left(Q_{\text{actor}}(s,y) - \tilde{Q}^{\epsilon}_{\text{actor}}(s,y)\right)^2$ otherwise. This is referred to as approach # 1 in Oikarinen et al. (2021), which yields a strict upper bound on the loss function under adversarial perturbation, that is, $\mathcal{L}_{\text{standard}}(s+\delta;\epsilon) \leq \mathcal{L}_{\text{adv}}(s;\epsilon)$ with $||\delta||_p \leq \epsilon$; approach # 2 is an alternative that aims to minimize the weighted overlapped IBP Q-values. Since approach # 2 empirically outperforms approach # 1 for RADIAL-DQN, we use approach # 2 in our experiments below for RADIAL

curriculum training. RADIAL-DQN (both approaches) uses $\tilde{Q}^{\epsilon}_{\text{actor}}(s, y)$ which is an upper or lower bound on $Q_{\text{actor}}(s, y)$ derived using interval bound propagation (IBP) for a given attack budget ϵ. As IBP yields relatively loose bounds, ϵ must of necessity be small for these to be meaningful, limiting the ability to achieve robustness beyond relatively low values of ϵ.

4. Bootstrapped Opportunistic Adversarial Curriculum Learning

Curriculum learning is an old concept in iterative learning in which easier examples are provided before more challenging examples (Bengio et al., 2009). A natural alternative is to start with small values of ϵ and gradually increase these during adversarial training. However, as we show in the experiments, this latter idea works extremely poorly for DRL. We propose a novel *Bootstrapped Opportunistic Adversarial Curriculum Learning (BCL)* framework for iterative adversarial training. The key idea is to bootstrap each training step to ensure that subsequent iterations begin with a partially robust baseline, and to also enable the algorithm to "skip forward" if robustness against several successive values of ϵ has already been achieved. Our BCL framework allows one to explicitly trade off between being conservative (paying more attention to the former) and opportunistic (greater focus on the latter).

4.1. The BCL Algorithm

At the high level, the proposed BCL algorithmic framework begins by creating a *baseline curriculum*, that is, an increasing sequence of L attack budgets $\{\epsilon_i\}$, with $\epsilon_1 < \epsilon_2 < \cdots < \epsilon_L$, where $\epsilon_L = \epsilon$ is our target robustness level. It also begins with a sufficiently small $\epsilon_0 > 0$ so that it is either already achievable (e.g., by standard DRL, prior art such as RADIAL, etc) or not difficult to achieve; we assume that BCL is initially *bootstrapped* with a model f_{θ_0} that is indeed able to achieve this relatively low bar. It then proceeds through a series of phases, where a phase is associated with attempting to achieve robustness against ϵ_i in the curriculum for some i (which is not necessarily identical to the phase number, as we discuss below). In each phase, we run adversarial training (AT) up to K times, where each AT run is bootstrapped by the best model obtained thus far, f_θ. Each model thereby learned is then independently evaluated, and if the best model obtained thus far in the current phase exhibits sufficiently good performance (a criterion for this can depend on ϵ_i, and represented by a function $\overline{V}(\epsilon)$ in Algorithm 1), we can stop and move to the next phase as long as we performed at least a minimum number $K_{\min}$ AT runs. The best model in the current phase then becomes the best model achieved thus far, updating f_θ. Algorithm 1 describes this procedure more precisely.

Algorithm 1 BCL algorithm.

Input: ϵ, K, $K_{\min}$, $\overline{V}(\epsilon)$, f_{θ_0}.
$f_\theta \leftarrow f_{\theta_0}$ // Initialization
$\{\epsilon_i\}_{i=1}^{L} \leftarrow \text{Curriculum}(\epsilon)$ // Create curriculum
$(i, \epsilon_{\text{best}}) \leftarrow \text{ChooseNext}(f_\theta, \{\epsilon_i\}, 0, \overline{V}(\epsilon))$
while $\epsilon_{\text{best}} < \epsilon$ **do**
  for $k = 1, \ldots, K$ **do**
    $f_{\theta_k} \leftarrow \text{Train}(f_\theta, \epsilon_i)$
    $V_k \leftarrow \text{Eval}(f_{\theta_k}, \epsilon_i)$
    if $k \geq K_{\min}$ and $V_k \geq \overline{V}(\epsilon_i)$ **then**
      break
    end if
  end for
  // Find the best model among training results
  $k^* \leftarrow \arg\max_{k \in [K]} V_k$
  $f_\theta \leftarrow f_{\theta_{k^*}}$
  $(i, \epsilon_{\text{best}}) \leftarrow \text{ChooseNext}(f_\theta, \{\epsilon_i\}, i, \overline{V}(\epsilon))$
end while
return f_θ

The next central feature of BCL is the ability to *skip forward* in the curriculum, omitting the next budget level ϵ_{i+1}, and potentially others after it, as shown in the ChooseNext step (Algorithm 2). The most we can skip forward is to the smallest ϵ_j to which the current model is *not robust* (this is the purpose of EvalRobust function in Algorithm 2). This skipping feature is most useful because it significantly reduces the time that BCL needs to run, but as we show in the experiments, there are times where it also yields better robustness than obtained by following the baseline curriculum.

Algorithm 2 ChooseNext

Input: $f_\theta, \{\epsilon_i\}_{i=1}^{L}, j, \overline{V}(\epsilon)$.
// Find smallest i such that f_θ is *not* robust for ϵ_i
$i \leftarrow \text{EvalRobust}(f_\theta, \{j+1, \ldots, L\}, \overline{V}(\epsilon))$
// Select index l to train with next
$l \leftarrow \text{Select}(\{j+1, \ldots, i\})$
return (l, ϵ_{l-1})

Algorithm 1 takes as input a fixed target ϵ that we wish to induce robustness to, but in practice it is often the case that we wish to be more opportunistic, and simply observe what is possible in trading off robustness and baseline (non-adversarial) efficacy. For example, we can set ϵ to be very high, but stop BCL well in advance of reaching it if we observe significant performance degradation.

For the RADIAL curriculum training, unlike training with adversarial examples (Section 4.2), it does not have a target ϵ to be robust against. We choose to always follow the baseline curriculum for the ϵ. We find that RADIAL training

does not increase the nominal reward in trend, and many times the significant decrease in nominal reward is accompanied by the decease in robustness as the model begin to collapse. Thus, we aim at maintaining the nominal reward at a high level. We set the a threshold for each model, and re-train the model for maximum K times if the nominal reward is below the threshold. We stop the training if nominal reward is below the threshold level for M consecutive curriculum phases.

Next, we illustrate the BCL framework with several special cases, noting first that both conventional adversarial training and naive curriculum learning can also be viewed as variants of BCL.

Adversarial Training (AT): Standard adversarial training can be viewed as a special case of BCL if $K = 1$ and the baseline curriculum is simply the singleton ϵ.

Naive Curriculum Learning (NCL): Setting $K = 1$ and always following the baseline curriculum (i.e., the next index returned by the $\mathrm{ChooseNext}$ function is always $i + 1$) recovers a naive implementation of curriculum learning.

Conservatively Bootstrapped Curriculum Learning (BCL-C): If we set $K_{\min} = K$ and always follow the baseline curriculum, BCL never opportunistically skips forward, and setting K sufficiently high ensures that each step is bootstrapped with an effective model trained using all smaller values of ϵ.

Maximum Opportunistic Skipping (BCL-MOS): If we always choose to skip to the smallest ϵ against which the current model f_θ is not (yet) robust, we obtain the most opportunistic version of the algorithm.

4.2. Generating Adversarial Perturbations

The key question left open in BCL is precisely how we train a model in a particular phase to be robust against a given adversarial budget ϵ. There are two major ways to do this: using bounds on the impact of adversarial perturbations, such as those produced by IBP, as done by RADIAL (Oikarinen et al., 2021), and using adversarial perturbations (Zhang et al., 2020; 2021). In addition to using IBP, RADIAL introduces a crucial insight in robustness training in distinguishing updates for actions that have been chosen (for which the immediate reward has been observed) from those that have not been, as discussed in Section 3.2. We leverage this idea, but replace IBP with adversarial examples. Next, we present a novel approach for generating adversarial examples for adversarial training in each phase of BCL that specifically leverages DDQN.

Recall that in RADIAL-DQN (approach # 1), $\tilde{Q}^{\epsilon}_{\text{actor}}(s, a)$ uses IBP bounds on the Q function that can be achieved through adversarial perturbations. Alternatively, we can define it as $\tilde{Q}^{\epsilon}_{\text{actor}}(s, a) = Q_{\text{actor}}(s + \delta^*, a)$, where δ^* (approximately) solves the following optimization problem:

$$\min_{||\delta||_\infty \leq \epsilon} \sum_{a \in A} \pi(s + \delta, a) Q_{\text{target}}(s, a), \tag{3}$$

where ϵ is the bound on l_∞-norm of the perturbation (as is common in prior literature on robust reinforcement learning), with $\pi(s + \delta, a) = 1$ iff a is the best action to be taken after observing $s + \delta$, i.e., $a \in \arg\max_{a'} Q_{\text{actor}}(s + \delta, a')$, and $\pi(s + \delta, a) = 0$ otherwise. In other words, Equation (3) aims to identify δ that minimizes the expected discounted sum of rewards as approximated by $Q_{\text{target}}(s, a)$. Note that here it is crucial to separate the Q_{actor}, which determines the policy, and Q_{target}, which serves as an "objective" evaluation of state-action values. This is in contrast with typical adversarial perturbation attacks on DRL described in Section 3.2, where the adversary merely aims to prevent a target (optimal) action from being chosen, but may well incentivize DRL to choose a near-optimal action instead.

In order to approximately solve the problem in Equation (3), we first replace a policy π by its differentiable approximation $\tilde{\pi}$, where

$$\tilde{\pi}(s, a) = \frac{e^{Q_{\text{actor}}(s,a)}}{\sum_{a'} e^{Q_{\text{actor}}(s,a')}}.$$

Equivalently, $\tilde{\pi}(s + \delta) = \mathrm{Softmax}(Q_{\text{actor}}(s + \delta))$, where we use $Q(s)$ to denote a vector with values for each action a. We then solve the following proxy optimization problem to approximate δ^*:

$$\min_{||\delta||_\infty \leq \epsilon} \mathrm{Softmax}(Q_{\text{actor}}(s + \delta)) \odot Q_{\text{target}}(s), \tag{4}$$

with $\odot$ denoting the dot-product.

Commonly, the problem in Equation (4) is solved using PGD (Madry et al., 2018). However, this becomes a major bottleneck in training, particularly when we use a large number of PGD iterations. We make two improvements to significantly reduce the time associated with computing δ. First, we use FGSM + Random Initialization (RI-FGSM) (see Section 3.2), proposed by Wong et al. (2020) for supervised adversarial training, for which it was shown highly effective. Ours is the first application of this idea in robust DRL. Second, we dynamically calculate the perturbation δ and push the entire tuple (s, a, s', r, δ) (i.e., including δ) into the replay buffer. This enables us to re-utilize the previously calculated perturbations to further improve training efficiency. Consequently, we chose a relatively small replay buffer size to ensure the perturbations δ stored in the buffer are frequently updated as the DRL model evolves.

With RADIAL-DQN (approach # 2), which minimize the weighted overlapping IBP Q-values, as well as the approach above for generating *specific* adversarial perturbations, which yields a lower bound on adversarial loss, we

have two specific ways that we can use to compute gradient updates in the Train step of BCL for a given perturbation magnitude ϵ. We refer to the former simply as RADIAL, and to the latter as AT (for adversarial training). Both can be "plugged in" to any variant of BCL. Additionally, we can *compose* these approaches, giving rise to a novel variant:

RADIAL + AT Bootstrapped Curriculum Learning (BCL-RADIAL+AT): First, run BCL-RADIAL until it reaches a point in the curriculum at which its performance degrades significantly; then, switch to BCL-X-AT (where X is either C or MOS) for the remainder of the curriculum.

5. Experiments

5.1. Experiment Setup

We evaluate the proposed approach using four Atari-2600 games from the OpenAI Gym (Bellemare et al., 2013): Pong, Freeway, BankHeist, and RoadRunner. Those environments have discrete action space. The walltime for all experiments are documented in the Appendix E. We use R_{nominal} to denote a model's nominal reward (i.e., average discounted sum of per-step rewards without adversarial perturbations), and $R^{\epsilon}_{\text{adv}}$ to represent a model's reward under adversarial attacks with l_∞ perturbation bounded by ϵ. For each model we calculate a score using $R_{\text{nominal}} + \frac{1}{3}\sum_{\epsilon} R^{\epsilon}_{\text{adv}}$ for all ϵ listed in Table 2 to measure the model's robustness level, and this score is used to choose the median and best final result (out of three independent runs); we present the median here, and the results of all runs, as well as the best are provided in the Appendix D. We experiment all the BCL variations in Section 4.1. We compare BCL-based approaches to six benchmarks: 1) standard Dueling DQN training (DQN (Vanilla)), 2) SA-DQN using convex relaxation (SA-DQN (Convex)) (Zhang et al., 2020), 3) RADIAL-DQN (Oikarinen et al., 2021), 4) standard adversarial training (AT-DQN) (Madry et al., 2018), 5) naive curriculum learning with adversarial examples (NCL-AT-DQN) (Cai et al., 2018; Sitawarin et al., 2021) and 6) naive curriculum learning with RADIAL method (NCL-RADIAL-DQN). For DQN (Vanilla) we use the results from Zhang et al. (2020), and for AT-DQN, NCL-AT-DQN as well as NCL-RADIAL-DQN we perform our own training as three restricted variants of the BCL algorithm. The AT method is the one we purposed in Section 4.2. The adversarial examples for all games are generated using RI-FGSM.

DQN Hyperparameters Our implementation is based on RADIAL-DQN (Oikarinen et al., 2021). For most hyperparameters we keep them the same as in RADIAL-DQN, with a few exceptions such as replay initial and replay buffer size, which are modified according to our model setting to improve training efficiency. We use buffer size 50,000 across all environments compared to 200,000 used by RADIAL. For replay initial we use 256 compared to 50,000 in RADIAL. We use RI-FGSM (Algorithm 3 in Wong et al. (2020), see Section 3.2) with hyperparameter $\alpha = 0.375$ for approximating δ during training. The detailed DQN specific hyperparameters for AT runs are in Table 1. The one exception is BCL-RADIAL+AT-DQN for RoadRunner environment: for the AT training we use 1.25×10^{-7} as the learning rate, as we find with learning rate 0.000125 the nominal reward would decrease significantly after training. For NCL/BCL-RADIAL-DQN, all hyperparameters are the same as in RADIAL-DQN.

To ensure a fair comparison, we let all methods to have the same computational constraints and evaluation metrics: for all environments we train for 4.5 million frames (same as RADIAL-DQN) for each run, evaluate over 20 test episodes and report the averaged reward.

Table 1. DQN specific hyperparameters (AT runs)

PARAMETER	VALUE
DISCOUNT FACTOR (γ)	0.99
BUFFER SIZE	50000
REPLAY INITIAL	256
BATCH SIZE	128
OPTIMIZER	ADAM
OPTIMIZER LEARNING RATE	0.000125

Adversarial Attacks for DQN As we observe significant issues with obfuscated gradients with NCL/BCL-RADIAL-DQN, we apply four types of adversarial attacks for DQN models: 1) 30-step untargeted PGD attack with step size 0.1 (this is stronger than the 10-step PGD used in Oikarinen et al. (2021)); 2) RI-FGSM ($\alpha = 0.375$); 3) RI-FGSM (Multi): sample $N = 1000$ random starts for RI-FGSM, and takes the first sample where the resulting adversarial example alters the action; 4) RI-FGSM (Multi-T): sample $N = 1000$ random starts for RI-FGSM, and takes the sample which results the agent taking the action corresponding to the lowest Q value among those N samples. We report the lowest reward obtained after running those four attacks. We observe that with obfuscated gradients, RI-FGSM (Multi-T) results in the strongest attack in many cases, while 30-step PGD is typically stronger otherwise (see the Appendix D for details).

Hyperparameters for AT-DQN and NCL-AT-DQN For AT-DQN, we experiment with a series of varying values of ϵ, and present the most effective results, with the comprehensive results deferred to Appendix D. We generate adversarial perturbations as in Section 4.2, and use DQN (Vanilla) as f_{θ_0},[1] setting $K = L = 1$. For each environment we only

[1]For the RoadRunner environment we used the implementation of vanilla DQN from RADIAL-DQN (version 1) as f_{θ_0}, which yields better results.

need one run which is 4.5 million frames.

For naive curriculum learning (NCL-AT-DQN), we use DQN (Vanilla) as f_{θ_0}, as for AT-DQN. We set $K = 1$ and $\epsilon_0 = 0$, with curriculum increment of 1/255 (i.e., using the baseline curriculum) until target ϵ is reached. The choice of ϵ of each environment is the same as the ones in BCL experiments. However, because NCL-AT-DQN significantly underperforms BCL and (unlike BCL) when the target ϵ is reached the final NCL-AT-DQN model performs extremely poorly, we instead report the best result along the curriculum path to provide the strongest benchmark.

Hyperparameters for NCL/BCL-RADIAL-DQN In NCL/BCL-RADIAL-DQN experiments, we bootstrap from RADIAL-DQN. We set $\epsilon_0 = 1/255$, with ϵ always follows the baseline curriculum. The increments of the baseline curriculum is $1/255$. We set $K = 1$ for NCL-RADIAL-DQN and $K = 3$ for BCL-RADIAL-DQN. We perform maximum K runs for each curriculum phase; if none of the K run results has a nominal reward above the threshold, we choose the one with the highest nominal reward and move to the next curriculum phase. We stop the training if nominal reward is below the threshold for $M = 2$ consecutive curriculum phases. The thresholds are shown in Table 10 in Appendix C. We report the best result along the curriculum path for NCL/BCL-RADIAL-DQN.

Hyperparamters for BCL-RADIAL+AT-DQN For BCL-RADIAL+AT-DQN, we choose the median run among three BCL-RADIAL-DQN runs and perform BCL-C-AT-DQN starting with this run. As we are only able to perform BCL-C-AT-DQN further for BankHeist and RoadRunner, we choose $K = 3$ for BankHeist, and $K = 1$ for RoadRunner. The baseline curriculum for BankHeist starts from $\epsilon_0 = 13/255$ and for RoadRunner $\epsilon_0 = 12/255$, with curriculum increment of $1/255$, and target $\epsilon = 15/255$.

Other BCL Hyperparameters In BCL, we also evaluate two concrete novel instantiations of the proposed BCL framework: conservatively bootstrapped curriculum learning (BCL-C-AT-DQN), and maximum opportunistic skipping (BCL-MOS-AT-DQN). We bootstrapped all instances by using RADIAL-DQN as f_{θ_0}.[2] Further, the baseline curriculum is created starting with $\epsilon_0 = 3/255$, since RADIAL-DQN (which we use as f_{θ_0}) is already robust up to $3/255$. The baseline curriculum is then created by using increments of $1/255$ until reaching the target ϵ. The BCL hyperparameters (K, $K_{\min}$ and ϵ) are listed in Table 11 in the Appendix C. The thresholds $\overline{V}(\epsilon)$ for BCL-MOS-AT-DQN are listed in Table 12 in the Appendix C, where $\overline{V}_{\text{nominal}}(\epsilon)$ is the threshold for nominal reward, and $\overline{V}_{\text{adv}}(\epsilon)$ is the threshold for rewards under adversarial attacks. As described in Section 4, if the model in phase i is trained against ϵ_i and $R_{\text{nominal}} \geq \overline{V}_{\text{nominal}}(\epsilon)$, we perform evaluation with adversarial attacks, find the maximum $j > i$ such that $R^{\epsilon_j}_{\text{adv}} \geq \overline{V}_{\text{adv}}(\epsilon)$, and skip forward in the baseline curriculum, training with ϵ_{j+1} in the next phase. In our implementation of BCL, we further smoothed the curriculum by gradually increasing the upper bound ϵ on adversarial perturbations from ϵ_i to ϵ_{i+1} in phase $i+1$ during the 4.5 million training frames. The function evaluating the quality of intermediate results in Algorithm 1, $\text{Eval}(f_{\theta_k}, \epsilon_i)$, returns the efficacy score $V_k = R_{\text{nominal}} + \frac{1}{2}(R^{\epsilon_i}_{\text{adv}} + R^{\epsilon_{i-1}}_{\text{adv}})$, which allows us to choose the best model among all the intermediate results. Note that it is crucial to include R_{nominal} as a part of the criterion for model selection, as a model with a high nominal reward tends to show considerably better stability in subsequent curriculum training.

We used a time-varying κ in Equation 2 for BCL. Specifically, we let κ decrease from 1 to 0.5 through the 4.5 million training frames for all experiments except when we use RADIAL-DQN as f_{θ_0} in Pong, Freeway and BankHeist; or when we use RADIAL-DQN (version 2) as f_{θ_0} in RoadRunner (i.e., NCL/BCL-RADIAL-DQN and BCL-RADIAL+AT-DQN). In these cases, κ is set to 0.8 throughout training. The choice of $\kappa = 0.8$ ensured consistency with the κ used in RADIAL-DQN (Oikarinen et al., 2021), which makes the bootstrapping process more stable.

5.2. Results

Our main results are presented in Table 2, with extensive additional results and analysis provided in the Appendix. We can readily observe that the novel instantiations of BCL outperform all benchmarks in terms of robustness in Pong, Freeway and BankHeist. The improvement for higher levels of ϵ is often dramatic.

For Pong, we observe that both BCL-C-AT-DQN and BCL-MOS-AT-DQN significantly outperform all the benchmark models as well as BCL-RADIAL-DQN for $\epsilon \geq 20/255$, and achieves a near flawless reward. This demonstates the value of our BCL framework as well as the AT curriculum learning approach.

In the Freeway setting, both BCL-MOS-AT-DQN and BCL-RADIAL-DQN achieve high robustness for ϵ up to $20/255$. In terms of benchmark models, while SA-DQN is competitive at $\epsilon = 20/255$, it is far worse at lower levels of ϵ; for example, when $\epsilon = 10/255$, BCL-MOS-AT-DQN achieves an average reward that is more than 50% higher than either SA-DQN or RADIAL-DQN, with DQN (Vanilla) achieving 0 reward at such levels of adversarial perturbations.

Note that we were unable to perform BCL-RADIAL+AT-

[2] For the RoadRunner environment we used the version 1 implementation of RADIAL-DQN (Oikarinen et al., 2021) as f_{θ_0}, which yields better bootstrapping performance. Nevertheless, we always use the best-performing version of RADIAL-DQN (version 2) as the benchmark in Table 2.

Table 2. Average episode rewards ± standard error of the mean (SEM) over 20 episodes. The gray rows are the most robust models (selected based on score $R_{\text{nominal}} + \frac{1}{3}\sum_{\epsilon} R^{\epsilon}_{\text{adv}}$). **Boldface** marks the best results for each value of ϵ, including $\epsilon = 0$ (nominal); we marked multiple row entries as boldface for a given ϵ if they are statistically indistinguishable (i.e., have overlapping confidence intervals).

PONG

METHOD/METRIC	NOMINAL	30-STEP PGD/RI-FGSM ATTACK		
ϵ	0	10/255	20/255	25/255
DQN (VANILLA)	**21.0 ± 0.0**	−21.0 ± 0.0	−21.0 ± 0.0	−21.0 ± 0.0
SA-DQN (CONVEX)	**21.0 ± 0.0**	−21.0 ± 0.0	−21.0 ± 0.0	−21.0 ± 0.0
RADIAL-DQN	**21.0 ± 0.0**	−21.0 ± 0.0	−21.0 ± 0.0	−21.0 ± 0.0
AT-DQN	**21.0 ± 0.0**	18.0 ± 2.2	−0.8 ± 4.4	−19.4 ± 0.1
NCL-AT-DQN	**21.0 ± 0.0**	20.4 ± 0.2	−21.0 ± 0.0	−21.0 ± 0.0
NCL-RADIAL-DQN	**21.0 ± 0.0**	−20.6 ± 0.1	−21.0 ± 0.0	−21.0 ± 0.0
BCL-C-AT-DQN	**21.0 ± 0.0**	**21.0 ± 0.0**	**21.0 ± 0.0**	**21.0 ± 0.0**
BCL-MOS-AT-DQN	**21.0 ± 0.0**	**21.0 ± 0.0**	20.9 ± 0.0	20.9 ± 0.0
BCL-RADIAL-DQN	**21.0 ± 0.0**	**21.0 ± 0.0**	−20.9 ± 0.1	−21.0 ± 0.0

FREEWAY

METHOD/METRIC	NOMINAL	30-STEP PGD/RI-FGSM ATTACK		
ϵ	0	10/255	15/255	20/255
DQN (VANILLA)	**33.9 ± 0.1**	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
SA-DQN (CONVEX)	30.0 ± 0.0	19.3 ± 0.4	19.3 ± 0.3	20.0 ± 0.3
RADIAL-DQN	33.2 ± 0.2	17.1 ± 0.3	13.4 ± 0.2	7.9 ± 0.3
AT-DQN	32.4 ± 0.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
NCL-AT-DQN	32.8 ± 0.2	22.0 ± 0.5	9.6 ± 0.4	0.0 ± 0.0
NCL-RADIAL-DQN	33.5 ± 0.2	9.7 ± 0.5	11.6 ± 0.5	18.0 ± 0.4
BCL-C-AT-DQN	**34.0 ± 0.0**	28.8 ± 0.4	21.6 ± 0.5	17.4 ± 0.2
BCL-MOS-AT-DQN	**34.0 ± 0.0**	31.1 ± 0.3	**25.9 ± 0.4**	**20.8 ± 0.3**
BCL-RADIAL-DQN	33.1 ± 0.1	**33.4 ± 0.1**	**25.9 ± 0.6**	**21.2 ± 0.5**

BANKHEIST

METHOD/METRIC	NOMINAL	30-STEP PGD/RI-FGSM ATTACK		
ϵ	0	5/255	10/255	15/255
DQN (VANILLA)	1325.5 ± 5.7	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
SA-DQN (CONVEX)	1237.5 ± 1.7	1126.0 ± 32.0	63.0 ± 3.5	16.0 ± 1.6
RADIAL-DQN	**1349.5 ± 1.7**	581.5 ± 16.7	0.0 ± 0.0	0.0 ± 0.0
AT-DQN	1271.0 ± 15.5	129.0 ± 10.2	5.5 ± 1.1	0.0 ± 0.0
NCL-AT-DQN	1311.0 ± 4.0	245.0 ± 23.7	1.0 ± 0.7	0.0 ± 0.0
NCL-RADIAL-DQN	1272.0 ± 10.7	1168.0 ± 3.4	59.5 ± 7.6	9.0 ± 1.9
BCL-C-AT-DQN	1285.5 ± 5.2	1143.5 ± 30.0	988.5 ± 12.3	250.5 ± 14.6
BCL-MOS-AT-DQN	1307.5 ± 9.5	1095.5 ± 6.2	664.0 ± 60.6	586.5 ± 105.6
BCL-RADIAL-DQN	1225.5 ± 4.9	**1225.5 ± 4.9**	**1223.5 ± 4.1**	228.5 ± 13.9
BCL-RADIAL+AT-DQN	1215.0 ± 8.4	1093.0 ± 5.3	1010.5 ± 8.0	**961.5 ± 9.2**

ROADRUNNER

METHOD/METRIC	NOMINAL	30-STEP PGD/RI-FGSM ATTACK		
ϵ	0	5/255	10/255	15/255
DQN (VANILLA)	43390 ± 973	0 ± 0	0 ± 0	0 ± 0
SA-DQN (CONVEX)	**45870 ± 1380**	985 ± 207	0 ± 0	0 ± 0
RADIAL-DQN	44595 ± 1165	7195 ± 929	495 ± 116	0 ± 0
AT-DQN	39890 ± 2092	20160 ± 1973	0 ± 0	0 ± 0
NCL-AT-DQN	**47925 ± 1123**	37745 ± 2014	10 ± 10	0 ± 0
NCL-RADIAL-DQN	41045 ± 1289	37865 ± 1082	**37865 ± 1082**	6350 ± 590
BCL-C-AT-DQN	**45815 ± 1422**	31305 ± 3590	11405 ± 1385	6335 ± 716
BCL-MOS-AT-DQN	44275 ± 1997	**40060 ± 1828**	15785 ± 1124	1195 ± 180
BCL-RADIAL-DQN	41045 ± 1289	37865 ± 1082	**37865 ± 1082**	6350 ± 590
BCL-RADIAL+AT-DQN	42490 ± 1309	**42490 ± 1309**	**37665 ± 1563**	**25325 ± 1057**

DQN training for Pong and Freeway. For Pong, this appears to be caused by obfuscated gradients, as we find that the model produced by BCL-RADIAL-DQN have gradients that are nearly zero almost everywhere. For Freeway, since the BCL-RADIAL-DQN result is comparable to BCL-MOS-AT-DQN, BCL-RADIAL-DQN might have already reached the robustness boundary of the AT curriculum training method.

For BankHeist and RoadRunner, BCL-RADIAL+AT-DQN models yield the most significant results. The results demonstrate that performing BCL-C-AT-DQN training on top of BCL-RADIAL-DQN could further improve the model robustness. This again demonstrates the value of our BCL framework as well as our AT curriculum learning method. Note that for RoadRunner, we find the curriculum training for BCL-RADIAL-DQN is relatively stable, and two of the BCL-RADIAL-DQN results are achieved by only performing one run for each curriculum phase. Consequently, we include those two runs as the NCL-RADIAL-DQN results.

The AT-DQN and NCL-AT/RADIAL-DQN benchmarks demonstrate the considerable value of the proposed BCL framework, which generalizes both methods. In particular, both AT-DQN and NCL-AT-DQN are not competitive for moderate or high values of ϵ, neither with our proposed methods, nor (in most cases) with the other state of the art robust benchmarks. The NCL-RADIAL-DQN benchmark is more competitive in a few cases (e.g., RoadRunner), but is still significantly outperformed by the best BCL variant. Note that AT-DQN only performs one training phase with a single fixed ϵ, and we evaluated versions of AT-DQN for many different values of ϵ values. The results in Table 2 for AT-DQN correspond to the best-performing result among these. In general, as shown in Appendix D, the performance of both AT-DQN and NCL-AT/RADIAL-DQN is relatively unstable. With AT-DQN in particular, smaller values of ϵ used in training generally yield poor robustness to stronger attacks, while higher values of ϵ lead to greater instability and only slightly higher robustness. In some games, such as Pong, AT-DQN outperforms several other benchmarks (e.g., both SA-DQN and RADIAL-DQN), but it is ineffective in others, such as Freeway. Similarly, NCL-AT-DQN is also unreliable, working relatively well in some settings (e.g., Freeway), but much worse in others, such as BankHeist.

Our final analysis compares the two proposed methods, BCL-C-AT-DQN and BCL-MOS-AT-DQN. The key advantage of BCL-MOS-AT-DQN over BCL-C-AT-DQN is that it potentially significantly reduces training time (in terms of the number of training phases). And, indeed, it does, as shown in Table 3: total curriculum training time for all the experiments is reduced by over 50%, and in the Pong environment, the reduction is over 70%.

Note that we set the thresholds $\overline{V}(\epsilon)$ quite conservatively, and such thresholds only allow us to skip 1-2 phases each time as observed in the experiments. We can further reduce training time by lowering it, albeit by sacrificing efficacy. Additionally, we chose an identical threshold for the rewards across all ϵ_i, which typically means that for higher values of ϵ skipping becomes infrequent as attainable reward drops; making the threshold itself adaptive may further reduce training time.

Surprisingly, however, in addition to the reduction in training time, BCL-MOS-AT-DQN also typically outperforms BCL-C-AT-DQN even in efficacy, both in terms of nominal reward and robustness. This could be a consequence of opportunistic skipping serving as a form of regularization during training, avoiding overfitting to particular lower-magnitude perturbations.

Table 3. Averaged number of phases of curriculum learning: comparing BCL-C-AT-DQN and BCL-MOS-AT-DQN.

METHOD/ENV.	PONG	FW	BH	RR
BCL-C-AT-DQN	66	51	60	36
BCL-MOS-AT-DQN	19.3	24.0	41.7	20.3

6. Conclusion

We purposed a flexible *Bootstrapped Opportunistic Adversarial Curriculum Learning (BCL)* framework. The framework allows multiple training runs for each curriculum phase to significantly increase the model stability, as well as opportunistic skipping forward in the curriculum based on custom target reward criteria to improve training efficiency. We experimentally study four concrete instantiations of the BCL framework, varying (a) whether or not we opportunistically skip forward in the curriculum (BCL-C-AT vs. BCL-MOS-AT), and (b) instantiation of the adversarial loss function (BCL-RADIAL vs. BCL-C-AT vs. hybrid BCL-RADIAL+AT). In our experiments, BCL-MOS-AT reduced the training time for all environments by over 50% compared to BCL-C-AT, demonstrating the value of opportunistic skipping. On the other hand, we find that there is no consistent advantage of one adversarial loss function over the other: in some settings, such as Pong, generating actual adversarial examples leads to far better results, while in others, such as RoadRunner, a combination of both loss functions yields the best performance. Nevertheless, in all cases the best variant of the proposed BCL framework significantly outperforms baselines.

Acknowledgments

This work was partially supported by the NSF (grants IIS-1905558, IIS-1939677, IIS-1903207, and ECCS-2020289), ARO (grants W911NF1910241 and W911NF1810208), NVIDIA, and Amazon.

References

Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

Athalye, A., Engstrom, L., Ilyas, A., and Kwok, K. Synthesizing robust adversarial examples. In *ICML*, 2018.

Balaji, Y., Goldstein, T., and Hoffman, J. Instance adaptive adversarial training: Improved accuracy tradeoffs in neural nets. *arXiv preprint arXiv:1910.08051*, 2019.

Behzadan, V. and Munir, A. Whatever does not kill deep reinforcement learning, makes it stronger. *arXiv preprint arXiv:1712.09344*, 2017.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Bellemare, M. G., Candido, S., Castro, P. S., Gong, J., Machado, M. C., Moitra, S., Ponda, S. S., and Wang, Z. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 2020.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In *International Conference on Machine Learning*, pp. 41–48, 2009.

Cai, Q.-Z., Liu, C., and Song, D. Curriculum adversarial training. In *International Joint Conference on Artificial Intelligence*, pp. 3740–3747, 2018.

Carlini, N. and Wagner, D. A. Towards evaluating the robustness of neural networks. *IEEE Symposium on Security and Privacy*, pp. 39–57, 2017.

Cobbe, K., Hesse, C., Hilton, J., and Schulman, J. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pp. 2048–2056. PMLR, 2020.

Cohen, J. M., Rosenfeld, E., and Kolter, J. Z. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, 2019.

Everett, M., Lütjens, B., and How, J. P. Certifiable robustness to adversarial state uncertainty in deep reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

Eykholt, K., Evtimov, I., Fernandes, E., Li, B., Rahmati, A., Xiao, C., Prakash, A., Kohno, T., and Song, D. X. Robust physical-world attacks on deep learning visual classification. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1625–1634, 2018.

Fortunato, M., Azar, M. G., Piot, B., Menick, J., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O., et al. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295*, 2017.

Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

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

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

Lin, Y.-C., Hong, Z.-W., Liao, Y.-H., Shih, M.-L., Liu, M.-Y., and Sun, M. Tactics of adversarial attack on deep reinforcement learning agents. In *International Joint Conference on Artificial Intelligence*, 2017.

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

Narvekar, S., Sinapov, J., and Stone, P. Autonomous task sequencing for customized curriculum design in reinforcement learning. In *International Joint Conference on Artificial Intelligence*, pp. 2536–2542, 2017.

Narvekar, S., Peng, B., Leonetti, M., Sinapov, J., Taylor, M. E., and Stone, P. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020.

Oikarinen, T., Zhang, W., Megretski, A., Daniel, L., and Weng, T.-W. Robust deep reinforcement learning through adversarial loss. In *Advances in Neural Information Processing Systems*, 2021.

Pattanaik, A., Tang, Z., Liu, S., Bommannan, G., and Chowdhary, G. Robust deep reinforcement learning with adversarial attacks. In *17th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2018*, pp. 2040–2042. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2018.

Raghunathan, A., Steinhardt, J., and Liang, P. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018.

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

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

Sitawarin, C., Chakraborty, S., and Wagner, D. Sat: Improving adversarial training via curriculum-based loss smoothing. In *ACM Workshop on Artificial Intelligence and Security*, pp. 25–36, 2021.

Sun, J., Zhang, T., Xie, X., Ma, L., Zheng, Y., Chen, K., and Liu, Y. Stealthy and efficient adversarial attacks against deep reinforcement learning. In *AAAI Conference on Artificial Intelligence*, volume 34, pp. 5883–5891, 2020.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ international conference on intelligent robots and systems*, pp. 23–30, 2017.

Tong, L., Li, B., Hajaj, C., Xiao, C., Zhang, N., and Vorobeychik, Y. Improving robustness of {ML} classifiers against realizable evasion attacks using conserved features. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pp. 285–302, 2019.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double Q-learning. In *AAAI Conference on Artificial Intelligence*, 2016.

Vorobeychik, Y. and Kantarcioglu, M. *Adversarial machine learning*. Morgan & Claypool Publishers, 2018.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pp. 1995–2003, 2016.

Weng, L., Zhang, H., Chen, H., Song, Z., Hsieh, C.-J., Daniel, L., Boning, D., and Dhillon, I. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pp. 5276–5285. PMLR, 2018.

Wong, E., Schmidt, F., Metzen, J. H., and Kolter, J. Z. Scaling provable adversarial defenses. In *Neural Information Processing Systems*, 2018.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020.

Wu, F., Li, L., Huang, Z., Vorobeychik, Y., Zhao, D., and Li, B. Crop: Certifying robust policies for reinforcement learning through functional smoothing. *arXiv preprint arXiv:2106.09292*, 2022.

Wu, T., Tong, L., and Vorobeychik, Y. Defending against physically realizable attacks on image classification. In *International Conference on Learning Representations*, 2020.

Wu, X., Guo, W., Wei, H., and Xing, X. Adversarial policy training against deep reinforcement learning. In *{USENIX} Security Symposium*, 2021.

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

Zhang, H., Chen, H., Boning, D., and Hsieh, C.-J. Robust reinforcement learning on state observations with learned optimal adversary. In *International Conference on Learning Representations*, 2021.

Zhou, S. K., Le, H. N., Luu, K., Nguyen, H. V., and Ayache, N. Deep reinforcement learning in medical imaging: A literature review. *Medical Image Analysis*, 73:102193, 2021.

A. Bootstrapped Opportunistic Adversarial Curriculum Learning (BCL) for PPO

In this section, we extend the application of BCL framework to PPO-style approaches. We evaluate our approaches on two Procgen (Cobbe et al., 2020) environments: FruitBot and Jumper. For FruitBot, both AT-PPO and BCL-MOS-AT-PPO show higher nominal rewards and significant improvements in terms of robustness for up to 20/255, while the current existing vanilla PPO and RADIAL-PPO has small or even negative rewards for $\epsilon \geq 10/255$. For Jumper, BCL-MOS-AT-PPO achieves significant improvements in terms of robustness for up to 40/255, with rewards under adversarial attacks for $\epsilon \geq 10/255$ more than doubled compare to vanilla PPO (the current most robust model). The experiments on Procgen also demonstrate that our models exhibit good generalization, as the evaluation rewards are high under both training and evaluation distributions.

PPO PPO (Schulman et al., 2017) is a policy gradient method for reinforcement learning, with the objective function as

$$\mathcal{L}(\theta) = \mathbb{E}_{(s_t,a_t,r_t)}\left[-\min\left(\frac{\pi\left(a_t|s_t;\theta\right)}{\pi\left(a_t|s_t;\theta_{\text{old}}\right)}A_t, \text{clip}\left(\frac{\pi\left(a_t|s_t;\theta\right)}{\pi\left(a_t|s_t;\theta_{\text{old}}\right)}, 1-\eta, 1+\eta\right)A_t\right)\right]. \tag{5}$$

Here π is the policy, A_t is the advantage function at time t and η is the hyperparameter. PPO modifies the surrogate objective by clipping the policy ratio to constrain the difference between old and new policy, which stabilizes the training and speeds up convergence. We use Equation (5) as the $\mathcal{L}_{\text{standard}}$ in Equation (2).

RADIAL-PPO RADIAL-PPO (Oikarinen et al., 2021) defines the adversarial loss function as

$$\mathcal{L}_{\text{adv}}(\theta,\epsilon) = \mathbb{E}_{(s_t,a_t,r_t)}\left[-\min\left(\frac{\pi^{\epsilon}\left(a_t|s_t,\epsilon;\theta\right)}{\pi\left(a_t|s_t;\theta_{\text{old}}\right)}A_t, \text{clip}\left(\frac{\pi^{\epsilon}\left(a_t|s_t,\epsilon;\theta\right)}{\pi\left(a_t|s_t;\theta_{\text{old}}\right)}, 1-\eta, 1+\eta\right)A_t\right)\right],$$

with π^{ϵ} the lower bound of the policy network if $A_t \geq 0$, and upper bound otherwise.

The goal for $\mathcal{L}_{\text{adv}}$ in RADIAL-PPO is to form a strict upper bound of the loss function under adversarial perturbations, that is, $\mathcal{L}_{\text{standard}}(s+\delta;\epsilon) \leq \mathcal{L}_{\text{adv}}(s;\epsilon)$ with $||\delta||_p \leq \epsilon$. Robustness is achieved through constraining the strict upper bound of the loss function. This is referred to as approach # 1 in Oikarinen et al. (2021), and was used for RADIAL-PPO training.

Generating Adversarial Perturbations In our model, we replace the upper and lower bounds $\pi^{\epsilon}(a|s_t)$ in RADIAL-PPO with $\tilde{\pi}_1^{\epsilon}(a+\delta^*|s_t)$ and $\tilde{\pi}_2^{\epsilon}(a+\delta^*|s_t)$. For RI-FGSM, δ^* (approximately) solves the following optimization problem:

$$\min_{||\delta||_\infty \leq \epsilon} \text{Softmax}(\text{Logits}(s+\delta)) \odot \text{Logits}(s+\delta), \tag{6}$$

where $\text{Logits}(s)$ is the output vector of the PPO-style neural network and is used to calculate the categorical distribution $\pi(s)$. Note that $\text{Logits}(s)$ has the same size as the action space. For PGD attacks, it is to (approximately) maximize $\mathcal{L}(\text{Softmax}(\text{Logits}(s+\delta^*;\theta)), \pi(s))$ with respect to δ^*, where $\mathcal{L}$ is the cross-entropy loss and δ^* is updated iteratively over a fixed number of iterations (same as in the PGD attack for DQN models).

Since π is a categorical distribution over possible (discrete) actions, we calculate $\tilde{\pi}_1^{\epsilon}(a+\delta^*|s_t)$ by having the a-th logit as the one under adversarial perturbation, and the rest are vanilla logits; $\tilde{\pi}_2^{\epsilon}(a+\delta^*|s_t)$ is calculated by having the a-th logit as the vanilla logit, and the rest are logits under adversarial perturbation. The adversarial loss function is defined as the maximum loss under those two policies:

$$\mathcal{L}_{\text{adv}}(\theta,\epsilon) = \mathbb{E}_{(s_t,a_t,r_t)}\left[-\min_{i\in\{1,2\}}\min\left(\frac{\tilde{\pi}_i^{\epsilon}\left(a_t|s_t,\epsilon;\theta\right)}{\pi\left(a_t|s_t;\theta_{\text{old}}\right)}A_t, \text{clip}\left(\frac{\tilde{\pi}_i^{\epsilon}\left(a_t|s_t,\epsilon;\theta\right)}{\pi\left(a_t|s_t;\theta_{\text{old}}\right)}, 1-\eta, 1+\eta\right)A_t,\right)\right]. \tag{7}$$

Note that instead of forming a strict upper bound of the adversarial loss function as in RADIAL-PPO, $\mathcal{L}_{\text{adv}}$ in Equation (7) provides a lower bound of the adversarial loss function with heuristic adversarial examples.

Experiment Setup We evaluate the purposed approach using two Procgen environments (Cobbe et al., 2020) with discrete action space: FruitBot and Jumper. Note that we did not experiment on the CoinRun environment as in Oikarinen et al. (2021). We find that for CoinRun environment there is an optimal action: we could achieve a reward comparable to RADIAL-PPO regardless of the magnitude of ϵ. For each model, we calculate an efficacy score $\sum_{\text{dist}\in\{\text{Train, Eval}\}}\left(R^{\text{dist}}_{\text{nominal}} + \frac{1}{3}\sum_{\epsilon} R^{\text{dist},\epsilon}_{\text{PGD}}\right)$

for all ϵ listed in Table 7, where $R^{\text{dist}}_{\text{nominal}}$ is the nominal reward and $R^{\text{dist},\epsilon}_{\text{PGD}}$ is the reward under 30-step PGD attack with adversarial perturbation size $||\delta||_\infty \leq \epsilon$ under Train/Eval distribution. Similar to DQN experiments, we conduct three independent runs for each experiment based on the efficacy score and present the median result in the main table (Table 7). For AT-PPO, we conduct experiments for all the three ϵ listed in Table 7, select the median run for each AT-PPO-ϵ, and present the best result as the strongest benchmark.

We find that the adversarial training with PPO-style approaches is relatively stable, and that $K = 1$ suffices. Thus, we use PPO (Vanilla), RADIAL-PPO (Oikarinen et al., 2021) as well as AT-PPO as benchmarks. For PPO (Vanilla) and RADIAL-PPO we use the results from Oikarinen et al. (2021), and we perform our own AT-PPO training with the method purposed above as a restricted version of BCL. We did not perform curriculum learning with RADIAL as it does not work for Jumper. For FruitBot, we use RI-FGSM to generate adversarial examples. For Jumper, as we find RI-FGSM is not effective, we instead use 10-step PGD to generate adversarial examples. We skipped BCL-C-AT-PPO experiments due to extensive computational costs; however, as we will show in the results section, the opportunistic skipping forward mechanism under the BCL framework makes the training possible for Jumper even with 10-step PGD.

To ensure a fair comparison, we let all methods to have the same computational constraints and evaluation metrics: for all environments we train for 25 million steps on the easy setting for each run. For evaluation, we use 30-step PGD attack with step size 0.1 for all models, which is stronger than the 10-step PGD attack used in Oikarinen et al. (2021). We evaluate all models over 1000 episodes using deterministic policy and report the averaged reward under both training distribution (easy setting) and evaluation distribution (full distribution), which is the same as in Oikarinen et al. (2021).

Hyperparameters The PPO specific hyperparameters as well as κ for AT-PPO and BCL-MOS-AT-PPO are the same as in RADIAL-PPO. In FruitBot we use RI-FGSM to generate adversarial examples, with hyperparameter $\alpha = 95.5$, which is approximately 0.375×255 (note that $\alpha = 0.375$ is used in DQN experiments for RI-FGSM). This is due to in RADIAL-PPO code when the gradients are calculated the state space has a range of $0 \sim 255$, instead of being normalized to $0 \sim 1$ as in RADIAL-DQN. The thresholds for BCL-MOS-AT-PPO are shown in Table 4, where $\overline{V}_{\text{PGD(Train)}}(\epsilon)$ is the threshold for the averaged reward under 30-step PGD attack under training distribution.

Table 4. Thresholds $\overline{V}(\epsilon)$ for BCL-MOS-AT-PPO

CRITERIA/ENV.	FRUITBOT	JUMPER
$\overline{V}_{\text{PGD(TRAIN)}}(\epsilon)$	25.0	6.0

We find that although RADIAL-PPO increases the robustness for lower ϵ compared to vanilla PPO (e.g., $\epsilon = 5/255$ under 10-step PGD attack as shown in Oikarinen et al. (2021)), it decreases the robustness for higher ϵ, accompanied by a lower nominal reward. Thus we perform two sets of BCL-MOS-AT-PPO experiments: 1) BCL-MOS(V)-AT-PPO, where we bootstrap from PPO (Vanilla); and 2) BCL-MOS(R)-AT-PPO, where we bootstrap from RADIAL-PPO.

Our baseline curriculum for PPO has an increment of 1/255, with ϵ_0 for each experiment shown in Table 5. For FruitBot we set target $\epsilon = 20/255$, and for Jumper we set target $\epsilon = 40/255$. For all experiments we set $K = 1$. We stop the curriculum training when the model is robust against the target ϵ, meaning the reward under training distribution is above the threshold in Table 4. Note that for FruitBot, we also stop the training when RI-FGSM attack is ineffective towards the target ϵ, meaning that although 30-step PGD indicates the model is not robust against ϵ_i, however, RI-FGSM returns a near perfect reward (close to nominal reward) when the magnitude of adversarial perturbation for the attack is ϵ. In this case, since RI-FGSM is not generating any meaningful adversarial examples, continue training will in fact decrease the model robustness.

Table 5. BCL-MOS-AT-PPO specific hyperparameters (ϵ_0).

CRITERIA/ENV.	FRUITBOT	JUMPER
BCL-MOS(V)-AT-PPO	0/255	1/255
BCL-MOS(R)-AT-PPO	6/255	9/255

Walltime Each adversarial training run takes 10 hours for FruitBot (with RI-FGSM), and 34 hours for Jumper (with 10-step PGD) on a single GeForce RTX 2080Ti GPU. The number of runs conducted for each experiment is shown in Table 6.

Table 6. Number of runs conducted for each experiment.

FRUITBOT	RUN1	RUN2	RUN3
AT-PPO-ϵ	1	1	1
BCL-MOS(V)-AT-PPO	3	2	2
BCL-MOS(R)-AT-PPO	2	2	2
JUMPER	**RUN1**	**RUN2**	**RUN3**
AT-PPO-ϵ	1	1	1
BCL-MOS(V)-AT-PPO	4	4	4
BCL-MOS(R)-AT-PPO	4	4	4

Results Our main results are shown in Table 7, with detailed results deferred to Table 8 and Table 9. The results show that the models trained with our BCL-MOS-AT-PPO approach exhibit significant improvements in terms of nominal reward, robustness as well as generalization compared to state of the art PPO (Vanilla) and RADIAL-PPO.

For FruitBot, both AT-PPO and BCL-MOS-AT-PPO achieve significant improvements in terms of robustness for ϵ up to 20/255, while PPO (Vanilla) and RADIAL-PPO has small or even negative rewards for $\epsilon \geq 10/255$. Furthermore, our AT models achieve higher nominal rewards under both training and evaluation distributions. The success of AT-PPO and BCL-MOS-AT-PPO demonstrate the value of our approach for training with adversarial examples.

For Jumper, BCL-MOS(V)-AT-PPO achieves the most robust model, with rewards under both training and evaluation distributions significantly outperform all benchmark models for $\epsilon \geq 10/255$; it also has a higher nominal reward under evaluation distribution, and comparable high nominal reward under training distribution. Furthermore, we find that BCL-MOS(V)-AT-PPO outperforms BCL-MOS(R)-AT-PPO both in terms of nominal rewards and robustness for $\epsilon \geq 20/255$. This is mainly because RADIAL-PPO has a lower nominal reward and is less robust compared to vanilla PPO for $\epsilon \geq 20/255$.

We also find that with BCL-MOS-AT-PPO training, the resulting models can be easily robust against a higher ϵ when trained against a lower one. The final models for BCL-MOS-AT-PPO in the Jumper environment are only trained for 4 curriculum phases to achieve robustness against ϵ = 40/255. This demonstrates the value of the opportunistic skipping forward mechanism in our BCL framework.

B. Additional Experimental Results for PPO models

Below we show the detailed experimental results for all three independent runs for each experiment. We separate the results by environments for better visualization and comparison. For PPO (Vanilla) and RADIAL-PPO we use the released models from Oikarinen et al. (2021). For AT-PPO, we perform our own training as a restricted case of the BCL algorithm.

For AT-PPO, we include all 9 runs for each environment, naming them as AT-PPO-ϵ, where ϵ is the single value of the base curriculum taken as an input by the model. The choices of ϵ are the ones we use for evaluation. For each AT-PPO-ϵ we conduct three independent runs, and use the median of those three runs as the representative result. The result reported in Table 7 for each environment is the best median result among those three ϵ.

Table 7. Average episode rewards ± standard error of the mean (SEM) over 1000 episodes on both training and evaluation set. The gray rows are the most robust models, selected based on efficacy score $\sum_{\text{dist}} \left(R^{\text{dist}}_{\text{nominal}} + \frac{1}{3} \sum_{\epsilon} R^{\text{dist},\epsilon}_{\text{PGD}} \right)$. **Boldface** marks the best results for each value of ϵ; we marked multiple row entries as boldface for a given ϵ if they are statistically indistinguishable (i.e., have overlapping confidence intervals).

FRUITBOT					
MODEL	DIST.	NOMINAL	30-STEP PGD ATTACK		
		$\epsilon = 0$	$\epsilon = 10/255$	$\epsilon = 15/255$	$\epsilon = 20/255$
PPO (VANILLA)	TRAIN	30.20 ± 0.23	2.40 ± 0.21	0.73 ± 0.16	-0.72 ± 0.14
	EVAL	26.09 ± 0.33	1.70 ± 0.20	0.11 ± 0.14	-0.50 ± 0.13
RADIAL-PPO	TRAIN	28.03 ± 0.24	-0.90 ± 0.13	-1.28 ± 0.10	-1.64 ± 0.10
	EVAL	26.08 ± 0.29	-1.24 ± 0.13	-1.53 ± 0.11	-1.81 ± 0.11
AT-PPO	TRAIN	31.14 ± 0.19	28.69 ± 0.29	26.35 ± 0.32	$\mathbf{24.41 \pm 0.35}$
	EVAL	$\mathbf{28.26 \pm 0.29}$	26.47 ± 0.34	24.56 ± 0.36	20.44 ± 0.40
BCL-MOS(V)-AT-PPO	TRAIN	$\mathbf{32.11 \pm 0.17}$	29.98 ± 0.24	27.40 ± 0.31	$\mathbf{24.23 \pm 0.36}$
	EVAL	$\mathbf{28.81 \pm 0.28}$	$\mathbf{27.61 \pm 0.31}$	$\mathbf{25.52 \pm 0.35}$	$\mathbf{21.63 \pm 0.39}$
BCL-MOS(R)-AT-PPO	TRAIN	31.40 ± 0.20	$\mathbf{30.80 \pm 0.21}$	$\mathbf{28.22 \pm 0.30}$	20.18 ± 0.40
	EVAL	26.95 ± 0.34	26.28 ± 0.35	24.17 ± 0.37	17.87 ± 0.41

JUMPER					
MODEL	DIST.	NOMINAL	30-STEP PGD ATTACK		
		$\epsilon = 0$	$\epsilon = 10/255$	$\epsilon = 20/255$	$\epsilon = 40/255$
PPO (VANILLA)	TRAIN	$\mathbf{8.69 \pm 0.11}$	3.42 ± 0.15	3.61 ± 0.15	2.94 ± 0.14
	EVAL	4.22 ± 0.16	2.81 ± 0.14	2.62 ± 0.14	2.50 ± 0.14
RADIAL-PPO	TRAIN	6.59 ± 0.15	5.43 ± 0.16	2.45 ± 0.14	1.44 ± 0.11
	EVAL	3.85 ± 0.15	3.03 ± 0.14	2.04 ± 0.13	1.44 ± 0.11
AT-PPO	TRAIN	7.57 ± 0.14	4.98 ± 0.16	4.35 ± 0.16	3.52 ± 0.15
	EVAL	$\mathbf{4.55 \pm 0.16}$	3.81 ± 0.15	3.35 ± 0.15	2.51 ± 0.14
BCL-MOS(V)-AT-PPO	TRAIN	$\mathbf{8.67 \pm 0.11}$	$\mathbf{8.15 \pm 0.12}$	$\mathbf{8.40 \pm 0.12}$	$\mathbf{7.84 \pm 0.13}$
	EVAL	$\mathbf{4.57 \pm 0.16}$	$\mathbf{4.64 \pm 0.16}$	$\mathbf{4.65 \pm 0.16}$	$\mathbf{4.41 \pm 0.16}$
BCL-MOS(R)-AT-PPO	TRAIN	8.09 ± 0.12	$\mathbf{8.29 \pm 0.12}$	$\mathbf{8.40 \pm 0.12}$	6.93 ± 0.15
	EVAL	$\mathbf{4.39 \pm 0.16}$	4.29 ± 0.16	4.09 ± 0.16	3.85 ± 0.15

Table 8. FruitBot environment. Average episode rewards ± standard error of the mean (SEM) over 1000 episodes on both training and evaluation set. The gray rows are the median of three runs, selected based on efficacy score $\sum_{\text{dist}} \left(R^{\text{dist}}_{\text{nominal}} + \frac{1}{3} \sum_{\epsilon} R^{\text{dist},\epsilon}_{\text{PGD}} \right)$.

		FRUITBOT			
MODEL	DIST.	NOMINAL	30-STEP PGD ATTACK		
		ϵ = 0	ϵ = 10/255	ϵ= 15/255	ϵ= 20/255
PPO (VANILLA)	TRAIN	30.20 ± 0.23	2.40 ± 0.21	0.73 ± 0.16	-0.72 ± 0.14
	EVAL	26.09 ± 0.33	1.70 ± 0.20	0.11 ± 0.14	-0.50 ± 0.13
RADIAL-PPO	TRAIN	28.03 ± 0.24	-0.90 ± 0.13	-1.28 ± 0.10	-1.64 ± 0.10
	EVAL	26.08 ± 0.29	-1.24 ± 0.13	-1.53 ± 0.11	-1.81 ± 0.11
AT-PPO-10/255 (RUN1)	TRAIN	24.69 ± 0.46	23.46 ± 0.46	22.38 ± 0.45	17.73 ± 0.47
	EVAL	24.38 ± 0.43	23.69 ± 0.42	21.66 ± 0.45	17.26 ± 0.46
AT-PPO-10/255 (RUN2)	TRAIN	30.27 ± 0.23	28.73 ± 0.29	27.03 ± 0.31	22.97 ± 0.36
	EVAL	28.15 ± 0.28	27.30 ± 0.30	25.17 ± 0.34	20.42 ± 0.40
AT-PPO-10/255 (RUN3)	TRAIN	31.20 ± 0.19	31.08 ± 0.17	29.13 ± 0.23	23.83 ± 0.36
	EVAL	28.93 ± 0.27	27.72 ± 0.30	25.99 ± 0.33	21.91 ± 0.39
AT-PPO-15/255 (RUN1)	TRAIN	29.62 ± 0.27	28.92 ± 0.29	26.82 ± 0.32	22.47 ± 0.38
	EVAL	26.48 ± 0.34	25.71 ± 0.35	24.10 ± 0.37	22.00 ± 0.38
AT-PPO-15/255 (RUN2)	TRAIN	31.48 ± 0.18	29.49 ± 0.26	28.82 ± 0.27	24.72 ± 0.35
	EVAL	28.48 ± 0.28	27.30 ± 0.30	25.97 ± 0.32	22.82 ± 0.37
AT-PPO-15/255 (RUN3)	TRAIN	31.04 ± 0.23	28.75 ± 0.29	26.76 ± 0.34	20.92 ± 0.42
	EVAL	28.07 ± 0.31	26.41 ± 0.34	24.09 ± 0.38	20.33 ± 0.41
AT-PPO-20/255 (RUN1)	TRAIN	30.65 ± 0.23	27.83 ± 0.32	26.16 ± 0.34	21.04 ± 0.39
	EVAL	27.44 ± 0.32	25.64 ± 0.35	22.67 ± 0.39	20.57 ± 0.40
AT-PPO-20/255 (RUN2)	TRAIN	31.14 ± 0.19	28.69 ± 0.29	26.35 ± 0.32	24.41 ± 0.35
	EVAL	28.26 ± 0.29	26.47 ± 0.34	24.56 ± 0.36	20.44 ± 0.40
AT-PPO-20/255 (RUN3)	TRAIN	28.62 ± 0.31	29.83 ± 0.24	28.12 ± 0.27	25.96 ± 0.32
	EVAL	27.90 ± 0.30	27.20 ± 0.31	25.62 ± 0.34	23.55 ± 0.35
BCL-MOS(V)-AT-PPO (RUN1)	TRAIN	31.32 ± 0.21	30.34 ± 0.23	28.64 ± 0.30	26.24 ± 0.35
	EVAL	28.94 ± 0.27	27.57 ± 0.30	26.34 ± 0.32	23.55 ± 0.36
BCL-MOS(V)-AT-PPO (RUN2)	TRAIN	31.37 ± 0.22	29.79 ± 0.26	26.64 ± 0.33	24.38 ± 0.35
	EVAL	28.06 ± 0.31	26.51 ± 0.33	24.41 ± 0.36	22.12 ± 0.39
BCL-MOS(V)-AT-PPO (RUN3)	TRAIN	32.11 ± 0.17	29.98 ± 0.24	27.40 ± 0.31	24.23 ± 0.36
	EVAL	28.81 ± 0.28	27.61 ± 0.31	25.52 ± 0.35	21.63 ± 0.39
BCL-MOS(R)-AT-PPO (RUN1)	TRAIN	31.13 ± 0.22	28.91 ± 0.26	26.56 ± 0.32	22.37 ± 0.37
	EVAL	26.62 ± 0.34	24.67 ± 0.37	21.76 ± 0.39	19.54 ± 0.41
BCL-MOS(R)-AT-PPO (RUN2)	TRAIN	30.85 ± 0.21	29.77 ± 0.25	26.93 ± 0.34	23.03 ± 0.37
	EVAL	27.30 ± 0.32	25.98 ± 0.34	24.54 ± 0.35	21.14 ± 0.39
BCL-MOS(R)-AT-PPO (RUN3)	TRAIN	31.40 ± 0.20	30.80 ± 0.21	28.22 ± 0.30	20.18 ± 0.40
	EVAL	26.95 ± 0.34	26.28 ± 0.35	24.17 ± 0.37	17.87 ± 0.41

Table 9. Jumper environment. Average episode rewards ± standard error of the mean (SEM) over 1000 episodes on both training and evaluation set. The gray rows are the median of three runs, selected based on efficacy score $\sum_{\text{dist}} \left(R^{\text{dist}}_{\text{nominal}} + \frac{1}{3} \sum_{\epsilon} R^{\text{dist},\epsilon}_{\text{PGD}} \right)$.

		JUMPER			
MODEL	DIST.	NOMINAL	30-STEP PGD ATTACK		
		ϵ = 0	ϵ = 10/255	ϵ= 20/255	ϵ= 40/255
PPO (VANILLA)	TRAIN	8.69 ± 0.11	3.42 ± 0.15	3.61 ± 0.15	2.94 ± 0.14
	EVAL	4.22 ± 0.16	2.81 ± 0.14	2.62 ± 0.14	2.50 ± 0.14
RADIAL-PPO	TRAIN	6.59 ± 0.15	5.43 ± 0.16	2.45 ± 0.14	1.44 ± 0.11
	EVAL	3.85 ± 0.15	3.03 ± 0.14	2.04 ± 0.13	1.44 ± 0.11
AT-PPO-10/255 (RUN1)	TRAIN	6.58 ± 0.15	6.96 ± 0.15	5.54 ± 0.16	0.97 ± 0.09
	EVAL	4.39 ± 0.16	4.24 ± 0.16	3.30 ± 0.15	0.43 ± 0.06
AT-PPO-10/255 (RUN2)	TRAIN	7.41 ± 0.14	6.70 ± 0.15	5.04 ± 0.16	1.31 ± 0.11
	EVAL	4.50 ± 0.16	4.05 ± 0.16	3.35 ± 0.15	0.68 ± 0.08
AT-PPO-10/255 (RUN3)	TRAIN	7.47 ± 0.14	6.94 ± 0.15	5.29 ± 0.16	1.02 ± 0.10
	EVAL	4.64 ± 0.16	4.31 ± 0.16	3.27 ± 0.15	0.58 ± 0.07
AT-PPO-20/255 (RUN1)	TRAIN	6.90 ± 0.15	6.35 ± 0.15	5.23 ± 0.16	2.41 ± 0.14
	EVAL	4.47 ± 0.16	4.31 ± 0.16	3.92 ± 0.15	1.46 ± 0.11
AT-PPO-20/255 (RUN2)	TRAIN	5.96 ± 0.16	5.98 ± 0.16	5.05 ± 0.16	1.83 ± 0.12
	EVAL	4.56 ± 0.16	4.29 ± 0.16	3.85 ± 0.15	1.78 ± 0.12
AT-PPO-20/255 (RUN3)	TRAIN	6.56 ± 0.15	6.14 ± 0.15	5.06 ± 0.16	1.85 ± 0.12
	EVAL	4.42 ± 0.16	4.34 ± 0.16	4.00 ± 0.15	1.27 ± 0.11
AT-PPO-40/255 (RUN1)	TRAIN	7.57 ± 0.14	4.98 ± 0.16	4.35 ± 0.16	3.52 ± 0.15
	EVAL	4.55 ± 0.16	3.81 ± 0.15	3.35 ± 0.15	2.51 ± 0.14
AT-PPO-40/255 (RUN2)	TRAIN	7.43 ± 0.14	4.74 ± 0.16	4.20 ± 0.16	3.98 ± 0.15
	EVAL	4.54 ± 0.16	3.88 ± 0.15	3.24 ± 0.15	3.39 ± 0.15
AT-PPO-40/255 (RUN3)	TRAIN	6.72 ± 0.15	4.66 ± 0.16	4.36 ± 0.16	4.01 ± 0.16
	EVAL	4.71 ± 0.16	3.90 ± 0.15	3.19 ± 0.15	2.76 ± 0.14
BCL-MOS(V)-AT-PPO (RUN1)	TRAIN	8.67 ± 0.11	8.15 ± 0.12	8.40 ± 0.12	7.84 ± 0.13
	EVAL	4.57 ± 0.16	4.64 ± 0.16	4.65 ± 0.16	4.41 ± 0.16
BCL-MOS(V)-AT-PPO (RUN2)	TRAIN	9.09 ± 0.09	8.85 ± 0.10	8.50 ± 0.11	7.64 ± 0.13
	EVAL	4.77 ± 0.16	4.77 ± 0.16	4.78 ± 0.16	4.43 ± 0.16
BCL-MOS(V)-AT-PPO (RUN3)	TRAIN	8.75 ± 0.10	8.73 ± 0.11	8.64 ± 0.11	5.97 ± 0.16
	EVAL	4.64 ± 0.16	4.63 ± 0.16	4.49 ± 0.16	4.14 ± 0.16
BCL-MOS(R)-AT-PPO (RUN1)	TRAIN	8.09 ± 0.12	8.29 ± 0.12	8.40 ± 0.12	6.93 ± 0.15
	EVAL	4.39 ± 0.16	4.29 ± 0.16	4.09 ± 0.16	3.85 ± 0.15
BCL-MOS(R)-AT-PPO (RUN2)	TRAIN	8.27 ± 0.12	7.27 ± 0.14	6.99 ± 0.15	6.11 ± 0.15
	EVAL	4.53 ± 0.16	4.33 ± 0.16	4.25 ± 0.16	3.91 ± 0.15
BCL-MOS(R)-AT-PPO (RUN3)	TRAIN	8.16 ± 0.12	8.1 ± 0.12	8.35 ± 0.12	7.36 ± 0.14
	EVAL	4.58 ± 0.16	4.3 ± 0.16	4.29 ± 0.16	4.15 ± 0.16

C. Hyperparameters for DQN models

Table 10. Thresholds $\overline{V}(\epsilon)$ for NCL/BCL-RADIAL-DQN

CRITERIA/ENV.	PONG	FW	BH	RR
$\overline{V}_{\text{NOMINAL}}(\epsilon)$	20	32	1200	35000

Table 11. BCL-AT specific hyperparameters

METHOD/ENV.	PONG			FREEWAY		
	K	K_{MIN}	ϵ	K	K_{MIN}	ϵ
BCL-C-AT-DQN	3	3	$\frac{30}{255}$	3	3	$\frac{20}{255}$
BCL-MOS-AT-DQN	3	1	$\frac{30}{255}$	3	1	$\frac{20}{255}$
METHOD/ENV.	BANKHEIST			ROADRUNNER		
	K	K_{MIN}	ϵ	K	K_{MIN}	ϵ
BCL-C-AT-DQN	5	5	$\frac{15}{255}$	3	3	$\frac{15}{255}$
BCL-MOS-AT-DQN	5	1	$\frac{15}{255}$	3	1	$\frac{15}{255}$
BCL-RADIAL+AT-DQN	3	3	$\frac{15}{255}$	1	1	$\frac{15}{255}$

Table 12. Thresholds $\overline{V}(\epsilon)$ for BCL-MOS-AT-DQN

CRITERIA/ENV.	PONG	FW	BH	RR
$\overline{V}_{\text{NOMINAL}}(\epsilon)$	20	30	1200	40000
$\overline{V}_{\text{ADV}}(\epsilon)$	20	25	1000	12000

D. Additional Experimental Results for DQN models

Below we show the detailed experimental results for all three independent runs for each experiment. We separate the results by environments for better visualization and comparison. That is, we have four tables, each for Pong (Table 13), Freeway (Table 17), BankHeist (Table 21) and RoadRunner (Table 25). The discount factor used for evaluation is 1, with maximum episode length 10000, which is the same as in Oikarinen et al. (2021). For benchmark models DQN (Vanilla) and SA-DQN (Convex) we use the released models from Zhang et al. (2020), and for RADIAL-DQN we use the released models from Oikarinen et al. (2021). For benchmark models AT-DQN and NCL-AT-DQN, we perform our own training as two restricted cases of the BCL algorithm.

For AT-DQN we include all 9 runs for each environment, naming them as AT-DQN-ϵ, where ϵ is the single value of the base curriculum taken as an input by the model. The choices of ϵ are the ones we use for evaluation. For each AT-DQN-ϵ we conduct three independent runs, and use the median of those three runs as the representative result. The result reported in Table 2 for each environment is the best median result among those three ϵ.

For NCL-AT-DQN and NCL-RADIAL-DQN we present the best result along the curriculum path for each run. For better comparison between different approaches, we set the target robustness level ϵ in curriculum for NCL-AT-DQN the same as in BCL experiments, that is, $\epsilon = 25/255$ for Pong, $\epsilon = 20/255$ for Freeway, $\epsilon = 15/255$ for BankHeist and $\epsilon = 15/255$ for RoadRunner.

Table 13. Pong environment. Average episode rewards $\pm$ standard error of the mean (SEM) over 20 episodes. The gray rows are the median of three runs (selected based on efficacy score $R_{\text{nominal}} + \frac{1}{3}\sum_{\epsilon} R^{\epsilon}_{\text{adv}}$). We report the lowest rewards among 30-step PGD, RI-FGSM, RI-FGSM (Multi) and RI-FGSM (Multi-T) attacks.

	PONG			
MODEL/METRIC	NOMINAL	30-STEP PGD/RI-FGSM ATTACK		
ϵ	0	10/255	20/255	25/255
DQN (VANILLA)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
SA-DQN (CONVEX)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
RADIAL-DQN	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
AT-DQN-10/255 (RUN1)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
AT-DQN-10/255 (RUN2)	21.0 ± 0.0	-16.7 ± 2.7	-21.0 ± 0.0	-20.9 ± 0.1
AT-DQN-10/255 (RUN3)	20.8 ± 0.1	-17.7 ± 1.3	-21.0 ± 0.0	-21.0 ± 0.0
AT-DQN-20/255 (RUN1)	21.0 ± 0.0	7.1 ± 3.0	-17.9 ± 2.1	-21.0 ± 0.0
AT-DQN-20/255 (RUN2)	20.8 ± 0.1	18.3 ± 0.4	-20.9 ± 0.1	-21.0 ± 0.0
AT-DQN-20/255 (RUN3)	21.0 ± 0.0	-13.6 ± 2.6	-20.5 ± 0.1	-21.0 ± 0.0
AT-DQN-25/255 (RUN1)	20.8 ± 0.1	9.9 ± 3.7	-20.8 ± 0.1	-21.0 ± 0.0
AT-DQN-25/255 (RUN2)	20.1 ± 0.3	-16.9 ± 2.5	-20.5 ± 0.2	-21.0 ± 0.0
AT-DQN-25/255 (RUN3)	21.0 ± 0.0	18.0 ± 2.2	-0.8 ± 4.4	-19.4 ± 0.1
NCL-AT-DQN (RUN1)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
NCL-AT-DQN (RUN2)	21.0 ± 0.0	20.4 ± 0.2	-21.0 ± 0.0	-21.0 ± 0.0
NCL-AT-DQN (RUN3)	21.0 ± 0.0	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
NCL-RADIAL-DQN (RUN1)	21.0 ± 0.0	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
NCL-RADIAL-DQN (RUN2)	21.0 ± 0.0	-20.6 ± 0.1	-21.0 ± 0.0	-21.0 ± 0.0
NCL-RADIAL-DQN (RUN3)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
BCL-C-AT-DQN (RUN1)	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
BCL-C-AT-DQN (RUN2)	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
BCL-C-AT-DQN (RUN3)	21.0 ± 0.0	20.8 ± 0.1	20.5 ± 0.2	20.1 ± 0.3
BCL-MOS-AT-DQN (RUN1)	21.0 ± 0.0	21.0 ± 0.0	20.7 ± 0.2	20.8 ± 0.1
BCL-MOS-AT-DQN (RUN2)	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0	20.9 ± 0.1
BCL-MOS-AT-DQN (RUN3)	21.0 ± 0.0	21.0 ± 0.0	20.9 ± 0.0	20.9 ± 0.0
BCL-RADIAL-DQN (RUN1)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
BCL-RADIAL-DQN (RUN2)	21.0 ± 0.0	21.0 ± 0.0	-20.9 ± 0.1	-21.0 ± 0.0
BCL-RADIAL-DQN (RUN3)	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0	-16.6 ± 1.0

Table 14. Detailed experiment results for Pong environment. The lowest reward under attacks are marked gray.

		PONG		
MODEL	ATTACK	ϵ = 10/255	ϵ= 20/255	ϵ =25/255
DQN (VANILLA)	30-STEP PGD	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
SA-DQN (CONVEX)	30-STEP PGD	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
RADIAL-DQN	30-STEP PGD	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
AT-DQN-10/255 (RUN1)	30-STEP PGD	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
AT-DQN-10/255 (RUN2)	30-STEP PGD	-16.7 ± 2.7	-21.0 ± 0.0	-20.8 ± 0.1
	RI-FGSM	20.1 ± 0.0	-20.8 ± 0.1	-20.8 ± 0.1
	RI-FGSM (MULTI)	19.1 ± 0.4	-20.6 ± 0.1	-20.8 ± 0.1
	RI-FGSM (MULTI-T)	18.1 ± 0.8	-21.0 ± 0.0	-20.9 ± 0.1
AT-DQN-10/255 (RUN3)	30-STEP PGD	-17.7 ± 1.3	-21.0 ± 0.0	-20.9 ± 0.1
	RI-FGSM	20.7 ± 0.1	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI)	15.0 ± 2.0	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI-T)	12.7 ± 2.8	-21.0 ± 0.0	-21.0 ± 0.0
AT-DQN-20/255 (RUN1)	30-STEP PGD	7.1 ± 3.0	-17.9 ± 2.1	-21.0 ± 0.0
	RI-FGSM	20.5 ± 0.2	20.4 ± 0.2	-20.6 ± 0.2
	RI-FGSM (MULTI)	12.9 ± 1.8	2.8 ± 2.7	-20.9 ± 0.0
	RI-FGSM (MULTI-T)	13.0 ± 1.7	-8.4 ± 2.2	-21.0 ± 0.0
AT-DQN-20/255 (RUN2)	30-STEP PGD	18.3 ± 0.4	-20.9 ± 0.1	-21.0 ± 0.0
	RI-FGSM	19.9 ± 0.5	20.6 ± 0.1	-21.0 ± 0.0
	RI-FGSM (MULTI)	20.3 ± 0.2	16.4 ± 0.9	-21.0 ± 0.0
	RI-FGSM (MULTI-T)	20.8 ± 0.1	6.7 ± 3.4	-21.0 ± 0.0
AT-DQN-20/255 (RUN3)	30-STEP PGD	-13.6 ± 2.6	-20.5 ± 0.1	-20.6 ± 0.1
	RI-FGSM	6.0 ± 3.7	17.8 ± 1.9	-21.0 ± 0.0
	RI-FGSM (MULTI)	1.3 ± 3.8	1.5 ± 2.7	-21.0 ± 0.0
	RI-FGSM (MULTI-T)	11.1 ± 2.8	-7.0 ± 3.1	-21.0 ± 0.0
AT-DQN-25/255 (RUN1)	30-STEP PGD	20.8 ± 0.1	-20.8 ± 0.1	-21.0 ± 0.0
	RI-FGSM	20.4 ± 0.3	19.8 ± 0.5	19.6 ± 0.5
	RI-FGSM (MULTI)	18.3 ± 0.9	-9.3 ± 3.0	-18.1 ± 0.6
	RI-FGSM (MULTI-T)	9.9 ± 3.7	-14.8 ± 1.9	-18.9 ± 0.4
AT-DQN-25/255 (RUN2)	30-STEP PGD	-16.9 ± 2.5	-20.5 ± 0.2	-21.0 ± 0.0
	RI-FGSM	18.1 ± 0.8	17.6 ± 0.9	19.0 ± 0.4
	RI-FGSM (MULTI)	8.2 ± 3.8	0.5 ± 3.5	-4.3 ± 3.2
	RI-FGSM (MULTI-T)	5.8 ± 4.0	-3.8 ± 3.6	-15.8 ± 1.5
AT-DQN-25/255 (RUN3)	30-STEP PGD	20.9 ± 0.1	-0.8 ± 4.4	-19.4 ± 0.1
	RI-FGSM	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM (MULTI)	18.0 ± 2.2	19.6 ± 0.8	0.1 ± 3.4
	RI-FGSM (MULTI-T)	18.0 ± 2.2	15.7 ± 2.0	-8.4 ± 2.0

Table 15. Detailed experiment results for Pong environment. The lowest reward under attacks are marked gray.

		PONG		
MODEL	ATTACK	ϵ = 10/255	ϵ= 20/255	ϵ =25/255
NCL-AT-DQN (RUN1)	30-STEP PGD	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
NCL-AT-DQN (RUN2)	30-STEP PGD	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM	20.9 ± 0.0	-20.1 ± 0.2	-20.9 ± 0.0
	RI-FGSM (MULTI)	20.4 ± 0.2	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI-T)	20.9 ± 0.1	-21.0 ± 0.0	-21.0 ± 0.0
NCL-AT-DQN (RUN3)	30-STEP PGD	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI-T)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
BCL-C-AT-DQN (RUN1)	30-STEP PGD	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM (MULTI)	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM (MULTI-T)	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
BCL-C-AT-DQN (RUN2)	30-STEP PGD	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM (MULTI)	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM (MULTI-T)	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
BCL-C-AT-DQN (RUN3)	30-STEP PGD	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	21.0 ± 0.0	20.9 ± 0.1
	RI-FGSM (MULTI)	20.8 ± 0.1	20.5 ± 0.2	20.6 ± 0.1
	RI-FGSM (MULTI-T)	20.8 ± 0.1	20.6 ± 0.2	20.1 ± 0.3
BCL-MOS-AT-DQN (RUN1)	30-STEP PGD	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM (MULTI)	21.0 ± 0.0	20.8 ± 0.1	20.8 ± 0.1
	RI-FGSM (MULTI-T)	21.0 ± 0.0	20.7 ± 0.2	20.8 ± 0.1
BCL-MOS-AT-DQN (RUN2)	30-STEP PGD	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM (MULTI)	21.0 ± 0.0	21.0 ± 0.0	20.9 ± 0.1
	RI-FGSM (MULTI-T)	21.0 ± 0.0	21.0 ± 0.0	20.9 ± 0.1
BCL-MOS-AT-DQN (RUN3)	30-STEP PGD	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM (MULTI)	21.0 ± 0.0	21.0 ± 0.0	20.9 ± 0.0
	RI-FGSM (MULTI-T)	21.0 ± 0.0	20.9 ± 0.0	20.9 ± 0.0

Table 16. Detailed experiment results for Pong environment. The lowest reward under attacks are marked gray.

	PONG			
MODEL	ATTACK	ϵ = 10/255	ϵ= 20/255	ϵ =25/255
NCL-RADIAL-DQN (RUN1)	30-STEP PGD	21.0 ± 0.0	12.9 ± 2.0	12.8 ± 2.0
	RI-FGSM	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI-T)	21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
NCL-RADIAL-DQN (RUN2)	30-STEP PGD	-20.6 ± 0.1	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM	-6.7 ± 1.1	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI)	-17.8 ± 1.0	-21.0 ± 0.0	-21.0 ± 0.0
	RI-FGSM (MULTI-T)	-16.1 ± 1.2	-21.0 ± 0.0	-21.0 ± 0.0
NCL-RADIAL-DQN (RUN3)	30-STEP PGD	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
BCL-RADIAL-DQN (RUN1)	30-STEP PGD	-21.0 ± 0.0	-21.0 ± 0.0	-21.0 ± 0.0
BCL-RADIAL-DQN (RUN2)	30-STEP PGD	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	12.3 ± 1.2	-20.7 ± 0.1
	RI-FGSM (MULTI)	21.0 ± 0.0	-20.9 ± 0.1	-21.0 ± 0.0
	RI-FGSM (MULTI-T)	21.0 ± 0.0	-20.7 ± 0.2	-21.0 ± 0.0
BCL-RADIAL-DQN (RUN3)	30-STEP PGD	21.0 ± 0.0	21.0 ± 0.0	21.0 ± 0.0
	RI-FGSM	21.0 ± 0.0	21.0 ± 0.0	20.6 ± 0.2
	RI-FGSM (MULTI)	21.0 ± 0.0	21.0 ± 0.0	-16.6 ± 1.0
	RI-FGSM (MULTI-T)	21.0 ± 0.0	21.0 ± 0.0	-6.5 ± 2.0

Table 17. Freeway environment. Average episode rewards ± standard error of the mean (SEM) over 20 episodes. The gray rows are the median of three runs (selected based on efficacy score $R_{\text{nominal}} + \frac{1}{3}\sum_{\epsilon} R^{\epsilon}_{\text{adv}}$). We report the lowest rewards among 30-step PGD, RI-FGSM, RI-FGSM (Multi) and RI-FGSM (Multi-T) attacks.

	FREEWAY			
MODEL/METRIC	NOMINAL	30-STEP PGD/RI-FGSM ATTACK		
ϵ	0	10/255	15/255	20/255
DQN (VANILLA)	33.9 ± 0.1	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
SA-DQN (CONVEX)	30.0 ± 0.0	19.3 ± 0.4	19.3 ± 0.3	20.0 ± 0.3
RADIAL-DQN	33.2 ± 0.2	17.1 ± 0.3	13.4 ± 0.2	7.9 ± 0.3
AT-DQN-10/255 (RUN1)	32.4 ± 0.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-10/255 (RUN2)	33.3 ± 0.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-10/255 (RUN3)	32.3 ± 0.2	0.1 ± 0.1	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-15/255 (RUN1)	32.9 ± 0.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-15/255 (RUN2)	30.9 ± 0.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-15/255 (RUN3)	32.0 ± 0.4	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-20/255 (RUN1)	29.1 ± 0.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-20/255 (RUN2)	31.4 ± 0.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-20/255 (RUN3)	32.2 ± 0.4	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
NCL-AT-DQN (RUN1)	32.8 ± 0.2	22.0 ± 0.5	9.6 ± 0.4	0.0 ± 0.0
NCL-AT-DQN (RUN2)	32.7 ± 0.2	26.2 ± 0.2	17.9 ± 0.3	3.9 ± 0.2
NCL-AT-DQN (RUN3)	32.6 ± 0.3	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
NCL-RADIAL-DQN (RUN1)	33.8 ± 0.1	11.9 ± 0.5	0.2 ± 0.1	13.0 ± 0.5
NCL-RADIAL-DQN (RUN2)	33.5 ± 0.1	33.1 ± 0.1	22.5 ± 0.5	21.6 ± 0.4
NCL-RADIAL-DQN (RUN3)	33.5 ± 0.2	9.7 ± 0.5	11.6 ± 0.5	18.0 ± 0.4
BCL-C-AT-DQN (RUN1)	34.0 ± 0.0	31.2 ± 0.4	25.9 ± 0.3	17.3 ± 0.5
BCL-C-AT-DQN (RUN2)	34.0 ± 0.0	28.8 ± 0.4	21.6 ± 0.5	17.4 ± 0.2
BCL-C-AT-DQN (RUN3)	34.0 ± 0.0	26.7 ± 0.3	22.6 ± 0.2	16.1 ± 0.3
BCL-MOS-AT-DQN (RUN1)	33.7 ± 0.1	30.0 ± 0.4	26.6 ± 0.3	21.5 ± 0.4
BCL-MOS-AT-DQN (RUN2)	34.0 ± 0.0	31.1 ± 0.3	25.9 ± 0.4	20.8 ± 0.3
BCL-MOS-AT-DQN (RUN3)	33.8 ± 0.1	29.1 ± 0.5	23.7 ± 0.5	17.6 ± 0.4
BCL-RADIAL-DQN (RUN1)	32.2 ± 0.2	32.2 ± 0.3	21.2 ± 0.4	21.1 ± 0.4
BCL-RADIAL-DQN (RUN2)	33.1 ± 0.1	33.4 ± 0.1	25.9 ± 0.6	21.2 ± 0.5
BCL-RADIAL-DQN (RUN3)	32.7 ± 0.2	32.7 ± 0.2	29.8 ± 0.5	20.7 ± 0.3

Table 18. Detailed experiment results for Freeway environment. The lowest reward under attacks are marked gray.

		FREEWAY		
MODEL	ATTACK	ϵ = 10/255	ϵ= 15/255	ϵ =20/255
DQN (VANILLA)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
SA-DQN (CONVEX)	30-STEP PGD	19.3 ± 0.4	19.3 ± 0.3	20.0 ± 0.3
	RI-FGSM	21.2 ± 0.4	21.5 ± 0.4	21.9 ± 0.3
	RI-FGSM (MULTI)	21.4 ± 0.3	21.3 ± 0.3	20.7 ± 0.5
	RI-FGSM (MULTI-T)	21.4 ± 0.3	21.1 ± 0.3	21.4 ± 0.3
RADIAL-DQN	30-STEP PGD	19.9 ± 0.3	13.4 ± 0.2	7.9 ± 0.3
	RI-FGSM	21.9 ± 0.3	21.8 ± 0.3	21.8 ± 0.3
	RI-FGSM (MULTI)	17.4 ± 0.4	21.7 ± 0.3	21.9 ± 0.3
	RI-FGSM (MULTI-T)	17.1 ± 0.3	21.7 ± 0.3	21.6 ± 0.2
AT-DQN-10/255 (RUN1)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-10/255 (RUN2)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-10/255 (RUN3)	30-STEP PGD	0.1 ± 0.1	0.0 ± 0.0	0.0 ± 0.0
	RI-FGSM	28.9 ± 0.3	23.4 ± 0.4	3.0 ± 0.2
	RI-FGSM (MULTI)	24.4 ± 0.4	1.9 ± 0.2	0.0 ± 0.0
	RI-FGSM (MULTI-T)	23.9 ± 0.3	2.1 ± 0.2	0.0 ± 0.0
AT-DQN-15/255 (RUN1)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-15/255 (RUN2)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-15/255 (RUN3)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-20/255 (RUN1)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-20/255 (RUN2)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-20/255 (RUN3)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0

Table 19. Detailed experiment results for Freeway environment. The lowest reward under attacks are marked gray.

	FREEWAY			
MODEL	ATTACK	ϵ = 10/255	ϵ= 15/255	ϵ =20/255
NCL-AT-DQN (RUN1)	30-STEP PGD	22.8 ± 0.4	17.0 ± 0.4	0.0 ± 0.0
	RI-FGSM	29.7 ± 0.3	29.9 ± 0.4	22.3 ± 0.3
	RI-FGSM (MULTI)	23.6 ± 0.5	11.3 ± 0.4	0.0 ± 0.0
	RI-FGSM (MULTI-T)	22.0 ± 0.5	9.6 ± 0.4	0.0 ± 0.0
NCL-AT-DQN (RUN2)	30-STEP PGD	26.2 ± 0.2	17.9 ± 0.3	3.9 ± 0.2
	RI-FGSM	30.2 ± 0.3	30.4 ± 0.4	30.4 ± 0.4
	RI-FGSM (MULTI)	26.6 ± 0.4	25.6 ± 0.5	12.3 ± 0.3
	RI-FGSM (MULTI-T)	27.5 ± 0.5	25.9 ± 0.3	11.9 ± 0.4
NCL-AT-DQN (RUN3)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
BCL-C-AT-DQN (RUN1)	30-STEP PGD	31.6 ± 0.4	25.9 ± 0.3	17.3 ± 0.5
	RI-FGSM	33.0 ± 0.2	32.8 ± 0.3	33.2 ± 0.2
	RI-FGSM (MULTI)	31.2 ± 0.4	29.6 ± 0.4	23.9 ± 0.4
	RI-FGSM (MULTI-T)	31.5 ± 0.4	29.7 ± 0.4	24.7 ± 0.5
BCL-C-AT-DQN (RUN2)	30-STEP PGD	29.2 ± 0.5	21.6 ± 0.5	17.4 ± 0.2
	RI-FGSM	32.0 ± 0.3	32.4 ± 0.3	33.0 ± 0.2
	RI-FGSM (MULTI)	29.2 ± 0.4	28.0 ± 0.4	25.4 ± 0.4
	RI-FGSM (MULTI-T)	28.8 ± 0.4	28.0 ± 0.5	24.8 ± 0.5
BCL-C-AT-DQN (RUN3)	30-STEP PGD	26.7 ± 0.3	22.6 ± 0.2	16.1 ± 0.3
	RI-FGSM	31.2 ± 0.4	31.5 ± 0.3	32.3 ± 0.3
	RI-FGSM (MULTI)	27.9 ± 0.4	25.9 ± 0.6	24.6 ± 0.3
	RI-FGSM (MULTI-T)	28.4 ± 0.3	25.9 ± 0.3	22.8 ± 0.4
BCL-MOS-AT-DQN (RUN1)	30-STEP PGD	30.0 ± 0.4	26.6 ± 0.3	21.5 ± 0.4
	RI-FGSM	33.0 ± 0.2	33.0 ± 0.2	32.8 ± 0.3
	RI-FGSM (MULTI)	31.2 ± 0.3	30.7 ± 0.4	26.2 ± 0.3
	RI-FGSM (MULTI-T)	31.4 ± 0.3	30.0 ± 0.4	25.4 ± 0.4
BCL-MOS-AT-DQN (RUN2)	30-STEP PGD	31.2 ± 0.4	25.9 ± 0.4	20.8 ± 0.3
	RI-FGSM	32.8 ± 0.3	32.7 ± 0.3	32.5 ± 0.3
	RI-FGSM (MULTI)	31.1 ± 0.3	29.1 ± 0.6	28.4 ± 0.3
	RI-FGSM (MULTI-T)	31.3 ± 0.4	28.3 ± 0.4	27.8 ± 0.3
BCL-MOS-AT-DQN (RUN3)	30-STEP PGD	29.2 ± 0.4	23.7 ± 0.5	17.6 ± 0.4
	RI-FGSM	32.1 ± 0.3	32.3 ± 0.2	32.6 ± 0.2
	RI-FGSM (MULTI)	29.1 ± 0.4	27.0 ± 0.4	26.9 ± 0.5
	RI-FGSM (MULTI-T)	29.1 ± 0.5	27.1 ± 0.4	25.3 ± 0.4

Table 20. Detailed experiment results for Freeway environment. The lowest reward under attacks are marked gray.

	FREEWAY			
MODEL	ATTACK	ϵ = 10/255	ϵ= 15/255	ϵ =20/255
NCL-RADIAL-DQN (RUN1)	30-STEP PGD	25.4 ± 0.4	15.8 ± 0.3	13.0 ± 0.5
	RI-FGSM	25.8 ± 0.5	20.6 ± 0.4	21.8 ± 0.3
	RI-FGSM (MULTI)	11.9 ± 0.5	0.2 ± 0.1	21.5 ± 0.4
	RI-FGSM (MULTI-T)	12.1 ± 0.5	0.3 ± 0.1	21.3 ± 0.4
NCL-RADIAL-DQN (RUN2)	30-STEP PGD	33.1 ± 0.1	23.3 ± 0.5	22.5 ± 0.3
	RI-FGSM	33.5 ± 0.1	22.5 ± 0.5	21.6 ± 0.4
	RI-FGSM (MULTI)	33.1 ± 0.1	23.4 ± 0.5	23.4 ± 0.4
	RI-FGSM (MULTI-T)	33.1 ± 0.1	23.7 ± 0.5	22.4 ± 0.4
NCL-RADIAL-DQN (RUN3)	30-STEP PGD	22.0 ± 0.2	17.8 ± 0.4	18.0 ± 0.4
	RI-FGSM	24.6 ± 0.3	21.7 ± 0.4	21.8 ± 0.3
	RI-FGSM (MULTI)	9.7 ± 0.5	11.6 ± 0.5	21.4 ± 0.2
	RI-FGSM (MULTI-T)	10.0 ± 0.4	11.8 ± 0.5	21.3 ± 0.3
BCL-RADIAL-DQN (RUN1)	30-STEP PGD	32.5 ± 0.3	22.9 ± 0.4	22.6 ± 0.4
	RI-FGSM	32.8 ± 0.3	22.6 ± 0.3	21.7 ± 0.3
	RI-FGSM (MULTI)	32.2 ± 0.3	21.2 ± 0.4	21.1 ± 0.4
	RI-FGSM (MULTI-T)	32.2 ± 0.3	21.3 ± 0.3	21.2 ± 0.3
BCL-RADIAL-DQN (RUN2)	30-STEP PGD	33.4 ± 0.1	30.0 ± 0.2	24.1 ± 0.5
	RI-FGSM	33.4 ± 0.1	29.7 ± 0.3	21.7 ± 0.3
	RI-FGSM (MULTI)	33.4 ± 0.1	26.5 ± 0.5	21.2 ± 0.5
	RI-FGSM (MULTI-T)	33.4 ± 0.1	25.9 ± 0.6	21.6 ± 0.3
BCL-RADIAL-DQN (RUN3)	30-STEP PGD	32.7 ± 0.2	32.0 ± 0.2	22.2 ± 0.5
	RI-FGSM	32.7 ± 0.2	32.2 ± 0.3	20.7 ± 0.3
	RI-FGSM (MULTI)	32.8 ± 0.2	30.8 ± 0.4	21.3 ± 0.2
	RI-FGSM (MULTI-T)	32.9 ± 0.2	29.8 ± 0.5	21.9 ± 0.3

Table 21. BankHeist environment. Average episode rewards ± standard error of the mean (SEM) over 20 episodes. The gray rows are the median of three runs (selected based on efficacy score $R_{\text{nominal}} + \frac{1}{3}\sum_{\epsilon} R^{\epsilon}_{\text{adv}}$). We report the lowest rewards among 30-step PGD, RI-FGSM, RI-FGSM (Multi) and RI-FGSM (Multi-T) attacks.

	BANKHEIST			
MODEL/METRIC	NOMINAL	30-STEP PGD/RI-FGSM ATTACK		
ϵ	0	5/255	10/255	15/255
DQN (VANILLA)	1325.5 ± 5.7	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
SA-DQN (CONVEX)	1237.5 ± 1.7	1126.0 ± 32.0	63.0 ± 3.5	16.0 ± 1.6
RADIAL-DQN	1349.5 ± 1.7	581.5 ± 16.7	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-5/255 (RUN1)	1200.0 ± 12.1	95.5 ± 5.9	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-5/255 (RUN2)	1217.0 ± 10.1	407.5 ± 30.6	4.5 ± 1.1	1.0 ± 0.7
AT-DQN-5/255 (RUN3)	778.5 ± 30.4	129.0 ± 9.0	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-10/255 (RUN1)	1312.5 ± 5.0	132.0 ± 4.1	15.0 ± 2.3	0.0 ± 0.0
AT-DQN-10/255 (RUN2)	1271.0 ± 15.5	129.0 ± 10.2	5.5 ± 1.1	0.0 ± 0.0
AT-DQN-10/255 (RUN3)	1244.5 ± 39.3	49.5 ± 8.0	8.5 ± 1.1	0.0 ± 0.0
AT-DQN-15/255 (RUN1)	1235.0 ± 12.6	27.5 ± 19.0	1.0 ± 0.7	1.0 ± 0.7
AT-DQN-15/255 (RUN2)	1295.5 ± 9.6	12.5 ± 2.1	0.0 ± 0.0	0.0 ± 0.0
AT-DQN-15/255 (RUN3)	1243.5 ± 18.1	30.5 ± 1.8	2.5 ± 1.0	0.0 ± 0.0
NCL-AT-DQN (RUN1)	1311.0 ± 4.0	245.0 ± 23.7	1.0 ± 0.7	0.0 ± 0.0
NCL-AT-DQN (RUN2)	1153.0 ± 38.1	3.0 ± 1.0	0.0 ± 0.0	0.0 ± 0.0
NCL-AT-DQN (RUN3)	1262.0 ± 11.4	740.0 ± 9.7	0.0 ± 0.0	0.0 ± 0.0
NCL-RADIAL-DQN (RUN1)	1285.5 ± 5.2	1265.0 ± 5.6	1243.0 ± 7.5	45.5 ± 2.1
NCL-RADIAL-DQN (RUN2)	1272.0 ± 10.7	1168.0 ± 3.4	59.5 ± 7.6	9.0 ± 1.9
NCL-RADIAL-DQN (RUN3)	1344.5 ± 5.4	1342.0 ± 5.4	198.5 ± 14.1	10.5 ± 1.8
BCL-C-AT-DQN (RUN1)	1295.0 ± 8.9	807.5 ± 82.5	693.0 ± 80.1	248.5 ± 23.7
BCL-C-AT-DQN (RUN2)	1330.5 ± 3.0	1022.0 ± 63.2	956.0 ± 17.9	720.0 ± 12.8
BCL-C-AT-DQN (RUN3)	1285.5 ± 5.2	1143.5 ± 30.0	988.5 ± 12.3	250.5 ± 14.6
BCL-MOS-AT-DQN (RUN1)	1307.5 ± 9.5	1095.5 ± 6.2	664.0 ± 60.6	586.5 ± 105.6
BCL-MOS-AT-DQN (RUN2)	1338.5 ± 3.0	1165.5 ± 9.2	922.5 ± 69.5	470.5 ± 35.6
BCL-MOS-AT-DQN (RUN3)	1281.5 ± 5.8	1184.0 ± 8.6	1003.5 ± 15.3	113.0 ± 4.0
BCL-RADIAL-DQN (RUN1)	1225.5 ± 4.9	1225.5 ± 4.9	1223.5 ± 4.1	228.5 ± 13.9
BCL-RADIAL+AT-DQN (RUN1)	1215.0 ± 8.4	1093.0 ± 5.3	1010.5 ± 8.0	961.5 ± 9.2
BCL-RADIAL-DQN (RUN2)	1261.0 ± 9.6	1251.5 ± 6.0	1200.5 ± 11.8	127.0 ± 10.7
BCL-RADIAL-DQN (RUN3)	1296.5 ± 8.3	1242.5 ± 11.9	1161.0 ± 13.9	8.5 ± 1.6

Table 22. Detailed experiment results for BankHeist environment. The lowest reward under attacks are marked gray.

		BANKHEIST		
MODEL	ATTACK	ϵ = 5/255	ϵ= 10/255	ϵ =15/255
DQN (VANILLA)	30-STEP PGD	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
SA-DQN (CONVEX)	30-STEP PGD	1155.5 ± 6.4	114.0 ± 3.8	24.5 ± 3.1
	RI-FGSM	1153.5 ± 29.5	556.0 ± 25.1	137.5 ± 8.3
	RI-FGSM (MULTI)	1126.0 ± 32.0	63.0 ± 3.5	20.0 ± 1.4
	RI-FGSM (MULTI-T)	1128.0 ± 31.3	75.5 ± 4.4	16.0 ± 1.6
RADIAL-DQN	30-STEP PGD	761.0 ± 24.5	0.0 ± 0.0	0.0 ± 0.0
	RI-FGSM	832.0 ± 33.5	348.0 ± 13.9	139.0 ± 8.8
	RI-FGSM (MULTI)	597.0 ± 20.5	33.0 ± 2.6	12.0 ± 1.7
	RI-FGSM (MULTI-T)	581.5 ± 16.7	40.5 ± 2.2	22.0 ± 2.6
AT-DQN-5/255 (RUN1)	30-STEP PGD	95.5 ± 5.9	0.0 ± 0.0	0.0 ± 0.0
	RI-FGSM	677.0 ± 63.5	497.0 ± 61.8	104.0 ± 11.9
	RI-FGSM (MULTI)	637.5 ± 66.4	344.5 ± 48.5	43.5 ± 4.7
	RI-FGSM (MULTI-T)	812.5 ± 38.8	204.5 ± 26.8	61.0 ± 4.7
AT-DQN-5/255 (RUN2)	30-STEP PGD	407.5 ± 30.6	4.5 ± 1.1	1.0 ± 0.7
	RI-FGSM	941.5 ± 15.6	204.0 ± 17.0	43.0 ± 3.1
	RI-FGSM (MULTI)	890.0 ± 17.9	97.0 ± 10.3	20.0 ± 2.4
	RI-FGSM (MULTI-T)	876.5 ± 22.2	135.0 ± 16.0	32.0 ± 3.0
AT-DQN-5/255 (RUN3)	30-STEP PGD	129.0 ± 9.0	0.0 ± 0.0	0.0 ± 0.0
	RI-FGSM	643.0 ± 65.6	445.0 ± 47.2	48.5 ± 4.9
	RI-FGSM (MULTI)	642.0 ± 65.0	189.0 ± 21.9	19.0 ± 2.1
	RI-FGSM (MULTI-T)	668.5 ± 67.0	212.5 ± 14.2	46.0 ± 4.7
AT-DQN-10/255 (RUN1)	30-STEP PGD	132.0 ± 4.1	15.0 ± 2.3	0.0 ± 0.0
	RI-FGSM	1002.5 ± 16.9	802.5 ± 49.1	525.5 ± 54.3
	RI-FGSM (MULTI)	881.5 ± 16.4	471.0 ± 47.3	181.0 ± 22.0
	RI-FGSM (MULTI-T)	904.5 ± 18.8	511.0 ± 41.9	226.5 ± 20.6
AT-DQN-10/255 (RUN2)	30-STEP PGD	129.0 ± 10.2	5.5 ± 1.1	0.0 ± 0.0
	RI-FGSM	746.0 ± 63.1	713.5 ± 58.3	544.5 ± 48.1
	RI-FGSM (MULTI)	282.0 ± 30.6	535.5 ± 43.7	448.5 ± 36.9
	RI-FGSM (MULTI-T)	277.0 ± 30.7	569.5 ± 49.9	401.5 ± 31.7
AT-DQN-10/255 (RUN3)	30-STEP PGD	49.5 ± 8.0	8.5 ± 1.1	0.0 ± 0.0
	RI-FGSM	831.5 ± 50.9	757.0 ± 33.3	449.5 ± 36.2
	RI-FGSM (MULTI)	758.5 ± 58.0	713.5 ± 45.3	460.0 ± 35.3
	RI-FGSM (MULTI-T)	879.5 ± 27.4	812.5 ± 32.9	456.0 ± 37.4
AT-DQN-15/255 (RUN1)	30-STEP PGD	27.5 ± 19.0	1.0 ± 0.7	1.0 ± 0.7
	RI-FGSM	946.5 ± 15.1	836.0 ± 15.7	771.0 ± 16.7
	RI-FGSM (MULTI)	851.0 ± 17.0	720.0 ± 26.8	658.0 ± 26.2
	RI-FGSM (MULTI-T)	838.5 ± 14.8	654.0 ± 24.2	645.5 ± 37.9
AT-DQN-15/255 (RUN2)	30-STEP PGD	12.5 ± 2.1	0.0 ± 0.0	0.0 ± 0.0
	RI-FGSM	228.5 ± 12.5	371.5 ± 39.9	422.5 ± 36.8
	RI-FGSM (MULTI)	208.0 ± 9.0	260.0 ± 28.7	285.0 ± 29.6
	RI-FGSM (MULTI-T)	214.5 ± 18.7	297.0 ± 27.1	343.0 ± 29.2
AT-DQN-15/255 (RUN3)	30-STEP PGD	30.5 ± 1.8	2.5 ± 1.0	0.0 ± 0.0
	RI-FGSM	628.5 ± 61.3	660.0 ± 45.6	674.0 ± 10.7
	RI-FGSM (MULTI)	301.5 ± 13.1	521.0 ± 55.1	553.5 ± 19.3
	RI-FGSM (MULTI-T)	332.5 ± 15.9	469.0 ± 58.0	647.5 ± 14.2

Table 23. Detailed experiment results for BankHeist environment. The lowest reward under attacks are marked gray.

		BANKHEIST		
MODEL	ATTACK	ϵ = 5/255	ϵ= 10/255	ϵ =15/255
NCL-AT-DQN (RUN1)	30-STEP PGD	245.0 ± 23.7	1.0 ± 0.7	0.0 ± 0.0
	RI-FGSM	1080.0 ± 16.6	34.0 ± 7.8	22.5 ± 2.0
	RI-FGSM (MULTI)	1003.5 ± 22.7	38.5 ± 4.7	15.5 ± 3.0
	RI-FGSM (MULTI-T)	1025.5 ± 25.5	28.5 ± 6.9	28.0 ± 3.4
NCL-AT-DQN (RUN2)	30-STEP PGD	3.0 ± 1.0	0.0 ± 0.0	0.0 ± 0.0
	RI-FGSM	215.0 ± 42.4	29.0 ± 3.4	7.0 ± 1.7
	RI-FGSM (MULTI)	120.5 ± 14.0	16.5 ± 2.3	6.5 ± 1.1
	RI-FGSM (MULTI-T)	133.0 ± 23.7	21.0 ± 3.4	4.0 ± 1.5
NCL-AT-DQN (RUN3)	30-STEP PGD	740.0 ± 9.7	0.0 ± 0.0	0.0 ± 0.0
	RI-FGSM	1074.0 ± 32.4	57.5 ± 5.1	7.0 ± 1.4
	RI-FGSM (MULTI)	1061.0 ± 18.7	38.0 ± 3.7	7.5 ± 1.4
	RI-FGSM (MULTI-T)	858.5 ± 69.9	59.5 ± 4.7	12.0 ± 1.7
BCL-C-AT-DQN (RUN1)	30-STEP PGD	807.5 ± 82.5	693.0 ± 80.1	248.5 ± 23.7
	RI-FGSM	1281.0 ± 10.7	1261.0 ± 15.5	1175.0 ± 17.6
	RI-FGSM (MULTI)	1282.0 ± 11.1	1252.0 ± 15.5	1184.5 ± 17.3
	RI-FGSM (MULTI-T)	1243.0 ± 7.4	1252.5 ± 14.3	1159.5 ± 21.3
BCL-C-AT-DQN (RUN2)	30-STEP PGD	1048.0 ± 8.0	956.0 ± 17.9	720.0 ± 12.8
	RI-FGSM	1041.0 ± 63.1	1155.0 ± 10.1	1201.5 ± 22.5
	RI-FGSM (MULTI)	1022.0 ± 63.2	1155.5 ± 11.8	1202.0 ± 14.6
	RI-FGSM (MULTI-T)	1050.5 ± 58.7	1163.5 ± 8.9	1186.0 ± 17.6
BCL-C-AT-DQN (RUN3)	30-STEP PGD	1143.5 ± 30.0	988.5 ± 12.3	250.5 ± 14.6
	RI-FGSM	1223.0 ± 10.4	1202.0 ± 15.3	1188.5 ± 19.7
	RI-FGSM (MULTI)	1159.0 ± 19.1	1175.0 ± 16.4	1147.0 ± 25.4
	RI-FGSM (MULTI-T)	1206.5 ± 7.7	1220.0 ± 8.5	1178.5 ± 26.1
BCL-MOS-AT-DQN (RUN1)	30-STEP PGD	1095.5 ± 6.2	664.0 ± 60.6	586.5 ± 105.6
	RI-FGSM	1230.0 ± 11.2	1214.0 ± 7.5	1255.5 ± 9.4
	RI-FGSM (MULTI)	1213.5 ± 9.6	1187.5 ± 8.0	1238.5 ± 10.9
	RI-FGSM (MULTI-T)	1198.5 ± 9.4	1158.0 ± 23.3	1233.0 ± 11.4
BCL-MOS-AT-DQN (RUN2)	30-STEP PGD	1165.5 ± 9.2	922.5 ± 69.5	470.5 ± 35.6
	RI-FGSM	1281.0 ± 4.8	1241.0 ± 7.5	1234.0 ± 17.0
	RI-FGSM (MULTI)	1238.0 ± 5.8	1225.5 ± 7.4	1235.5 ± 13.5
	RI-FGSM (MULTI-T)	1182.0 ± 7.5	1195.0 ± 12.7	1260.0 ± 9.1
BCL-MOS-AT-DQN (RUN3)	30-STEP PGD	1214.0 ± 7.7	1003.5 ± 15.3	113.0 ± 4.0
	RI-FGSM	1258.5 ± 12.0	1270.0 ± 9.2	1142.0 ± 18.7
	RI-FGSM (MULTI)	1261.0 ± 7.7	1258.0 ± 16.7	1154.0 ± 22.8
	RI-FGSM (MULTI-T)	1184.0 ± 8.6	1243.0 ± 10.4	1111.5 ± 22.4

Table 24. Detailed experiment results for BankHeist environment. The lowest reward under attacks are marked gray.

	BANKHEIST			
MODEL	ATTACK	ϵ = 5/255	ϵ= 10/255	ϵ =15/255
NCL-RADIAL-DQN (RUN1)	30-STEP PGD	1272.0 ± 2.9	1264.0 ± 4.1	45.5 ± 2.1
	RI-FGSM	1266.0 ± 7.0	1253.5 ± 7.1	968.0 ± 16.6
	RI-FGSM (MULTI)	1265.0 ± 5.6	1243.0 ± 7.5	810.5 ± 14.4
	RI-FGSM (MULTI-T)	1265.0 ± 5.6	1243.0 ± 7.5	822.0 ± 18.0
NCL-RADIAL-DQN (RUN2)	30-STEP PGD	1242.5 ± 3.5	89.5 ± 9.1	9.0 ± 1.9
	RI-FGSM	1226.5 ± 2.9	69.0 ± 10.8	11.5 ± 1.1
	RI-FGSM (MULTI)	1171.0 ± 4.9	60.5 ± 6.9	12.5 ± 1.0
	RI-FGSM (MULTI-T)	1168.0 ± 3.4	59.5 ± 7.6	10.5 ± 1.1
NCL-RADIAL-DQN (RUN3)	30-STEP PGD	1342.5 ± 5.0	323.5 ± 18.3	18.0 ± 1.3
	RI-FGSM	1342.0 ± 5.4	324.5 ± 16.8	18.5 ± 6.0
	RI-FGSM (MULTI)	1347.0 ± 3.2	214.0 ± 10.5	10.5 ± 1.8
	RI-FGSM (MULTI-T)	1347.0 ± 3.2	198.5 ± 14.1	33.0 ± 5.5
BCL-RADIAL-DQN (RUN1)	30-STEP PGD	1225.5 ± 4.9	1225.5 ± 4.9	931.0 ± 45.0
	RI-FGSM	1225.5 ± 4.9	1225.5 ± 4.9	1043.5 ± 18.1
	RI-FGSM (MULTI)	1225.5 ± 4.9	1223.5 ± 4.1	228.5 ± 13.9
	RI-FGSM (MULTI-T)	1225.5 ± 4.9	1224.5 ± 4.4	248.0 ± 16.0
BCL-RADIAL+AT-DQN (RUN1)	30-STEP PGD	1113.5 ± 3.8	1010.5 ± 8.0	961.5 ± 9.2
	RI-FGSM	1119.0 ± 7.7	1154.5 ± 9.8	1190.5 ± 8.5
	RI-FGSM (MULTI)	1099.5 ± 6.2	1070.5 ± 8.5	1166.0 ± 14.8
	RI-FGSM (MULTI-T)	1093.0 ± 5.3	1094.0 ± 9.1	1169.0 ± 15.6
BCL-RADIAL-DQN (RUN2)	30-STEP PGD	1270.0 ± 2.0	1253.0 ± 2.0	888.5 ± 20.5
	RI-FGSM	1264.5 ± 5.6	1252.5 ± 3.9	576.5 ± 55.4
	RI-FGSM (MULTI)	1251.5 ± 6.0	1226.0 ± 4.9	127.0 ± 10.7
	RI-FGSM (MULTI-T)	1258.0 ± 4.2	1200.5 ± 11.8	137.0 ± 13.5
BCL-RADIAL-DQN (RUN3)	30-STEP PGD	1271.0 ± 7.2	1261.5 ± 9.1	190.5 ± 14.4
	RI-FGSM	1273.5 ± 10.7	1267.0 ± 9.1	381.0 ± 23.2
	RI-FGSM (MULTI)	1242.5 ± 11.9	1161.0 ± 13.9	8.5 ± 1.6
	RI-FGSM (MULTI-T)	1242.5 ± 11.9	1161.0 ± 13.9	17.0 ± 2.7

Table 25. RoadRunner environment. Average episode rewards ± standard error of the mean (SEM) over 20 episodes. The gray rows are the median of three runs (selected based on efficacy score $R_{\text{nominal}} + \frac{1}{3}\sum_{\epsilon} R^{\epsilon}_{\text{adv}}$). We report the lowest rewards among 30-step PGD, RI-FGSM, RI-FGSM (Multi) and RI-FGSM (Multi-T) attacks.

	RoadRunner			
Model/Metric	Nominal	30-step PGD/RI-FGSM Attack		
ϵ	0	5/255	10/255	15/255
DQN (Vanilla)	43390 ± 973	0 ± 0	0 ± 0	0 ± 0
SA-DQN (Convex)	45870 ± 1380	985 ± 207	0 ± 0	0 ± 0
RADIAL-DQN	44595 ± 1165	7195 ± 929	495 ± 116	0 ± 0
AT-DQN-5/255 (Run1)	44065 ± 1896	13150 ± 2116	0 ± 0	0 ± 0
AT-DQN-5/255 (Run2)	39890 ± 2092	20160 ± 1973	0 ± 0	0 ± 0
AT-DQN-5/255 (Run3)	43945 ± 1681	375 ± 200	0 ± 0	0 ± 0
AT-DQN-10/255 (Run1)	42330 ± 936	22430 ± 1948	1000 ± 188	50 ± 17
AT-DQN-10/255 (Run2)	37770 ± 2074	15585 ± 1647	2360 ± 352	0 ± 0
AT-DQN-10/255 (Run3)	37040 ± 1269	22225 ± 1699	2305 ± 795	0 ± 0
AT-DQN-15/255 (Run1)	36580 ± 1634	15860 ± 2118	3650 ± 615	1115 ± 249
AT-DQN-15/255 (Run2)	30000 ± 1314	15640 ± 845	4690 ± 469	1555 ± 307
AT-DQN-15/255 (Run3)	42085 ± 2050	5465 ± 825	20 ± 9	0 ± 0
NCL-AT-DQN (Run1)	43500 ± 2999	40235 ± 2261	1100 ± 234	0 ± 0
NCL-AT-DQN (Run2)	49290 ± 1576	39045 ± 2382	15 ± 8	5 ± 5
NCL-AT-DQN (Run3)	47925 ± 1123	37745 ± 2014	10 ± 10	0 ± 0
NCL-RADIAL-DQN (Run1)	41045 ± 1289	37865 ± 1082	37865 ± 1082	6350 ± 590
NCL-RADIAL-DQN (Run2)	45320 ± 1292	45320 ± 1292	45320 ± 1292	4505 ± 661
NCL-RADIAL-DQN (Run3)	41230 ± 1920	40885 ± 1921	17050 ± 1092	6100 ± 428
BCL-C-AT-DQN (Run1)	44010 ± 1347	33535 ± 2369	13205 ± 1510	4845 ± 399
BCL-C-AT-DQN (Run2)	45815 ± 1422	31305 ± 3590	11405 ± 1385	6335 ± 716
BCL-C-AT-DQN (Run3)	46575 ± 966	35535 ± 1296	19110 ± 2704	6445 ± 929
BCL-MOS-AT-DQN (Run1)	53225 ± 983	36330 ± 3105	15670 ± 1646	300 ± 78
BCL-MOS-AT-DQN (Run2)	44275 ± 1997	40060 ± 1828	15785 ± 1124	1195 ± 180
BCL-MOS-AT-DQN (Run3)	41620 ± 1594	30635 ± 2021	18735 ± 2363	2905 ± 505
BCL-RADIAL-DQN (Run1)	41045 ± 1289	37865 ± 1082	37865 ± 1082	6350 ± 590
BCL-RADIAL+AT-DQN (Run1)	42490 ± 1309	42490 ± 1309	37665 ± 1563	25325 ± 1057
BCL-RADIAL-DQN (Run2)	45320 ± 1292	45320 ± 1292	45320 ± 1292	4505 ± 661
BCL-RADIAL-DQN (Run3)	38725 ± 933	38025 ± 1004	37995 ± 1000	5750 ± 595

Table 26. Detailed experiment results for RoadRunner environment. The lowest reward under attacks are marked gray.

		ROADRUNNER		
MODEL	ATTACK	ϵ = 5/255	ϵ= 10/255	ϵ =15/255
DQN (VANILLA)	30-STEP PGD	0 ± 0	0 ± 0	0 ± 0
SA-DQN (CONVEX)	30-STEP PGD	985 ± 207	0 ± 0	0 ± 0
	RI-FGSM	9740 ± 677	5615 ± 645	3575 ± 490
	RI-FGSM (MULTI)	6170 ± 759	1045 ± 307	105 ± 56
	RI-FGSM (MULTI-T)	6870 ± 833	1220 ± 318	55 ± 25
RADIAL-DQN	30-STEP PGD	7195 ± 929	495 ± 116	0 ± 0
	RI-FGSM	24425 ± 939	7855 ± 637	4605 ± 582
	RI-FGSM (MULTI)	21115 ± 744	9300 ± 656	2330 ± 255
	RI-FGSM (MULTI-T)	22345 ± 700	9225 ± 589	1940 ± 180
AT-DQN-5/255 (RUN1)	30-STEP PGD	13150 ± 2116	0 ± 0	0 ± 0
	RI-FGSM	33110 ± 2316	16520 ± 679	4140 ± 325
	RI-FGSM (MULTI)	31240 ± 1991	10510 ± 997	3215 ± 296
	RI-FGSM (MULTI-T)	32285 ± 2134	8795 ± 1020	820 ± 273
AT-DQN-5/255 (RUN2)	30-STEP PGD	20160 ± 1973	0 ± 0	0 ± 0
	RI-FGSM	33670 ± 2138	11685 ± 1295	1725 ± 221
	RI-FGSM (MULTI)	30245 ± 2492	10750 ± 1145	1495 ± 268
	RI-FGSM (MULTI-T)	35895 ± 1553	6500 ± 908	115 ± 68
AT-DQN-5/255 (RUN3)	30-STEP PGD	375 ± 200	0 ± 0	0 ± 0
	RI-FGSM	29520 ± 2090	8535 ± 539	1590 ± 156
	RI-FGSM (MULTI)	25665 ± 2032	7535 ± 524	1555 ± 132
	RI-FGSM (MULTI-T)	27275 ± 2299	6795 ± 549	235 ± 95
AT-DQN-10/255 (RUN1)	30-STEP PGD	22430 ± 1948	1000 ± 188	50 ± 17
	RI-FGSM	34570 ± 2092	27860 ± 1841	9400 ± 830
	RI-FGSM (MULTI)	30510 ± 1934	25580 ± 2072	7405 ± 601
	RI-FGSM (MULTI-T)	35115 ± 1444	25810 ± 1453	7590 ± 552
AT-DQN-10/255 (RUN2)	30-STEP PGD	15585 ± 1647	2360 ± 352	0 ± 0
	RI-FGSM	26575 ± 2622	21550 ± 2148	15635 ± 1072
	RI-FGSM (MULTI)	22940 ± 2325	19430 ± 1913	9815 ± 807
	RI-FGSM (MULTI-T)	25825 ± 1603	19820 ± 2120	13120 ± 837
AT-DQN-10/255 (RUN3)	30-STEP PGD	22225 ± 1699	2305 ± 795	0 ± 0
	RI-FGSM	29605 ± 1927	29755 ± 2168	5610 ± 398
	RI-FGSM (MULTI)	33095 ± 1701	25010 ± 2207	4295 ± 342
	RI-FGSM (MULTI-T)	32140 ± 2227	27540 ± 2184	4535 ± 545
AT-DQN-15/255 (RUN1)	30-STEP PGD	15860 ± 2118	3650 ± 615	1115 ± 249
	RI-FGSM	31560 ± 1799	22910 ± 1709	21275 ± 1329
	RI-FGSM (MULTI)	27450 ± 2328	23775 ± 1497	18740 ± 1355
	RI-FGSM (MULTI-T)	24630 ± 1795	19015 ± 2005	19375 ± 1242
AT-DQN-15/255 (RUN2)	30-STEP PGD	15640 ± 845	4690 ± 469	1555 ± 307
	RI-FGSM	22430 ± 1273	21180 ± 1289	21350 ± 1069
	RI-FGSM (MULTI)	23655 ± 1662	21615 ± 1204	20415 ± 1283
	RI-FGSM (MULTI-T)	24195 ± 1248	23970 ± 1389	19695 ± 1540
AT-DQN-15/255 (RUN3)	30-STEP PGD	5465 ± 825	20 ± 9	0 ± 0
	RI-FGSM	23305 ± 2632	23110 ± 2399	24815 ± 1342
	RI-FGSM (MULTI)	24205 ± 2379	20985 ± 2002	18800 ± 2135
	RI-FGSM (MULTI-T)	23790 ± 2375	18800 ± 1906	20255 ± 2058

Table 27. Detailed experiment results for RoadRunner environment. The lowest reward under attacks are marked gray.

		ROADRUNNER		
MODEL	ATTACK	ϵ = 5/255	ϵ= 10/255	ϵ =15/255
NCL-AT-DQN (RUN1)	30-STEP PGD	40235 ± 2261	1100 ± 234	0 ± 0
	RI-FGSM	45595 ± 1781	13750 ± 1118	5235 ± 319
	RI-FGSM (MULTI)	45645 ± 1113	10925 ± 1098	4030 ± 328
	RI-FGSM (MULTI-T)	44035 ± 1818	5860 ± 812	985 ± 228
NCL-AT-DQN (RUN2)	30-STEP PGD	39045 ± 2382	15 ± 8	5 ± 5
	RI-FGSM	41365 ± 1850	14235 ± 1387	4480 ± 408
	RI-FGSM (MULTI)	41900 ± 1944	12885 ± 1023	3475 ± 335
	RI-FGSM (MULTI-T)	41160 ± 1669	11555 ± 1141	1775 ± 325
NCL-AT-DQN (RUN3)	30-STEP PGD	37745 ± 2014	10 ± 10	0 ± 0
	RI-FGSM	41145 ± 1886	23715 ± 1367	5345 ± 473
	RI-FGSM (MULTI)	43665 ± 1671	22000 ± 1434	4710 ± 382
	RI-FGSM (MULTI-T)	38025 ± 3114	12940 ± 2058	3475 ± 416
BCL-C-AT-DQN (RUN1)	30-STEP PGD	33535 ± 2369	13205 ± 1510	4845 ± 399
	RI-FGSM	45905 ± 1408	39650 ± 2101	40815 ± 2399
	RI-FGSM (MULTI)	43330 ± 1997	41075 ± 2426	42560 ± 1505
	RI-FGSM (MULTI-T)	45915 ± 1440	40320 ± 2408	42965 ± 1958
BCL-C-AT-DQN (RUN2)	30-STEP PGD	31305 ± 3590	11405 ± 1385	6335 ± 716
	RI-FGSM	39125 ± 2295	35325 ± 2971	40395 ± 1375
	RI-FGSM (MULTI)	43580 ± 2661	38775 ± 1794	35080 ± 2420
	RI-FGSM (MULTI-T)	44490 ± 1806	36695 ± 3274	33810 ± 3044
BCL-C-AT-DQN (RUN3)	30-STEP PGD	35535 ± 1296	19110 ± 2704	6445 ± 929
	RI-FGSM	41405 ± 2278	42030 ± 1810	38930 ± 1571
	RI-FGSM (MULTI)	43230 ± 2042	41785 ± 2535	36575 ± 1618
	RI-FGSM (MULTI-T)	42640 ± 1142	40145 ± 1861	36160 ± 2858
BCL-MOS-AT-DQN (RUN1)	30-STEP PGD	36330 ± 3105	15670 ± 1646	300 ± 78
	RI-FGSM	44285 ± 3146	42440 ± 2481	9730 ± 539
	RI-FGSM (MULTI)	41195 ± 3076	44160 ± 1354	4435 ± 352
	RI-FGSM (MULTI-T)	39615 ± 3609	40940 ± 1490	4535 ± 357
BCL-MOS-AT-DQN (RUN2)	30-STEP PGD	40060 ± 1828	15785 ± 1124	1195 ± 180
	RI-FGSM	39815 ± 2273	40440 ± 2066	12465 ± 596
	RI-FGSM (MULTI)	41645 ± 1604	37375 ± 1993	9815 ± 550
	RI-FGSM (MULTI-T)	41390 ± 1556	39800 ± 1958	9475 ± 950
BCL-MOS-AT-DQN (RUN3)	30-STEP PGD	30635 ± 2021	18735 ± 2363	2905 ± 505
	RI-FGSM	37610 ± 1186	36775 ± 1477	13930 ± 846
	RI-FGSM (MULTI)	38470 ± 1856	38025 ± 1279	12465 ± 850
	RI-FGSM (MULTI-T)	37130 ± 1460	32180 ± 1762	14620 ± 798

Table 28. Detailed experiment results for RoadRunner environment. The lowest reward under attacks are marked gray.

		ROADRUNNER		
MODEL	ATTACK	ϵ = 5/255	ϵ= 10/255	ϵ =15/255
NCL-RADIAL-DQN (RUN3)*	30-STEP PGD	41230 ± 1920	41230 ± 1920	41270 ± 1926
	RI-FGSM	40885 ± 1921	37275 ± 2212	9935 ± 783
	RI-FGSM (MULTI)	40885 ± 1921	18845 ± 966	6325 ± 446
	RI-FGSM (MULTI-T)	40885 ± 1921	17050 ± 1092	6100 ± 428
BCL-RADIAL-DQN (RUN1)	30-STEP PGD	41045 ± 1289	41045 ± 1289	41045 ± 1289
	RI-FGSM	41045 ± 1289	37865 ± 1082	12120 ± 639
	RI-FGSM (MULTI)	37865 ± 1082	38245 ± 1134	7970 ± 740
	RI-FGSM (MULTI-T)	37865 ± 1082	38245 ± 1134	6350 ± 590
BCL-RADIAL+AT-DQN (RUN1)	30-STEP PGD	42490 ± 1309	42490 ± 1309	42490 ± 1309
	RI-FGSM	42490 ± 1309	42570 ± 1310	44115 ± 1726
	RI-FGSM (MULTI)	42490 ± 1309	38650 ± 1836	27500 ± 851
	RI-FGSM (MULTI-T)	42490 ± 1309	37665 ± 1563	25325 ± 1057
BCL-RADIAL-DQN (RUN2)	30-STEP PGD	45320 ± 1292	45320 ± 1292	45320 ± 1292
	RI-FGSM	45320 ± 1292	45320 ± 1292	11945 ± 795
	RI-FGSM (MULTI)	45320 ± 1292	45320 ± 1292	4845 ± 450
	RI-FGSM (MULTI-T)	45320 ± 1292	45320 ± 1292	4505 ± 661
BCL-RADIAL-DQN (RUN3)	30-STEP PGD	38725 ± 932	38725 ± 932	38880 ± 1008
	RI-FGSM	38725 ± 932	38035 ± 1004	8420 ± 652
	RI-FGSM (MULTI)	38025 ± 1004	38685 ± 959	7240 ± 663
	RI-FGSM (MULTI-T)	38025 ± 1004	37995 ± 1000	5750 ± 595

* NCL-RADIAL-DQN (Run1) & (Run2) are the same as BCL-RADIAL-DQN (Run1) & (Run2).

E. Walltime for DQN models

The training time for each run takes around 12 hours on a single GeForce RTX 2080Ti GPU. Below in Table 29 we present the number of runs conducted for each experiment.

Table 29. Number of runs conducted for each experiment.

PONG	RUN1	RUN2	RUN3
AT-DQN-ϵ	1	1	1
NCL-AT-DQN	25	25	25
NCL-RADIAL-DQN	10	4	4
BCL-C-AT-DQN	66	66	66
BCL-MOS-AT-DQN	20	18	20
BCL-RADIAL-DQN	8	21	29
FREEWAY	**RUN1**	**RUN2**	**RUN3**
AT-DQN-ϵ	1	1	1
NCL-AT-DQN	20	20	20
NCL-RADIAL-DQN	6	12	7
BCL-C-AT-DQN	51	51	51
BCL-MOS-AT-DQN	25	26	21
BCL-RADIAL-DQN	20	18	23
BANKHEIST	**RUN1**	**RUN2**	**RUN3**
AT-DQN-ϵ	1	1	1
NCL-AT-DQN	15	15	15
NCL-RADIAL-DQN	10	9	9
BCL-C-AT-DQN	60	60	60
BCL-MOS-AT-DQN	38	43	44
BCL-RADIAL-DQN	18	15	13
BCL-RADIAL+AT-DQN	24	N/A	N/A
ROADRUNNER	**RUN1**	**RUN2**	**RUN3**
AT-DQN-ϵ	1	1	1
NCL-AT-DQN	15	15	15
NCL-RADIAL-DQN	13	13	8
BCL-C-AT-DQN	36	36	36
BCL-MOS-AT-DQN	22	21	18
BCL-RADIAL-DQN	17	17	17
BCL-RADIAL+AT-DQN	20	N/A	N/A