MI-RFGSM: Efficient and Robust Adversarial Attacks Against Deep Reinforcement Learning

Haider Ali, Peter Yurkovich, Jamous Bitrick

haiderali@vt.edu, petery97@vt.edu, jbitrick1@vt.edu

Abstract

The high performance of Deep Reinforcement Learning (DRL) algorithms in complex tasks has increased the need to ensure the robustness of the DRL against advanced adversarial attacks. However, compared to a large volume of studies exploring adversarial attacks that perturb Deep Neural Networks (DNN), adversarial attacks of disrupting DRL have been significantly less explored. The existing adversarial algorithms for DRL have focused on finding the optimal time to attack a DRL agent. However, little work has been explored in injecting perturbation attacks in input data (i.e., adversarial examples) in the context of DRL settings. In this paper, we propose efficient and robust perturbation-based adversarial attacks to disturb the DRL agent's decision-making, called *Momentum Iterative Randomized Fast Gradient Sign Method* (MI-RFGSM). We significantly refined the Randomized Fast Gradient Sign Method (RFGSM) previously studied under DNN settings applicable under DRL environments. We enhanced it further by incorporating *momentum* on each gradient update. We considered targeted or non-targeted attacks under DRL with or without defenses to investigate the efficiency, effectiveness, and robustness of the MI-RFGSM. We conducted extensive experiments using the Deep Q-Network (DQN), Deep Deterministic Policy Gradient (DDPG), and Proximal Policy Optimization (PPO) DRL agents. Each DRL agent type was tested within appropriate discrete (Atari Games) or continuous environments (MuJoCo), comparing our MI-RFGSM against various state-of-the-art DRL attacks in terms of attack execution time, average reward under defenses and attack success rate metrics. Our results proved that our proposed MI-RFGSM attack outperformed all of its existing gradient based counterparts. Our results showed that MI-RFGSM is nine times faster than the state-of-the-art Carlini & Wagner (CW) method while showing the outperformance in robustness and giving a highly comparable attack success rate within discrete environments.

Introduction

Deep Reinforcement Learning (DRL) algorithms learn policies to guide DRL agents to take optimal actions based on an environment state. These algorithms have successfully achieved high performance on the various complex as well as critical tasks, such as robotics (Amarjyoti 2017), autonomous vehicles (Kiran et al. 2021; Ferdowsi et al. 2018), resource allocation (Yoon et al. 2021b), intrusion response systems (Yoon et al. 2021a), various cybersecurity problems (Basori and Malebary 2020), or networking and communication problems (Luong et al. 2019). As DRL has been used to solve multiple problems as above, adversaries aiming to disrupt the DRL process and misleading the DRL agent's decision-making have been known as a severe issue.

A policy, a probabilistic distribution of actions by the DRL agent, is often learned by Deep Neural Networks (DNN) to approximate the action-value function. The vulnerabilities of the DNN to adversarial attacks have been significantly studied (Szegedy et al. 2013; Goodfellow, Shlens, and Szegedy 2014a; Yuan et al. 2019) to mitigate the impact when they are exploited by adversaries. Common adversarial examples include adversarial perturbations imperceptible to humans but fooling DNNs easily in the testing or deploying stage (Yuan et al. 2019). Researchers have explored various attacks and defenses for supervised DNN applications, such as image classification (Goodfellow, Shlens, and Szegedy 2014a) or natural language processing (Alzantot, Balaji, and Srivastava 2018). However, adversarial attacks and defenses are largely unexplored in DRL environments. DRL has numerous critical safety and security applications and accordingly drew our attention to the need for robust DRL. For robust DRL, there is a prerequisite of developing efficient, effective, and robust adversarial attacks which can evaluate the robustness of defense mechanisms.

Researchers have developed adversarial attacks in DRL by answering the following two questions: (1) *How to attack?* and (2) *When to attack?* The first *how-to-attack* question is related to what perturbation method to use for disrupting the state during an episode. The second *when-to-attack* question is associated with identifying an optimal time to attack during an episode. In this work, we focus on answering *how-to-attack* by proposing a novel scheme named *Momentum Iterative Randomized Fast Gradient Sign Method (MI-RFGSM)*. To be specific, **the goal of this work** is to develop robust and fast attacks by generating effective adversarial states in DRL. In addition, we validate the performance of the proposed MI-RFGSM by comparing it against those of the state-of-the-art adversarial attacks under DRL with or without defense where the attacker can perform targeted or non-targeted attacks. We validated the performance of the MI-RFGSM in terms of attack success rate metrics (ASR),

average attack execution time per perturbation (AET), and average reward (AR) by the DRL agent via the extensive comparative performance analyses.

We made the following **key contributions** in this work:

1. We develop the MI-RFGSM, which generates fast and robust adversarial perturbations to compromise a DRL agent under either targeted or non-targeted perturbation attack. Where trained DQN, DDPG, and PPO algorithms are utilized as the DRL agents within appropriate continuous or discrete environments.
2. The proposed MI-RFGSM is an enhanced version of RFGSM (Tramèr et al. 2017), Momentum Iterative FGSM (MI-FGSM) (Dong et al. 2018), and Iterative FGSM (I-FGSM) (Kurakin et al. 2016), to efficiently attack the states of the DRL process.
3. We consider Robust ADversarIAL Loss (RADIAL) (Oikarinen, Weng, and Daniel 2020), State Adversarial (SA) (Zhang et al. 2020), and Alternating Training of Learned Adversaries (ATLA) (Zhang et al. 2021) as a robust defenses and investigated the defensive effect in DRL on the robustness of the MI-RFGSM attack compared to those of the state-of-the-art attacks.
4. To validate the outperformance of the proposed MI-RFGSM, we conduct extensive performance analysis to compare the MI-RFGSM with the state-of-the-art adversarial examples in DRL, including (Carlini and Wagner 2017; Goodfellow, Shlens, and Szegedy 2014b; Madry et al. 2017; Dong et al. 2018; Xie et al. 2019), in terms of ASR, AET, and AR. Among the existing adversarial examples, we devised two baseline examples, which are more robust and scalable, by extending the Diversity Iterative FGSM (DI-FGSM) (Xie et al. 2019) and Momentum Iterative FGSM (MI-FGSM) (Dong et al. 2018) used in DNNs into DRL settings.
5. We consider state of the art attacks and defenses within continuous and discrete environments. We consider traditional attack metrics and propose novel success rate metrics within continuous environments, which are currently missing within the literature. We show MAE as a strong metric in itself, as well as a powerful tool to perform further statistical analysis on to create more accurate and descriptive metrics such as Attack Sensitivity and Binned Success Rate.
6. Our results show that our proposed MI-RFGSM outperformed overall in ASR, AET, and AR in discrete environments of Atari Games played by DQN and RADIAL-DQN. It significantly outperforms the Carlini & Wagner method (CW) in AET while maintaining ASR and AR comparable to other counterparts. In terms of ASR and AR under the defense, MI-RFGSM outperforms the state-of-the-art perturbation methods, proving its robustness. However, for continuous environments of MuJoCo, played by PPO and DDPG, MI-RFGSM was above average and was comparable to CW in some cases. CW outperformed in such environments. We proved that MI-RFGSM is fast and best in discrete environments with and without the defenses whereas our best performing baseline CW was not robust under RADIAL-DQN. In summary, compared to other counterparts, we only give here the results of DQN playing Atari Pong Game with or without defenses: (i) ASR of MI-RFGSM under defense for targeted attacks is 6% more than PGD, 7% more than MI-FGSM, 27% more than DI-FGSM, 64% more than FGSM, and 83% more than CW; (ii) AET of MI-RFGSM under targeted attacks is 634 milliseconds (ms) faster than CW, 46 ms faster than MI-FGSM, 21 ms faster than DI-FGSM, and 12 ms faster than PGD; (iii) AET of MI-RFGSM under non-targeted attacks is 865 ms faster than CW, 40 ms faster than MI-FGSM, 37 ms faster than DI-FGSM, and 19 ms faster than PGD; and (iv) AR of MI-RFGSM is comparable with that of PGD while significantly outperforming those of other baselines.

We have made our codebase available on github at following links: DQN, PPO and DDPG

Related Work

This section provides a brief overview of the state-of-the-art adversarial attacks in Deep Neural Networks (DNN), mainly used for supervised deep learning. Table 1 contains a comprehensive comparison of previous work as well as our own. More comprehensible comparisons between each section are contained within Table 19 and Table 20 within Appendix A. In addition, we briefly discuss existing adversarial attacks which disrupt the DRL process and identify the differences between our proposed MI-RFGSM and their approaches.

Backdoor attacks in Deep Neural Networks (DNNs). Szegedy et al. (2013) initially presented an attack with small perturbations leading to misclassifications. Later, researchers have proposed various attacks and defenses and evaluated their performance in terms of scalability, ASR, and robustness. Goodfellow, Shlens, and Szegedy (2014a) proposed the Fast Gradient Sign Method (FGSM), which was efficient but showed less robust, not guaranteeing a 100% ASR. Carlini and Wagner (2017) proposed the Carlini & Wagner (CW) method that guarantees 100% ASR, but it was slow. Naïve FGSM provided the basis to build more sophisticated and better variants of FGSM, including Randomized FGSM (RFGSM) (Tramèr et al. 2017), Diversity Iterative FGSM (DI-FGSM) (Xie et al. 2019), and Momentum Iterative FGSM (MI-FGSM) (Dong et al. 2018). Madry et al. (2017) proposed the Projected Gradient Descent (PGD), a variant of Iterative FGSM (I-FGSM) using projected gradient descent, which makes it robust. Currently, MI-FGSM and PGD attacks are considered the most efficient and robust state-of-the-art adversarial examples in DNNs. In particular, RFGSM, DI-FGSM, and MI-FGSM are only designed and evaluated in the context of DNNs and have never been used in DRL. In this work, we proposed MI-RFGSM by enhancing RFGSM, incorporating momentum, and applying it in DRL. In addition, we refined DI-FGSM and MI-FGSM to be applicable in DRL and evaluated against the MI-RFGSM in their performance.

Adversarial Attacks in Deep Reinforcement Learning (DRL). Huang et al. (2017) made the preliminary attempt to attack the DRL agent by extending FGSM. Later, state-of-the-art attacks of DRL, such as Lin et al. (2017); Sun et al. (2020), mainly focused on finding an optimal time to attack

Paper	Algorithms	Application	Metrics	Perturbations	Setting	Parameters
Sun et al. (2020)	A3C, DDPG, PPO	Atari (Pong, Breakout), TORCS, MuJoCo	Average Return	C&W	White Box	Number of Steps, Damage Awareness Delta
Lin et al. (2017)	A3C, DQN	Atari (Pong, MsPacman, Seaquest, Qbert, ChopperCommand)	Success rate, Average Reward	C&W	White Box	Number of Steps, Epsilon
Behzadan and Munir (2017)	DQN	Atari (Pong)	Success rate, Average Reward	FGSM, JSMA	Black Box	Number of Observations, Epochs
Huang et al. (2017)	A3C, TRPO, DQN	Atari (Pong, Seaquest, SpaceInvaders, ChopperCommand)	Average Return	FGSM	Both	Epsilon, 3 Norm Constraints
Pattanaik et al. (2017)	DDPG, DDQN	MountainCar, MuJoCo (Hopper, HalfCheetah, CartPole)	Average Return	GB	White Box	MuJoCo Physical Parameters
Kos and Song (2017)	A3C	Atari (Pong)	Average Reward	FGSM, Random Noise	White Box	Epsilon, Noise Amount, Value Function
Our Attack	DQN, PPO, DDPG	Atari (Pong, BankHeist, RoadRunner), MuJoCo (Ant)	Success Rate, Average Reward, Attack Execution Time, Novel Continuous Attack Success Rate	MI-RFGSM, I-RFGSM, FGSM, C&W, Robust Sarsa, MAD, PGD, MI-FGSM, DI-FGSM	White Box	Epsilon, Alpha, Number of Steps

Table 1: Environment Setup of papers

the DRL agent. However, Lin et al. (2017); Sun et al. (2020) ignored the question of *how-to-attack* and blindly used the state-of-the-art state perturbation technique, such as Carlini & Wagner (CW) method. However, CW is very slow and less robust under defense in DRL. Therefore, we compared our perturbation method with the CW to prove the effectiveness of MI-RFGSM.

Fast perturbation methods, such as naïve FGSM (Goodfellow, Shlens, and Szegedy 2014b), have been used in DRL by (Huang et al. 2017); however, the naïve FGSM is easily detectable. Another variant of FGSM, called PGD, is one of the state-of-the-art DRL attacks as it is fast and more robust than the CW. However, the state-of-the-art defenses in DRL (Zhang et al. 2020; Oikarinen, Weng, and Daniel 2020; Fischer et al. 2019) have recently challenged the robustness of PGD-based attacks (Madry et al. 2017), which were originally considered in DNNs. The Robust Sarsa (RS) attack which learns a a value function to create perturbations and the Maximal Action Difference (MAD) attack which maximizes the difference from the desired action have also been proposed (Zhang et al. 2020), but only consider the average return metric. Therefore, there is a critical need to develop scalable, effective, and robust state perturbation attacks in DRL settings, which is the goal of our paper.

Preliminaries

This section provides an overview of the DRL and adversarial states generation methods considered in this work.

Deep Reinforcement Learning (DRL)

Reinforcement learning (RL) algorithms optimize the expected cumulative reward by training a policy π. This policy can be a deterministic or probabilistic function that maps a state s to action a, $\pi : S \rightarrow A$, where S and A are state and action spaces, respectively. S and A can be constructed of high or low dimensional discrete or continuous spaces. In DRL, this policy function π is learned by a neural network. Deep Q-Networks (DQN) (Mnih et al. 2013), Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al. 2015), and Proximal Policy Optimization (PPO) (Schulman et al. 2017) are three well known DRL algorithms.

Deep Q-Networks (DQN). In Q-learning, the Q-value is the expected cumulative discounted reward when action a is taken in state s. DQN is a kind of Q-learning where Q-values are learned by a neural network (NN) while minimizing the squared Bellman error. DQN has a feature, called *Experience Replay*, which helps decrease the high variance due to Q-learning updates. In addition, DQN has *Replay Buffer* to store all recent transitions. Random samples are taken from this buffer to mitigate the correlation due to time. DQN takes action based on the maximum Q-value. DQN has been utilized best in discrete environments.

Deep Deterministic Policy Gradient (DDPG). DDPG utilizes an *Actor-Critic Network* containing a Critic which learns the Q-values and an Actor which learns the actions to take. The Critic intakes action a and state s and outputs the Q-value which is learned by minimizing the difference from the next environmental reward output. The Actor intakes

state s and outputs action a which is learned from maximizing the Q-Values from the Critic Network. DDPG also utilizes a *Replay Buffer*, but instead of using an *Experience Replay* it trains *Off-Policy* to reduce variance from Q-learning updates. Off-Policy training utilizes a target network for exploration, with a second main network which learns the policy and will periodically partially update the target network. DDPG was proposed for better control within continuous environments.

Proximal Policy Optimization (PPO). PPO operates on an Actor-Critic Network. Just like DDPG an Actor-Critic network consists of an Actor, who observes action s and outputs action a, and a Critic network that determines an output network. PPO implements clipped surrogate objectives and continuous operation. Clipped surrogate objectives observe policy gradient objectives after every actor action and truncates the objective observations within +-1 epsilon (the hyperparameter). PPO performs policy updates between every action utilizing clipped surrogate actions to prevent small changes in an action from causing large changes in the model. PPO was proposed to work in a continuous environment requiring frequent policy updates.

Adversarial States Generation Methods

We will compare the performance of the following methods (or variants of them) against that of our MI-RFGSM: FGSM (Goodfellow, Shlens, and Szegedy 2014a), Carlini & Wagner (CW) (Carlini and Wagner 2017), Projected Gradient Descent (PGD) (Madry et al. 2017), Diversity Iterative FGSM (DI-FGSM) (Xie et al. 2019), Robust Sarsa and MAD Attacks (Zhang et al. 2020). Due to the space constraint, interested readers can refer to the specified corresponding references.

We leveraged the following three methods (i.e., I-FGSM, MI-FGSM and RFGSM) to develop our MI-RFGSM.

Iterative Fast Gradient Sign Method (I-FGSM). Kurakin et al. (2016) proposed I-FGSM, a variant of FGSM, taking multiple small steps of size α in the direction of the gradient. However, FGSM takes only one step of size ϵ to make perturbation small enough. This method also clips the result of each step by ϵ. I-FGSM outperformed FGSM. Starting with $x'_0 = 0$, on every iteration, it performs:

$$x'_i = x'_{i-1} - \text{clip}_\epsilon(\alpha \cdot \text{sign}(\nabla \text{loss}_{F,t}(x'_{i-1}))), \quad (1)$$

where α is a step in the direction of the gradient.

Randomized FGSM (RFGSM). A single-step FGSM method has a problem of converging to a degenerate global minimum. I-FGSM obfuscates a linear approximation of the loss due to small steps. Due to these problems, they generate weak perturbations, which can be easily defended. To tackle this problem, Tramèr et al. (2017) presented RFGSM, which adds a small random step to FGSM to escape the non-smooth vicinity of the data point before linearizing the model's loss. Single-step RFGSM is computationally efficient and has a high ASR than I-FGSM in DNNs. It performs:

$$x^{\text{adv}}_{RFGSM} = x' + (\epsilon - \alpha) \cdot \text{sign}(\nabla_{x'} \text{loss}(x', y_{\text{true}})), \quad (2)$$

where

$$x' = x + \alpha \cdot \text{sign}(\mathcal{N}(0^d, I^d)). \quad (3)$$

Momentum Iterative FGSM (MI-FGSM). Dong et al. (2018) proposed MI-FGSM, which is a variant of FGSM that integrates the momentum on each step of an iterative FGSM to escape from poor local maxima and stabilize update directions. Starting with $x'_0 = 0$, on every iteration, it performs:

$$g_i = \mu \cdot g_{i-1} + \frac{\nabla \text{loss}_{F,t}(x'_{i-1})}{||\nabla \text{loss}_{F,t}(x'_{i-1})||_1}, \quad (4)$$

$$x'_i = x'_{i-1} - \text{clip}_\epsilon(\alpha \cdot \text{sign}(g_i)), \quad (5)$$

where μ is the decay factor and g_i is the accumulated gradient at iteration i.

Problem Statement

A DRL agent interacts with an environment and learns policy π to choose an action a given a state s. This policy π can be a probabilistic model $\pi(\text{s}, \text{a}) \sim [0, 1]$, which gives the probability of taking an action a given a state s. The π can also be a deterministic model where we can obtain an action a directly from the policy function π given the state $s : a = \pi(\text{s})$. The goal of the DRL agent is to maximize the cumulative reward R_o by learning an optimal policy π^*. The reward that the DRL agent aims to maximize is the expected discounted reward until the next $T - 1$ time steps, represented by:

$$R_o = \sum_{t=0}^{T-1} E_{a_t \sim u(s_t)} \Big[\gamma^t r(s_t, a_t) \Big]. \quad (6)$$

Instead of maximizing this reward, an adversary aims to minimize reward R_o by adding a perturbation δ_t into the agent's observation s_t to mislead the agent to take an adverse action a_t. The adversary has to generate perturbation δ as small as possible to be undetected. Out of all time steps, the adversary may or may not choose to add perturbation δ to the state s_t based on his strategy. In this way, the adversary's expected cumulative reward is given by:

$$R_{\text{adv}} = \sum_{t=0}^{T-1} E_{a_t \sim u(s_t + x_t \delta_t)} \Big[\gamma^t r(s_t, a_t) \Big], \quad (7)$$

where x_t at given time t is 0 if the adversary does not inject any perturbation; otherwise, it returns 1.

We need to find the perturbations $\delta_0, \delta_1, \ldots, \delta_{T-1}$ for all time steps during an episode to compromise the DRL agent to take the corresponding adversarial actions $a^{\text{adv}}_0, a^{\text{adv}}_2, \ldots, a^{\text{adv}}_{T-1}$. These adversarial actions should then minimize the reward of the DRL agent, R_o, and the reward of the DRL agent under defense, R_{defense}. We also want to find each perturbation δ_t in the minimum time possible.

Proposed Approach: MI-RFGSM

We propose the Momentum Iterative Randomized Fast Gradient Sign Method (MI-RFGSM) to find the robust and effective perturbations in the minimum time possible. MI-RFGSM is designed by combining RFGSM, MI-FGSM, and

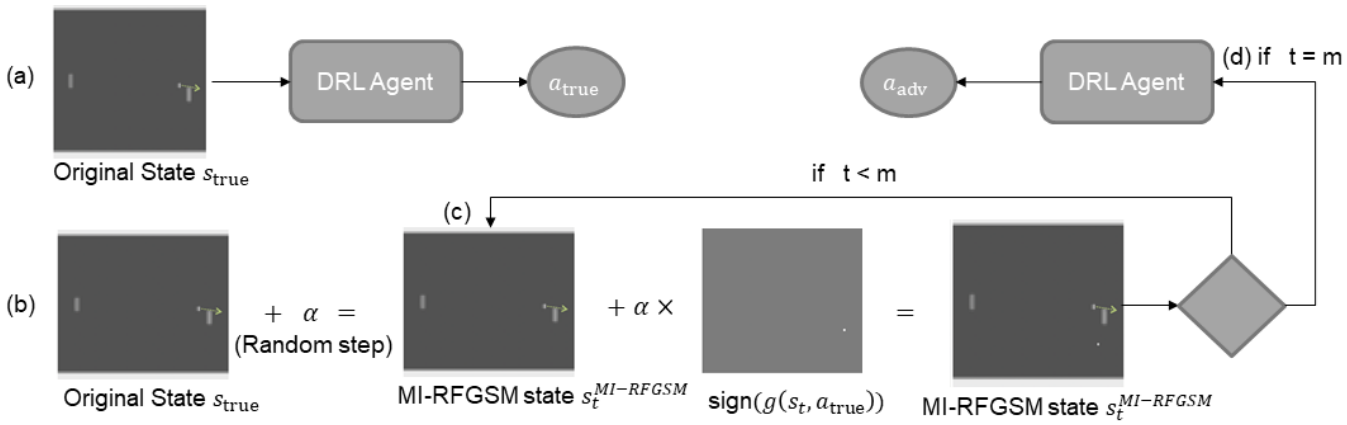


Figure 1: The overview of Non-Targeted MI-RFGSM: MI-RFGSM attacks a single frame during an episode to compromise a DRL agent following: (a) DRL agent takes an true non-adversarial state s_{true} to give non-adversarial, true action, a_{true}; (b) Start by adding the random step of size α to s_{true}; (c) Until the number of steps, m, compute the MI-RFGSM state $s_t^{\text{MI-RFGSM}}$ by calibrating the momentum-based accumulated gradient, g, using s_{true} and a_{true}, and then clipping it with α; and (d) Compute adversarial action, a_{adv}, by giving a final MI-RFGSM adversarial state $s_{m-1}^{\text{MI-RFGSM}}$ to the DRL agent.

I-FGSM and extending their applications from DNNs to DRL. This novel combination of FGSM variants has never been used in DRL setting as well as DNN setting before.

For the extension to the DRL setting, we consider true label y_{true} as a the action produced by the policy of DRL agent, represented by:

$$a_{\text{true}} = \text{DRLAgent}(s_{\text{true}}), \tag{8}$$

where s_{true} is the original non-adversarial state and a_{true} is the original non-adversarial action given by the DRL agent.

In MI-RFGSM, we start with taking a step of size α in the random direction as in Eq. (9). Then, at each step t, we calculate the adversarial state $s_t^{\text{MI-RFGSM}}$ by taking the step of size α in the direction of gradient g. Starting with:

$$s_0^{\text{MI-RFGSM}} = s_{\text{true}} + \alpha \cdot \text{sign}(\mathcal{N}(0^d, I^d)). \tag{9}$$

On every iteration i, it performs:

$$g_t = \mu \cdot g_{t-1} + \frac{\nabla_{s'} \text{loss}(s', a_{\text{true}})}{||\nabla_{s'} \text{loss}(s', a_{\text{true}})||_1}, \tag{10}$$

$$s_t^{MI-RFGSM} = s_{t-1}^{MI-RFGSM} + \alpha \cdot \text{sign}(g_t), \tag{11}$$

where μ is the decay factor and g_t is the accumulated gradient at iteration t.

$$a_{\text{adv}} = \text{DRLAgent}(s^{\text{MI-RFGSM}}). \tag{12}$$

Eq. (11) means that we move policy π away from an optimal action by using the direction of the gradient. We make it iterative to take multiple gradient direction steps. Multiple gradient steps move the policy away from the optimal action, contributing to generating high ASR and low AR. Adding a random step in the start contributes to enhancing this method robust as it allows to escape the non-smooth vicinity of the data point before linearizing the model's loss. Eq. (10) shows the addition of a momentum term in gradient on each step. This solves the attack becoming stuck at a poor local maxima and gives the gradient a necessary boost, called a momentum, to try to reach the global or optimal maxima. The momentum also stabilizes the gradient updates. Hence, MI-RFGSM is effective in terms of ASR and AR because it tries to reach the global maxima, which helps minimize AR and maximize ASR as much as possible. In summary, MI-RFGSM uses the novel combination of *random start of fixed size alpha* α, *iterative* steps of size alpha α and *momentum* in DRL setting. To apply in DRL setting, it uses true state s_{true} and true action a_{true} at a given time step t to compute the gradients. We summarize the procedures of MI-RFGSM in Figure 1.

Differences between FGSM variants and MI-RFGSM are highlighted in Figure 2. Differences between PGD (i.e., the most similar baseline to ours) and MI-RFGSM is that MI-RFGSM takes a fixed size step α in the random direction instead of uniformly choosing the random point. This is the same α which MI-RFGSM takes in iterative steps after the random start. We think MI-RFGSM is relatively faster than PGD because we already have the step size, α that is sometimes smaller than the step size of PGD as PGD can choose a distant random step. In addition, incorporating *momentum* will add extra time cost in MI-RFGSM. Hence, we tested MI-RFGSM with and and without the momentum to investigate its impact. In maximizing ASR and minimizing AR under defense, MI-RFGSM performs better than the PGD because in selecting the size of a random step, PGD may choose a distant step, making it noticeable by the defender algorithms. However, our proposed MI-RFGSM is more effective against defenses as it uses fixed step α in a random direction, making it less noticeable. Other baselines including CW are not robust due to lack of random initialization.

Our proposed MI-RFGSM attacks all steps during an episode to evaluate the effectiveness, efficiency, and robustness of MI-RFGSM in all situations during the episode. In addition, this allows a fair comparison of our MI-RFGSM with the other state-of-the-art attacks. We always consider

$x_t = 1$ in Eq. (7) as we attacked all number of steps during the episode by $\sum_{t=0}^{T-1} x_t = T$ where T is the same as an episode length.

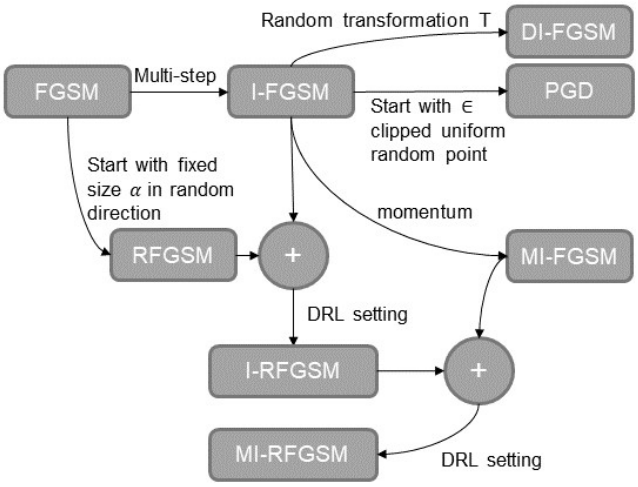


Figure 2: Difference between FGSM variants and our proposed MI-RFGSM and I-RFGSM.

Experimental Setup

Metrics. We use the following traditional **metrics** for our analyses:

- **Average Attack Execution Time Per Perturbation (AET)** captures the average time required to generate a perturbed state. For the respective attacks, targeted or non-targeted, we measure AET by $\text{AET}_T = \frac{\mathcal{T}(N_{NTS})}{N_S}$ and $\text{AET}_T = \frac{\mathcal{T}(N_{TS})}{N_S}$, where N_S is the total number of adversarial states computed, $\mathcal{T}(N_{NTS})$ is the total times elapsed to generate all non-targeted adversarial states, and $\mathcal{T}(N_{TS})$ is the total times elpased to generate all targeted adversarial states.
- **Average Reward (AR)** measures the average reward of the rewards of all episodes. Given N_e be the number of episodes and r_i be the total rewards accumulated during an episode i, AR is measured by $\text{AR} = \frac{\sum_{i=0}^{N} r_i}{N_e}$.
- **Attack Success Rate (ASR)** measures the total number of attack successes over the total number of attack attempts. Considering either targeted or non-targeted attacks, ASR is measured by $\text{ASR}_{NT} = \frac{N_{NT}^{AS}}{N_{NT}}$ and $\text{ASR}_T = \frac{N_T^{AS}}{N_T}$, where N_{NT}^{AS} refers to the total number of successes by non-targeted attacks and N_{NT} is the total number of attempts by non-targeted attacks. Similarly, N_T^{AS} refers to the total number of successes by targeted attacks and N_T is the total number of attempts by targeted attacks. Under non-targeted attacks, a failure indicates the case where the attack is entirely unable to succeed or the DRL agent takes an action maximizing the reward. An attack attempt is defined by one-time attack per time step. On the other hand, targeted attacks mean generating a perturbation targeting to lead the DRL agent to take a targeted action. For example, in Atari Pong, a targeted attack can aim to make the DRL agent take a targeted action 'up' at a given point. Thus, reward reduction by other causes, rather than the perturbation targeted attack, is not simply considered as attack success in ASR_T.

Traditional metrics are important to understand and utilize within experiments, however as with any pure numeric metrics there are a number of weaknesses and lack of comprehensions which are apparent within them. Average Reward is important to utilize as minimizing rewards will be a primary attack vector and should be considered, however, not every attack will be solely focused on reducing overall reward. Attacks which seek to cause an action at a single point of time need to be considered, and average reward fails to encompass that. In addition, the usage of the Average Reward metric as a common baseline can lead to a comparison of finding the worst action for every timestep, which may or may not be a part of an attack and can lead newly developed attacks to only focus on this which could lead to a failure of development in other areas. Attack Success is one metric which does look at causing an action at a single point of time. However, with attack success rate defined as it is by the literature, it can only be utilized within discrete action environments. Within continuous environments, extremely subtle changes within the environment, can lead to extremely minuscule changes within the continuous actions. These resulting actions however, are not the same as the original and as such different success rate metrics for continuous environments need to be defined. Each metric below is constructed in a novel way or is being used in a novel way within continuous action success rates. The following **novel** continuous action success rate metrics have been utilized within our experiment.

- **Action Mean Average Error (MAE)** measures the average difference between each continuous action. Smaller MAE values are better. Given N as the number of actions, a_i as the ith action, p_i as the ith peturbed action, and t_i as the ith target action at each time step, Action MAE is measured by $\text{MAE}_T = \frac{\sum_{i=0}^{N} |t_i - p_i|}{N}$ and $\text{MAE}_{NT} = \frac{\sum_{i=0}^{N} |a_i - p_i|}{N}$ for targeted and nontargeted attacks respectively.
- **Action Attack Sensitivity (AS)** inverses the MAE value to create a scalable metric for increasingly small values of MAE. A value of 1 AS would indicate a MAE of 0.1 and a value of 3 AS would indicate a MAE of 0.001. AS is measured by $\text{AS} = -\log MAE$.
- **Binned Success Rate (BSR)** discretized success rate for continuous control environments. The BSR takes in a accuracy bin size, and determines if the perturbed action is within the bin size for a targeted attack or outside the bin size for an untargeted attack. For a targeted attack, a difference of 0.02 would pass a bin size of 0.1 but fail a bin size of 0.01, and vice versa for a untargeted attack. Given a accuracy bin size of B, a_i as the ith action, p_i as the ith peturbed action, t_i as the ith target action at each time step, the non-targeted attack success for any given bin, S_{NT}^B, and the targeted attack success for any given bin, S_T^B, can be measured by $\text{S}_{NT}^B = (|a_i - p_i| > B)$ and

$S_T^B = (|t_i - p_i| < B)$. The binned success rate for N samples can then be measured by $\text{BSR}_{NT}^B = \frac{\sum_{i=0}^{N} S_{NT}^B}{N}$ and $\text{BSR}_T^B = \frac{\sum_{i=0}^{N} S_T^B}{N}$. Since targeted attacks will look to minimize the difference from the target action, and non-targeted attacks will look to maximize the difference from the original action, binned success rate has inverse meaning between the two. Success in a small bin would indicate a small difference, which is shows a good targeted attack and a poor non-targeted attack. Bin sizes are important to choose correctly, because success in a single bin doesn't indicate that for targeted attacks a smaller bin wouldn't be possible and for non-targeted attacks a larger bin wouldn't be possible. As such, at least two bin sizes are required at a bare minimum to demonstrate the accuracy of any attack, targeted or non-targeted.

Comparing Schemes. We use the following perturbation attacks to be compared against our MI-RFGSM. As discussed earlier, we did not consider Lin et al. (2017); Sun et al. (2020) as comparing schemes because they focus on answering the second question, *when-to-attack*, rather than *how-to-attack*. The comparing schemes include:

- **Fast Gradient Sign Method (FGSM)** (Goodfellow, Shlens, and Szegedy 2014b): We chose this as our baseline because it has been not only extensively used in DNN settings but also has been numerous times used in DRL settings, such as *uniform attack* (Huang et al. 2017).
- **Carlini & Wagner Method (CW)** (Carlini and Wagner 2017): This method is considered one of the well-known state-of-the-art methods in DRL settings guaranteeing 100% ASR. The state-of-the-art end goal-based DRL attacks (Lin et al. 2017; Sun et al. 2020) use the CW method to generate targeted perturbations. We adapt CW for usage within continuous action and observation environments by utilizing a more general f function and update to best attack.
- **Projected Gradient Method (PGD)** (Madry et al. 2017): PGD is not much popular in end goal-based DRL attacks, but it is considered to be the robust, fast, and successful state-of-the-art attack in DRL settings. Due to its robustness, PGD is usually employed by defenders to test their defenses as in (Zhang et al. 2020; Oikarinen, Weng, and Daniel 2020; Fischer et al. 2019).
- **Momentum Iterative Fast Gradient Sign Method (MI-FGSM)** (Dong et al. 2018): We extended MI-FGSM from DNN settings to DRL settings and compared it with our method. MI-FGSM is the state-of-the-art adversarial example in DNN settings. This effective method has not been enhanced to be applied in DRL before.
- **Diversity Iterative Fast Gradient Sign Method (DI-FGSM)** (Xie et al. 2019): We also extended DI-FGSM from DNNs to DRL and compared it with our method. We included it as our baseline due to its similarity with MI-RFGSM in considering generalizability. DI-FGSM is not utilized in continuous control environments due to its focus on adjusting images.
- **Maximal Action Difference (MAD)** (Zhang et al. 2020): MAD attack is included for its usage within the SA defense literature and its unique perspective as a maximized untargeted attack. MAD attack has only been evaluated on the AR metric before, which we run under other metrics.
- **Robust Sarsa (RS)** (Zhang et al. 2020): Robust Sarsa is included within our comparisons as the only trained attack for targeting a specific model. It has also only been evaluated on the AR metric before, and more metrics are utilized for our usage here.

Perturbation Method	Steps (m)	DQN and PPO		DDPG	
		Epsilon	Alpha	Epsilon	Alpha
CW	1000	NA	NA	NA	NA
PGD	20	8/255	2/255	0.2	0.03
DI-FGSM	20	8/255	2/255	NA	NA
MI-FGSM	20	8/255	2/255	0.2	0.03
FGSM	1	8/255	NA	0.2	NA
I-RFGSM	20	8/255	2/255	0.2	0.03
MI-RFGSM	20	8/255	2/255	0.2	0.03

Table 2: Optimal Values of the Parameters identified Under Each Perturbation Method

Tuned Parameters. We tuned the following parameters for optimizing the performance of the MI-RFGSM: The size of a perturbation (ϵ), the number of steps of perturbation (m), and the size of the step (α).

Defense Setting in DRL. There are very few defenses proposed for adversarial attacks in the DRL. The conventional adversarial training defense has been widely used in DNNs (Kurakin, Goodfellow, and Bengio 2016; Madry et al. 2017) and DRL (Kos and Song 2017; Pattanaik et al. 2017; Behzadan and Munir 2017). However, recently more robust and efficient defense methods were proposed specifically for DRL, such as Robust ADversarIAl Loss (RADIAL) (Oikarinen, Weng, and Daniel 2020), State Adversarial (SA) (Zhang et al. 2020), and Alternating Training of Learned Adversaries (ATLA) (Zhang et al. 2021).

We compare robustness of our attack under RADIAL, SA and ATLA within the different DRL agents. RADIAL (Oikarinen, Weng, and Daniel 2020) improved the robustness of DRL agents by designing the adversarial loss functions with robustness verification bounds during training. It leverages robustness verification bounds to keep the loss as low as possible because low loss leads to better performance of the DRL agent. It primarily parametrizes k to minimize the loss where $L_{adv} = kL_S + (1-k)L_W$. RADIAL applies robustness verification algorithms on the DRL to obtain the layer-wise output bounds of Q-Networks and uses these output bounds to calculate an upper bound of the original loss function under worst-case adversarial perturbation L_W, given L_S is the standard loss function.

SA (Zhang et al. 2020) utilizes a State Adversarial Markovian Decision Process (SA-MDP) which introduces $v(s)$ adversary which perturbs the observed input state while not changing the underlying state. The agents action $\pi(a|v(s))$ is then potentially sub-optimal. The agent is then able to be trained under the range of action of the adversary. ATLA (Zhang et al. 2021) furthers the SA-MDP by training a optimal $v(s)$ policy and training a DRL model under the optimal adversary.

Environmental Setup. We consider a white-box attack where an adversary does not have access to the training time. However, the adversary has the test time access of the DRL

DRL	Environment	Defenses	Attacks	Targeted/ Non-Targeted
DQN	Atari (Pong, RoadRunner, BankHeist)	No Defense, RADIAL-DQN	C&W, PGD, DIFGSM, MIFGSM, FGSM, MI-RFGSM, I-RFGSM	Both
DDPG	MuJoCo(Ant)	No Defense, SA	C&W, PGD, DIFGSM, MIFGSM, FGSM, MI-RFGSM, I-RFGSM	Both
PPO	MuJoCo(Ant)	No Defense, SA, ATLA	C&W, PGD, DIFGSM, MIFGSM, FGSM, MI-RFGSM, I-RFGSM	Non-Targeted

Table 3: Comparison Of Experiments on Different DRL Algorithms.

agent where it knows the neural network architecture and has access to craft its adversarial state. For our experiments, we utilize trained DQN, DDPG and PPO models using the implementation of (Mnih et al. 2015), (Zhang et al. 2020), and (Zhang et al. 2021) to play within several environments. All of the DRL algorithms were run using Pytorch and Open AI Gym. The discrete DQN was used within Atari games, specifically Pong, Road Runner and Bank Heist because of their prevalence in the literature. When we observed that state-of-the-art CW method is not robust when RADIAL-DQN is playing Pong, we extended our experiments to Road Runner an BankHeist to see similar results. For Bank Heist and Road Runner, we did not test on Vanilla DQN as our main goal was to strengthen our conclusion about state-of-the-art CW method under RADIAL-DQN. The continuous focused DDPG and PPO were run under the continuous MuJoCo simulation environment, specifically Ant and Hopper. Both were chosen due to their prevalence in the literature, with Ant specifically being chosen due to its longer episode length. We chose Ant for both PPO and DDPG since its a common practice in literature to test a same application on multiple algorithms. We did not choose Atari Games for PPO and DDPG since discrete environments are not fit for continuous algorithms like DDPG and PPO. For PPO, we included non-targeted variants only. For DQN, we could not use SA and ATLA because we could not train models due to lack of time and infrastructure. Rather, we focused on pre-trained models made available by RADIAL, ATLA and SA authors. Similarly, for PPO, we could not use RADIAL as pre-trained models were not available. For DDPG, we could not use ATLA and RADIAL for the similar reasons. However, in total, we have been able to conduct 15 comprehensive experiments in total against 4 environments, 3 DRL algorithms, 7 state-of-the-art baseline attacks and 3 defenses as shown in Table 4.

The loss function is the cross-entropy loss for gradient-based attacks. We have used L_∞ norm in all FGSM-based baselines, including MI-RFGSM. For the CW, we have used L_2 norm. Assuming that the attacker will have the computational power equivalent to the commodity CPUs, we used the Google Colaboratory CPU (AMD EPYC 7B12, 2 CPUs @ 2.3 GHz, 13 GB RAM) for DQN experiments and personal computers for DDPG and PPO due to the inability of MuJoCo to be run within Google Colab. DDPG was run on a a 3070ti, 32 GB of RAM and a 5950x. PPO experiments were run on Intel(R) Core(TM) i7-9750H CPU @ 2.60 GHz 32 GB RAM. We run every experiment 100 times to minimize the changes in metrics due to random seeds and environmental factors. We reported the average and standard deviation of AR and AETs for each experiment, as well as success rate metrics appropriate for the environment. Table 4 gives a full comparison of environments, attacks, defenses, target and DRL algorithms in terms of specifically AR.

Algorithm 1: **Evaluation Experiment**

Inputs:
$A \leftarrow$ an actions set
$s_0 \leftarrow$ an initial non-adversarial state
Parameters:
$\mathrm{targeted} \leftarrow$ return 1 if attack is targeted; 0 otherwise
$\mathrm{defense} \leftarrow$ return 1 if defense is applied; 0 otherwise
$\mathrm{PerturbationMethod}() \leftarrow$ a perturbation method used
for each episode **do**
  for each step t during an episode **do**
    if targeted is true **then**
      $a_t^{adv*} = \mathrm{RandomStrategy}(A)$
      $s_t^{adv} = \mathrm{PerturbationMethod}(s_t, a_t^{adv*})$
    else
      $s_t^{adv} = \mathrm{PerturbationMethod}(s_t)$
    end if
    if defense is true **then**
      $a_t^{adv} = \mathrm{DRL}_{\mathrm{defense}}(s_t^{adv})$
    else
      $a_t^{adv} = \mathrm{DRL}(s_t^{adv})$
    end if
    $r_t^{adv}, s_{t+1}, \mathrm{done} = \mathrm{Perform}(a_t^{adv})$
    if done is true **then**
      break
    end if
  end for
end for

We conducted extensive experiments to evaluate the efficiency, effectiveness, and robustness of the two variants of the MI-RFGSM compared against the other attacks are targeted or non-targeted under no defense or defenses.

For non-targeted attacks, we avoid the desired action found by the DRL model. For targeted attacks, we take a particular perturbation to mislead a DRL agent to the desired action. To evaluate the proposed two variants of the MI-RFGSM in DRL, we considered random actions in perturbation methods to evaluate their abilities to generate a targeted action. We detailed the designed algorithm in Algorithm 1.

We compared the baseline attacks (i.e., CW, PGD, DI-FGSM, MI-FGSM, FGSM, Robust Sarsa, MAD) with the two variants of MI-RFGSM. We incorporated momentum in first variant of MI-RFGSM while second variant is only a randomized iterative version without the momentum, which we call I-RFGSM. The number of steps (m) is fixed to 20, optimal for FGSM-based perturbation attacks. However, the CW method performed best at $m = 1000$, which was used in our experiment. For a fair comparison, we also added CW results at $m = 20$. We considered naïve FGSM, using $m = 1$ by definition, as a baseline. We fixed $\epsilon = 8/255$ and $\alpha = 2/255$ for our experiments of DQN and PPO, which are identified as optimal after rigorous sensitivity analysis on all considered perturbation methods. We fixed $\epsilon = 0.2$ and $\alpha = 0.03$ for DDPG based upon other research re-

Sr. No	DRL Algorithm	Environment	Defenses	Targeted/ Non-Targeted	Best AR	Comparable AR to best	Worst AR	Best ASR	Comparable ASR to best	Worst ASR
1-	DQN	Atari Pong	No Defense	Non-Targeted	**MI-RFGSM**	All	NA	**MI-RFGSM**	All	FGSM
2-	DQN	Atari Pong	No Defense	Targeted	CW	**MI-RFGSM**	DI-FGSM	CW	**MI-RFGSM**	DI-FGSM
3-	DQN	Atari Pong	RADIAL-DQN	Non-Targeted	**MI-RFGSM**, PGD	I-RFGSM, MI-FGSM	CW, FGSM	**MI-RFGSM**, PGD, I-RFGSM	NA	CW, FGSM
4-	DQN	Atari Pong	RADIAL-DQN	Targeted	**MI-RFGSM**, PGD, MI-FGSM	NA	CW, FGSM	**I-RFGSM**	**MI-RFGSM**, PGD, MI-FGSM	CW, FGSM
5-	DQN	Atari RoadRunner	RADIAL-DQN	Non-Targeted	**MI-RFGSM**	NA	CW	**MI-RFGSM, I-RFGSM**	MI-FGSM	CW
6-	DQN	Atari RoadRunner	RADIAL-DQN	Targeted	MI-FGSM	**MI-RFGSM**, **I-RFGSM**, PGD	CW	**MI-RFGSM**	NA	CW
7-	DQN	Atari BankHeist	RADIAL-DQN	Non-Targeted	CW, **MI-RFGSM**	All	NA	**MI-RFGSM**	MI-FGSM	CW
8-	DQN	Atari BankHeist	RADIAL-DQN	Targeted	CW, **MI-RFGSM**	All	NA	**MI-RFGSM**	NA	CW
9-	DDPG	MuJoCo (Ant)	No Defense	Non-Targeted	CW	**MI-RFGSM**, PGD	MI-FGSM, FGSM	CW	NA	FGSM, MIFGSM
10-	DDPG	MuJoCo (Ant)	No Defense	Targeted	CW	NA	NA	**MI-RFGSM**	CW, MIFGSM	FGSM
11-	DDPG	MuJoCo (Ant)	SA	Non-Targeted	CW	NA	FGSM, MI-FGSM	CW	NA	FGSM, MIFGSM
12-	DDPG	MuJoCo (Ant)	SA	Targeted	CW	NA	NA	CW	NA	FGSM
13-	PPO	MuJoCo (Ant)	No Defense	Non-Targeted	CW	NA	MAD	CW, **MI-FGSM**	FGSM Variants	NA
14-	PPO	MuJoCo (Ant)	SA	Non-Targeted	CW	**MI-RFGSM**, FGSM Variants	Robust Sarsa, MAD	**MI-RFGSM**, FGSM Variants	NA	CW
15-	PPO	MuJoCo (Ant)	ATLA	Non-Targeted	CW	NA	MAD, Robust Sarsa	**MI-RFGSM**	FGSM Variants	CW

Table 4: Comparison Of 15 Experiments against different DRL algorithms, defenses, and environments.

Perturbation Method (steps)	No Defense		Defense with RADIAL-DQN	
	Non-targeted	Targeted	Non-targeted	Targeted
CW (1000)	100%	97%	3%	0%
CW (20)	100%	0%	3%	0%
PGD (20)	100%	82%	99%	77%
DIFGSM (20)	100%	73%	73%	54%
MIFGSM (20)	100%	83%	75%	76%
FGSM (1)	85%	76%	28 %	19%
MI-RFGSM (20)	**100%**	**83%**	**98%**	**77%**
I-RFGSM (20)	**100%**	**84%**	**99%**	**83%**

Table 5: Comparison of Attack Success Rate for Pong using DQN

sults (Zhang et al. 2020). All the optimal values of the parameters under each method are summarized in Table 2.

Results & Analyses

Performance Analysis in Attack Execution Time (AET). Table 7 shows the comprehensive comparison between MI-RFGSM variants and the baselines in AET when tested on DQN playing Pong. MI-RFGSM variants outperformed all baselines and showed comparable or better AET than PGD, which performed best among the baselines. Non-targeted I-RFGSM, the best performing MI-RFGSM, is 12 ms faster than the non-targeted PGD attack. For targeted attacks, PGD shows comparable performance with MI-RFGSM. I-RFGSM outperformed again in AET for targeted attacks. Interestingly, CW with 20 steps is faster than most FGSM variants with 20 steps, while CW performs poorly with 20 steps. This speed is due to CW not needing to complete gradient calculations at every time step. MI-RFGSM (20 steps) is six to nine times faster than the state-of-the-art CW (1000 steps) with better robustness. Similarly, Table 17 is the AET comparison when DDPG and SA DDPG is playing Ant under non-targeted and targeted variants. Similar trends can be seen in DDPG. Overall, FGSM variants are taking similar time and MI-RFGSM is particularly 18-20 times faster than CW. Table 8 and Table 9 are the AET comparisons when DQN is playing BankHeist and RoadRunner respectively. We did not do performance analysis for PPO since, we are gaining same results over DDPG and DQN.

Perturbation Method (steps)	No Defense		Defense with RADIAL-DQN	
	Non-targeted	Targeted	Non-targeted	Targeted
CW (1000)	-21.00 ± 0.00	-21.00 ± 0.00	$+20.85 \pm 0.36$	$+20.50 \pm 0.50$
CW (20)	-21.00 ± 0.00	$+20.75 \pm 0.43$	$+20.70 \pm 0.46$	$+20.80 \pm 0.40$
PGD (20)	-21.00 ± 0.00	-20.39 ± 0.8	-20.96 ± 0.20	-20.44 ± 0.80
DI-FGSM (20)	-21.00 ± 0.00	-19.97 ± 1.27	-19.87 ± 1.32	-16.78 ± 2.67
MI-FGSM (20)	-21.00 ± 0.00	-20.30 ± 1.06	-20.56 ± 0.75	-20.47 ± 0.73
FGSM (1)	-21.00 ± 0.00	-20.62 ± 0.75	$+20.75 \pm 0.43$	$+16.80 \pm 7.88$
MI-RFGSM (20)	$\mathbf{-21.00 \pm 0.00}$	$\mathbf{-20.35 \pm 0.8}$	$\mathbf{-20.91 \pm 0.29}$	$\mathbf{-20.32 \pm 0.95}$
I-RFGSM (20)	$\mathbf{-21.00 \pm 0.00}$	$\mathbf{-20.28 \pm 0.9}$	$\mathbf{-20.90 \pm 0.36}$	$\mathbf{-20.24 \pm 0.91}$

Table 6: Comparison of Average Reward for Pong using DQN.

Performance Analysis in Average Reward (AR). Table 6 shows the comprehensive comparison between MI-RFGSM variants and the baselines in terms of AR when DQN is playing Pong. For Atari Pong, -21 is the least possible reward, which is desired by the attacker, while $+21$ is the maximum possible reward desired by the defender. For non-targeted attacks under no defense, all methods achieved the minimum possible AR of -21. However, under defense, MI-RFGSM variants outperformed all baselines except PGD and showed

Perturbation Method (steps)	Attack Execution Time (ms)	
	Non-Targeted	Targeted
CW (1000)	716 ± 32	963 ± 20
MIFGSM (20)	128 ± 8	138 ± 8
DIFGSM (20)	103 ± 6	135 ± 14
PGD (20)	94 ± 6	117 ± 13
CW (20)	21 ± 2	26 ± 2
FGSM (1)	6 ± 2	6.3 ± 0.7
MI-RFGSM (20)	$\mathbf{126 \pm 7}$	$\mathbf{125 \pm 7}$
I-RFGSM (20)	$\mathbf{82 \pm 6}$	$\mathbf{98 \pm 6}$

Table 7: Comparison of Attack Execution Time for Pong using DQN.

Perturbation Method (steps)	Attack Execution Time (ms)	
	Non-Targeted	Targeted
CW (1000)	693 ± 55	715 ± 134
MIFGSM (20)	92 ± 6	91 ± 6
DIFGSM (20)	101 ± 5	100 ± 6
PGD (20)	87 ± 5	94 ± 7
CW (20)	21 ± 2	21 ± 2
FGSM (1)	5 ± 0.5	5 ± 0.6
MI-RFGSM (20)	$\mathbf{122 \pm 5}$	$\mathbf{122 \pm 5}$
I-RFGSM (20)	$\mathbf{87 \pm 7}$	$\mathbf{92 \pm 11}$

Table 8: Comparison of Attack Execution Time for BankHeist using DQN.

comparable results with PGD. While MI-RFGSM performs comparably to PGD in AR, it still outperforms all baselines in ASR and ASR under defense. MI-FGSM is better than DI-FGSM in ASR and AR. When no defense is used, the performance of CW in ASR and AR with $m = 1000$ has no match; however, it showed poor robustness and inefficiency in AET. Naïve FGSM also works only under no defense.

Table 10 also shows the comprehensive comparison when RADIAL-DQN is playing RoadRunner and BankHeist. Lower reward means better attack here too. For RoadRunner, MI-RFGSM performs best and CW performs poor, showing CW's poor robustness against RADIAL-DQN, strengthening our results. For BankHeist, however, CW is quite comparable to MI-RFGSM.

Table 12 shows the comprehensive results for MuJoCo Ant environment played by Vanilla DDPG and SA DDPG (defense) under non-targeted and targeted variants. The insights can also be seen in Table 4 where CW is outperforming in terms of AR. MI-RFGSM was comparable in one experiment of no defense non-targeted.

Table 13 shows the comprehensive results for MuJoCo Ant environment played by Vanilla PPO, SA PPO (defense) and ATLA PPO (defense) under non-targeted variant. CW again outperformed here and MI-RFGSM was comparable in one experiment.

Overall, MI-RFGSM outperformed in discrete environments of Atari and under defense of RADIAL. However, under continuous environments of MuJoCo, CW did considerably well in terms of AR under defenses of SA and ATLA. MI-RFGSM did not perform worse under any condition however, CW performed worst or did not perform under RADIAL defense. MI-RFGSM outperformed or was comparable in most cases and was above average in other cases. It would be an interesting result to see how CW performs under RADIAL-PPO and RADIAL-DDPG because CW does not perform under RADIAL-DQN. We are comparing our results mostly with CW in continuous environments because CW was best performing in continuous environments.

Perturbation Method (steps)	Attack Execution Time (ms)	
	Non-Targeted	Targeted
CW (1000)	777 ± 90	790 ± 94
MIFGSM (20)	77 ± 5	81 ± 5
DIFGSM (20)	91 ± 8	93 ± 5
PGD (20)	79 ± 5	82 ± 5
CW (20)	22 ± 17	21 ± 9
FGSM (1)	5 ± 0.4	5 ± 0.5
MI-RFGSM (20)	$\mathbf{78 \pm 5}$	$\mathbf{82 \pm 5}$
I-RFGSM (20)	$\mathbf{81 \pm 5}$	$\mathbf{82 \pm 7}$

Table 9: Comparison of Attack Execution Time for RoadRunner using DQN.

Perturbation Method (steps)	BankHeist		RoadRunner	
	Non-Targeted	Targeted	Non-Targeted	Targeted
CW (1000)	0 ± 0	0 ± 0	14000 ± 1000	18400 ± 15400
CW (20)	0.33 ± 1.8	1 ± 3	13683 ± 18511	13787 ± 18508
PGD (20)	2.33 ± 5.59	4 ± 6.11	17 ± 45	23 ± 76
DI-FGSM (20)	1 ± 3.96	4 ± 6.63	87 ± 106	280 ± 399
MI-FGSM (20)	0.67 ± 2.49	3.67 ± 5.47	3 ± 18	3 ± 18
FGSM (1)	0 ± 0	2.33 ± 4.23	247 ± 394	220 ± 338
MI-RFGSM (20)	$\mathbf{1.67 \pm 4.53}$	$\mathbf{3 \pm 5.26}$	$\mathbf{0.00 \pm 0.00}$	$\mathbf{73 \pm 254}$
I-RFGSM (20)	$\mathbf{1 \pm 3}$	$\mathbf{3 \pm 4.58}$	$\mathbf{13 \pm 34}$	$\mathbf{33 \pm 79}$

Table 10: Comparison of Average Reward for RoadRunner and BankHeist using Radial-DQN

Performance Analysis in Discrete Attack Success Rate(ASR). Within discrete action environments, only the traditional usage of ASR is needed. Table 5 shows the comprehensive comparison between MI-RFGSM variants and the baselines in ASR when tested on DQN playing Pong. MI-RFGSM variants outperformed all baselines except for CW (1000) for targeted attacks on no defense. Within the RADIAL defense, all alternatives are impacted, however the MI-RFGSM variants are impacted the least. CW, however, completely fails in both non-targeted and targeted attacks, performing worse that even naïve FGSM. PGD performs comparable to the worst MI-RFGSM variant, but doesn't match the same results that the best variant shows.

Performance Analysis of Continuous Success Metrics. Table 15 and Table 14 show full evaluations of the different metrics we have proposed for the Ant environment running under DDPG. As Attack sensitivity is the same value as MAE, the analysis section will focus on MAE for consistency. For targeted attacks under no defense, MI-RFGSM and CW outperform all other perturbation methods, being only 0.0044 MAE and 0.2% binned success rate of a bin size of 0.1 apart from one another. For targeted attacks under the SA defense, CW performs better than all other options, however, MI-RFGSM variants remain consistent or better than the rest of the perturbation options for both MAE and binned success rates. For untargeted attacks CW performed best both without and with the SA defense enabled in both MAE and binned success rates. However, MI-RFGSM variants again performed equal to or better than the rest of the alternatives within untargeted attacks.

The Table 16 shows the binned attack success rates for untargeted attacks in the Ant environment run using PPO, under no defense, and SA and ATLA defenses. CW performs the best under no defense, but not substantially so compared to the MI-RFGSM variants. Under defenses, there was an interesting result between different bin sizes. While CW sig-

Perturbation Method (steps)	BankHeist		RoadRunner	
	Non-Targeted	Targeted	Non-Targeted	Targeted
CW (1000)	0%	0%	1%	0%
CW (20)	18%	0%	2%	0%
PGD (20)	99%	78%	99%	89%
DIFGSM (20)	88%	68%	89%	79%
MIFGSM (20)	100%	88%	100%	91%
FGSM (1)	47%	47%	91%	85%
MI-RFGSM (20)	**100%**	**90%**	**100%**	**95%**
I-RFGSM (20)	**99%**	**79%**	**100%**	**91%**

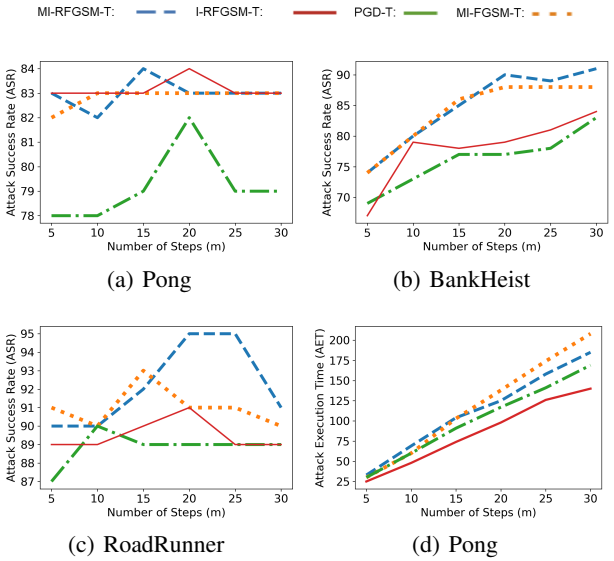
Table 11: Comparison of Attack Success Rate for RoadRunner and BankHeist using Radial-DQN

nificantly outperformed in the larger (harder) bin size, it underperformed for the smaller (easier) bin sizes. This would indicate that CW is more accurate and precise within its attacks under defense, but also has the greatest chance for any perturbation to completely fail to produce any results, regardless of the accuracy needed.

Performance Analysis across different algorithms and environments

Table 4 shows the comparison in terms of AR and ASR of 15 different experiments run on 3 different DRL algorithms, 3 defenses, 7 baseline attacks and 4 environments under targeted and non-targeted variants. For discrete environments such as Atari Pong, RoadRunner, BankHeist, MI-RFGSM was best in terms of AR and ASR under defenses and no defenses as compared to CW and PGD. PGD was comparable in some cases. We can conclude that for DQN, MI-RFGSM outperformed and CW was either comparable or worst. However, for DDPG and PPO or continuous environments, CW outperformed and MI-RFGSM was above average. Having taken approximately 10 times less time than CW, MI-RFGSM is performing above average in continuous environments, which is quite an achievement. In terms of ASR and AR, CW has performed worst for many number of experiments, which shows its unreliability.

Sensitivity Analyses We extensively parameterized each perturbation method to identify the optimal parameter values for comparison, as summarized in Table 2. Here we compared the most promising four perturbation methods (i.e., FGSM-based attacks) we observed in our experiment, which are MI-RFGSM, I-RFGSM, MI-FGSM, and PGD, denoted by MI-RFGSM-T, I-RFGSM-T, MI-FGSM-T, and PGD-T in Figure 3(a), respectively. This figure is for DQN playing Pong game. We varied $m = 5, 10, 15, 20, 25$, and 30 and analyzed their impact on ASR and AET. For non-targeted attacks under no defense, we found no change in ASR and AR. For targeted attacks under no defense, we found $m = 20$ to be optimal for all perturbation methods. Figure 3(a) clearly shows the outperformance of MI-RFGSM in ASR for the targeted attacks. Figure 3(b) and Figure 3(c) shows similar trends for BankHeist and RoadRunner respectively when RADIAL-DQN is playing under targeted attacks. We also see that at $m = 20$, there is optimal performance for all methods. I-RFGSM-T performs the best at $m = 20$. MI-RFGSM-T performs the best at $m = 15$, with an AET of 104 ms, faster than 20 steps baselines, including PGD. We can also observe the change in AET when varying m under the targeted perturbation attacks, as shown in Figure 3(d). I-RFGSM is the fastest for targeted attacks. In Figure 3(d),



(a) Pong (b) BankHeist

(c) RoadRunner (d) Pong

Figure 3: Sensitivity analysis of MI-RFGSM, MI-RFGSM, MI-FGSM, and PGD under varying the number of steps (m) in Attack Success Rate (ASR) and Attack Execution Time (AET). (a)ASR of targeted attack under no defense for DQN playing Pong. (b) ASR of targeted attack under defense for RADIAL-DQN playing BankHeist. (c)ASR of targeted attack under defense for RADIAL-DQN playing RoadRunner. (d)AET of targeted attack for DQN playing Pong

we observe that AET clearly increases linearly as m increases under all perturbation methods. For CW, we varied $m = 5, 10, 15, 20, 50, 200, 500$, and 1000 and found 1000 being optimal in ASR and AR.

Discussion and Future Work

Although our proposed MI-RFGSM is a combination of previously designed FGSM variants, yet, it proved to outperform all baseline attacks. For future work, we want to check the performance of MI-RFGSM and CW against RADIAL-PPO and RADIAL-DDPG, so we know whether RADIAL can beat CW in continuous environments or not. Furthermore, we want to build a novel attack on top of CW that would be faster and more robust than CW. As far as the realistic analysis of MI-RFGSM is concerned, we included white-box attack, which might not be considered as realistic. But, it is known that we can easily convert white-box attacks to black-box attacks using transfer-ability principles. Our attack is scalable, which is always required in real word scenarios. Overall, we claim that our attack completely works under realistic scenarios.

Conclusion

In this paper we proposed an efficient and robust perturbation attack applicable in deep reinforcement learning (DRL) environments, MI-RFGSM. Our validation procedures applied algorithms, applications, and metrics from previous, referenced papers. We also incorporated the best preforming perturbations from all the surveyed papers and introduced several novel FGSM based perturbations. Our environments were built from DQN, DDPG or PPO DRL al-

Perturbation Method (steps)	Non-Targeted		Targeted	
	Vanilla DDPG	SA DDPG	Vanilla DDPG	SA DDPG
CW (1000)	-112 ± 66	-1961 ± 442	-130 ± 80	336 ± 140
CW (20)	-1430 ± 1000	394 ± 63	-418 ± 475	762 ± 170
FGSM (1)	455 ± 171	2282 ± 415	-80 ± 73	1264 ± 35
MIFGSM (20)	458 ± 188	2320 ± 347	-84 ± 80	1213 ± 76
PGD (20)	-799 ± 817	839 ± 31	-28 ± 56	1156 ± 101
I-RFGSM (20)	-45 ± 26	816 ± 75	-20 ± 45	1210 ± 55
MI-RFGSM (20)	-199 ± 243	871 ± 115	-99 ± 73	1216 ± 81

Table 12: Comparison of Targeted Perturbation Methods on DDPG in Average Reward (AR) for Ant.

Perturbation method (steps)	No Defense PPO	Defense with SA-PPO	Defense with ATLA-PPO
CW (1000)	-699 ± 810	527 ± 265	207 ± 151
CW (20)	-142.9 ± 224.9	1684.7 ± 846.7	1596 ± 693
PGD (20)	744.95 ± 4.0	886.6 ± 1.23	856.2 ± 2.7
MI-FGSM (20)	751.5 ± 9.9	889.1 ± 4.7	855.3 ± 2.2
FGSM (1)	742.12 ± 12.3	886.99 ± 1.5	855.9 ± 1.87
Robust Sarsa	900 ± 444	4171 ± 56	4722 ± 469
MAD Attack	2009 ± 311	4284 ± 165	5183 ± 96
MI-RFGSM (20)	748.6 ± 10.5	889.1 ± 4.7	854.4 ± 1.9
I-RFGSM (20)	747.43 ± 13.2	886.6 ± 1.23	858.4 ± 2.3

Table 13: Comparison of Non-targeted Perturbation Methods in Average Reward (AR) for Ant. 5687 is the highest achieved average reward for Vanilla PPO under no attack

gorithms operated on Atari Pong, RoadRunner, BankHeist, and MuJoCo Ant and Hopper. Our evaluation of success rate metrics within continuous action spaces shows MAE as both a valid metric as well as a strong metric to create more specific variations of such as Attack Sensitivity and Binned Success Rate. We validated MI-RFGSMs performance by comparing it with the state-of-the-art attack algorithms in attack execution time (AET), and average reward (AR), discrete attack success rate (ASR), and novel continuous attack success rate metrics. We obtained the following **key findings** from our study: (1) FGSM-based perturbation methods outperform CW attack in ASR and AR under defense in discrete DRL; (2) The proposed MI-RFGSM showed the most robust, scalable, and effective perturbation attacks among all compared state-of-the-art schemes in Discrete environments played by DQN; (3) The proposed MI-RFGSM performed six to nine times faster than state-of-the-art CW attack, with better discrete ASR under the defense. In addition, MI-RFGSM outperformed PGD, the best baseline, in ASR under defense and AET while showing comparable AR to PGD; (4) CW outperformed FGSM variants under continuous environments and defenses such as SA and ATLA. However, this research shows that CW shows no robustness against RADIAL defense, whereas our proposed MI-RFGSM outperformed against RADIAL, SA and ATLA, proving its robustness. Overall, MI-RFGSM is robust and fast in both discrete and continuous environments.

References

Alzantot, M.; Balaji, B.; and Srivastava, M. 2018. Did you hear that? adversarial examples against automatic speech recognition. *arXiv preprint arXiv:1801.00554*.

Amarjyoti, S. 2017. Deep reinforcement learning for robotic manipulation-the state of the art. *arXiv preprint arXiv:1701.08878*.

Basori, A. H.; and Malebary, S. J. 2020. Deep reinforcement learning for adaptive cyber defense and attacker's pattern identification. In *Advances in Cyber Security Analytics and Decision Systems*, 15–25. Springer.

Behzadan, V.; and Munir, A. 2017. Vulnerability of Deep Reinforcement Learning to Policy Induction Attacks. *CoRR*, abs/1701.04143.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, 39–57. IEEE.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 9185–9193.

Ferdowsi, A.; Challita, U.; Saad, W.; and Mandayam, N. B. 2018. Robust deep reinforcement learning for security and safety in autonomous vehicle systems. In *The IEEE 21st International Conference on Intelligent Transportation Systems (ITSC)*, 307–312.

Fischer, M.; Mirman, M.; Stalder, S.; and Vechev, M. 2019. Online robustness training for deep reinforcement learning. *arXiv preprint arXiv:1911.00887*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014a. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014b. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Huang, S. H.; Papernot, N.; Goodfellow, I. J.; Duan, Y.; and Abbeel, P. 2017. Adversarial Attacks on Neural Network Policies. *CoRR*, abs/1702.02284.

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

Kos, J.; and Song, D. 2017. Delving into adversarial attacks on deep policies. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*.

Kurakin, A.; Goodfellow, I.; Bengio, S.; et al. 2016. Adversarial examples in the physical world.

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

Lin, Y.-C.; Hong, Z.-W.; Liao, Y.-H.; Shih, M.-L.; Liu, M.-Y.; and Sun, M. 2017. Tactics of Adversarial Attack on Deep Reinforcement Learning Agents. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 3756–3762.

Luong, N. C.; Hoang, D. T.; Gong, S.; Niyato, D.; Wang, P.; Liang, Y.-C.; and Kim, D. I. 2019. Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Comms. Surveys & Tutorials*, 21(4): 3133–3174.

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

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

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533.

Oikarinen, T.; Weng, T.-W.; and Daniel, L. 2020. Robust deep reinforcement learning through adversarial loss. *arXiv preprint arXiv:2008.01976*.

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

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*.

Sun, J.; Zhang, T.; Xie, X.; Ma, L.; Zheng, Y.; Chen, K.; and Liu, Y. 2020. Stealthy and Efficient Adversarial Attacks against Deep Reinforcement Learning. *Proc. the AAAI Conference on Artificial Intelligence*, 34(04): 5883–5891.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Tramèr, F.; Kurakin, A.; Papernot, N.; Goodfellow, I.; Boneh, D.; and McDaniel, P. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*.

Xie, C.; Zhang, Z.; Zhou, Y.; Bai, S.; Wang, J.; Ren, Z.; and Yuille, A. L. 2019. Improving transferability of adversarial examples with input diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2730–2739.

Yoon, S.; Cho, J.-H.; Dixit, G.; and Chen, R. 2021a. Resource-Aware Intrusion Response Based on Deep Reinforcement Learning for Software-Defined Internet-of-Battle-Things. *Game Theory and Machine Learning for Cyber Security*.

Yoon, S.; Cho, J.-H.; Kim, D. S.; Moore, T. J.; Free-Nelson, F.; and Lim, H. 2021b. DESOLATER: Deep Reinforcement Learning-Based Resource Allocation and Moving Target Defense Deployment Framework. *IEEE Access*, 9: 70700–70714.

Yuan, X.; He, P.; Zhu, Q.; and Li, X. 2019. Adversarial Examples: Attacks and Defenses for Deep Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 30(9): 2805–2824.

Zhang, H.; Chen, H.; Boning, D.; and Hsieh, C.-J. 2021. Robust Reinforcement Learning on State Observations with Learned Optimal Adversary. *arXiv preprint arXiv:2101.08452*.

Zhang, H.; Chen, H.; Xiao, C.; Li, B.; Liu, M.; Boning, D.; and Hsieh, C.-J. 2020. Robust deep reinforcement learning against adversarial perturbations on state observations. *arXiv preprint arXiv:2003.08938*.

Appendix A

Perturbation Method (steps)	MAE		Attack Sensitivity		Bin 1.0 BSR		Bin 0.1 BSR	
	Vanilla	SA	Vanilla	SA	Vanilla	SA	Vanilla	SA
CW (1000)	1.4730 ± 0.1765	0.9389 ± 0.1222	-0.1650 ± 0.0534	0.0313 ± 0.0594	87.57%	39.35%	99.28%	99.72%
CW (20)	0.9314 ± 0.2356	0.3912 ± 0.1661	0.0459 ± 0.1179	0.4427 ± 0.1734	45.88%	5.43%	93.71%	83.06%
FGSM (1)	0.2186 ± 0.1491	0.0940 ± 0.0442	0.7629 ± 0.3112	1.0845 ± 0.2448	1.64%	0%	60.27%	36.99%
MIFGSM (20)	0.2324 ± 0.1564	0.0947 ± 0.0439	0.7384 ± 0.3199	1.0784 ± 0.2353	2.01%	0%	61.82%	37.25%
PGD (20)	1.1533 ± 0.2036	0.3698 ± 0.06423	-0.0550 ± 0.0784	0.4461 ± 0.0811	63.80%	0.72%	97.13%	85.18%
I-RFGSM (20)	1.1938 ± 0.1974	0.2955 ± 0.0847	-0.0708 ± 0.0735	0.4966 ± 0.1241	66.58%	0.47%	96.82%	82.54%
MI-RFGSM (20)	1.1643 ± 0.2114	0.2955 ± 0.0856	-0.0586 ± 0.0813	0.5486 ± 0.1330	63.67%	0.27%	96.91%	79.82%

Table 14: Comparison Of Non-Targeted Attacks on DDPG in MAE, AS, BSR.

Perturbation Method (steps)	MAE		Attack Sensitivity		Bin 1.0 BSR		Bin 0.1 BSR	
	Vanilla	SA	Vanilla	SA	Vanilla	SA	Vanilla	SA
CW (1000)	0.0826 ± 0.0362	0.2101 ± 0.0702	1.1141 ± 0.1596	0.7039 ± 0.1571	99.99%	100%	68.42%1	29.07%
CW (20)	0.5414 ± 0.1361	0.4344 ± 0.4344	0.2808 ± 0.1138	0.3756 ± 0.1110	84.82%	96.60%	12.85%	12.90%
FGSM (1)	0.4410 ± 0.4410	0.4848 ± 0.1098	0.3752 ± 0.1326	0.3263 ± 0.1039	91.79%	95.69%	15.75%	10.70%
MIFGSM (20)	0.0910 ± 0.0518	0.3850 ± 0.1057	1.1014 ± 0.2280	0.4326 ± 0.1302	99.99%	99.35%	66.52%	14.29%
PGD (20)	0.1424 ± 0.05428	0.3737 ± 0.1039	0.8753 ± 0.1575	0.4462 ± 0.1332	99.97%	99.58%	42.80%	14.66%
I-RFGSM (20)	0.1343 ± 0.0525	0.3773 ± 0.1050	0.9019 ± 0.1606	0.4421 ± 0.1329	99.98%	99.55%	45.40%	14.58%
MI-RFGSM (20)	0.0870 ± 0.0504	0.3820 ± 0.1050	1.1213 ± 0.2280	0.4362 ± 0.1316	99.99%	99.42%	68.62%	14.36%

Table 15: Comparison Of Targeted Attacks on DDPG.

Perturbation method (steps)	No Defense PPO (0.2)	No Defense PPO (0.3)	Defense with SA-PPO (0.2)	Defense with SA-PPO (0.3)	Defense with ATLA-PPO (0.2)	Defense with ATLA-PPO (0.3)
CW (1000)	100%	100%	88%	77%	88%	56%
CW (20)	100%	94%	13%	2%	54%	7%
PGD (20)	99%	91%	99%	0%	93%	15%
MI-FGSM (20)	99%	93%	99%	0%	93%	16%
FGSM (1)	99%	90%	99%	0%	92%	8%
MI-RFGSM (20)	99%	92%	99%	0%	92%	23 %
I-RFGSM (20)	99%	93%	99%	0%	94%	13%

Table 16: Comparison of Non-targeted Perturbation Methods in Binned Success Rate for Ant.

Perturbation Method (steps)	Non-Targeted		Targeted	
	Vanilla DDPG	SA DDPG	Vanilla DDPG	SA DDPG
CW (1000)	492 ± 160	393 ± 123	332 ± 131	387 ± 106
CW (20)	6.6 ± 1.8	6.2 ± 1.1	5.5 ± 0.9	5.6 ± 0.5
FGSM (1)	1.0 ± 0.1	1.2 ± 0.5	1.0 ± 0.1	1.0 ± 0.3
MIFGSM (20)	20.7 ± 2.4	20.5 ± 1.6	20.3 ± 0.8	21.9 ± 1.1
PGD (20)	18.3 ± 0.6	19.2 ± 1.0	18.7 ± 0.9	19.9 ± 0.8
I-RFGSM (20)	24.6 ± 3.0	19.0 ± 0.9	19.2 ± 0.8	20.0 ± 0.7
MI-RFGSM (20)	26.2 ± 3.5	20.2 ± 1.2	20.6 ± 1.0	21.4 ± 1.3

Table 17: Comparison of Targeted Perturbation Methods on DDPG in Attack Execution Time (AET) for Ant.

Perturbation method (steps)	No Defense PPO Targeted	Defense with SA-PPO Targeted	Defense with ATLA-PPO Targeted
CW (1000)	3 ± 1	0 ± 0	2 ± 0
CW (20)	5 ± 2	129 ± 119	3 ± 1
PGD (20)	4 ± 0	193 ± 1	196 ± 59
MI-FGSM (20)	4 ± 0	193 ± 1	213 ± 37
FGSM (1)	4 ± 0	192 ± 1	229 ± 12
MAD Attack	1576 ± 483	3324 ± 675	1681 ± 628
MR-RFGSM (20)	4 ± 0	193 ± 1	205 ± 43
RFGSM (20)	4 ± 0	193 ± 1	196 ± 56

Table 18: Comparison of targeted attacks on PPO in Average Reward (AR) for Hopper.

Paper	Algorithm					Applications		Metrics			
	A3C	DQN	DDPG	PPO	TRPO	Atari	MuJoCo	Average Return	Attack Execution Time	Success Rate	Continuous Success Rate
Sun et al. (2020)	✓	×	✓	✓	×	✓	✓	✓	×	×	×
Lin et al. (2017)	✓	✓	×	×	×	✓	×	✓	×	✓	×
Behzadan and Munir (2017)	×	✓	×	×	×	✓	×	✓	×	✓	×
Huang et al. (2017)	✓	✓	×	×	✓	✓	×	✓	×	×	×
Pattanaik et al. (2017)	✓	×	✓	✓	×	✓	✓	✓	×	×	×
Kos and Song (2017)	✓	×	×	×	×	✓	×	✓	×	×	×
Our Attack	×	✓	✓	✓	×	✓	✓	✓	✓	✓	✓

Table 19: Comparison of Deep Reinforcement Learning within Papers

Paper	Perturbations												Settings	
	C&W	FGSM	JSMA	GB	Random Noise	MI-RFGSM	RFGSM	Robust Sarsa	MAD	PGD	MIFGSM	DIFGSM	White Box	Black Box
Sun et al. (2020)	✓	×	×	×	×	×	×	×	×	×	×	×	✓	×
Lin et al. (2017)	✓	×	×	×	×	×	×	×	×	×	×	×	✓	×
Behzadan and Munir (2017)	×	✓	✓	×	×	×	×	×	×	×	×	×	✓	✓
Huang et al. (2017)	×	✓	×	×	×	×	×	×	×	×	×	×	✓	✓
Pattanaik et al. (2017)	×	×	×	✓	×	×	×	×	×	×	×	×	✓	×
Kos and Song (2017)	×	✓	×	×	✓	×	×	×	×	×	×	×	✓	×
Our Attack	✓	✓	×	×	×	✓	✓	✓	✓	✓	✓	✓	✓	×

Table 20: Comparison of Deep Reinforcement Learning Attacks within Papers