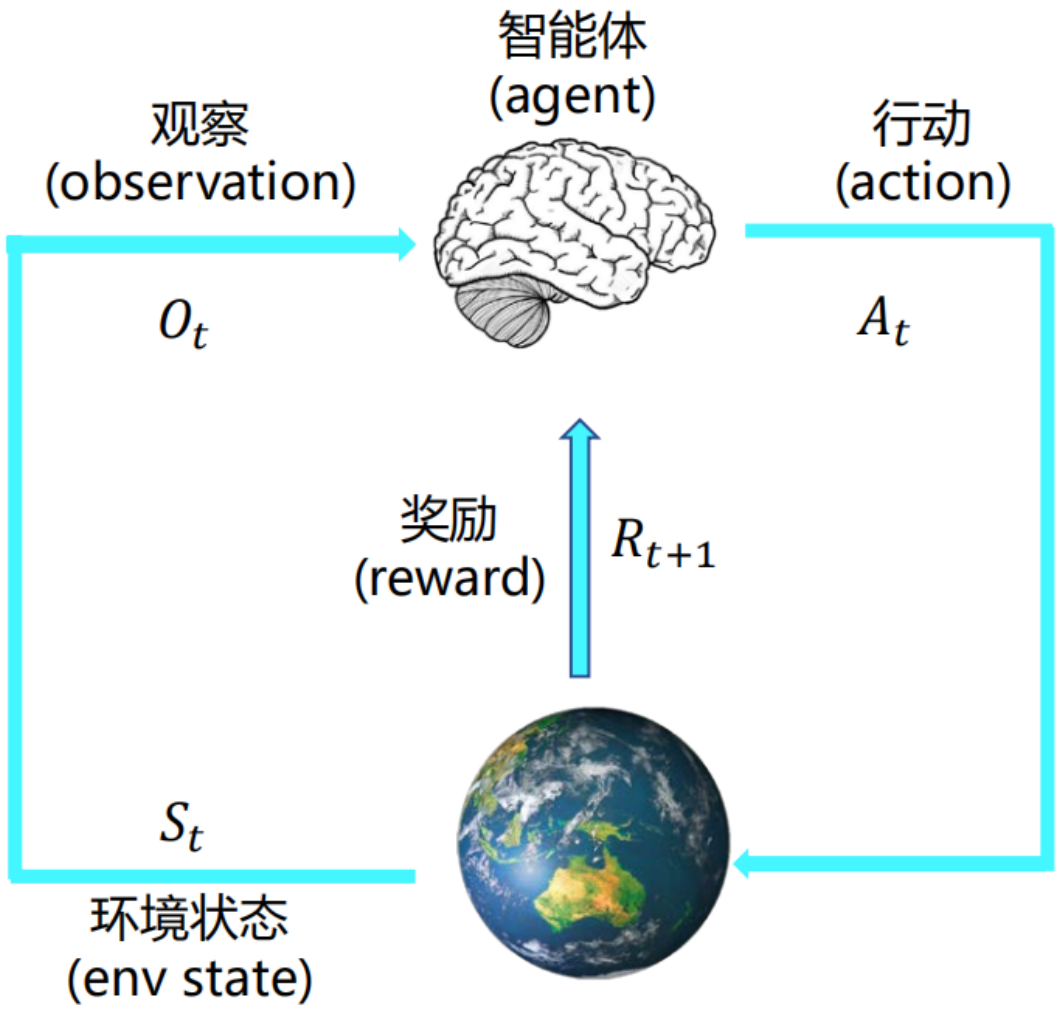
Adversarial Attacks on Neural Network Policies

S Huang 2017 https://arxiv.org/abs/1702.02284v1
截至5月 被引用量562

对强化学习智能体的攻击方法



攻击方式

- 基于**观测**的攻击
- 基于**奖励**的攻击
- 基于**环境**的攻击
- 基于**动作**的攻击
- 基于**策略**的攻击

分类	攻击方法	攻击模型	攻击策略	攻击阶段	对手知识
观测攻击(见2.1)	FGSM[19]	DQN[1-2]、TRPO[7]、A3C[6]	在观测上加上FGSM攻击	测试阶段	白盒/黑盒
	策略诱导攻击[41]	DQN[1-2]	训练敌手策略; 对抗样本的转移性	训练阶段	黑盒
	战略时间攻击[42]	DQN[1-2]、A3C[6]	在一些关键时间步进行攻击	测试阶段	白盒
	迷惑攻击[42]	DQN[1-2]、A3C[6]	通过预测模型诱导智能体做出动作	测试阶段	白盒
	基于值函数的对抗攻击[44]	A3C[6]	在值函数的指导下选择部分观测进行攻击	测试阶段	白盒
	嗅探攻击[45]	DQN[1-2]、PPO[39]	用观测以及奖励、动作信号来获取代理模型并进行攻击	测试阶段	黑盒
	基于模仿学习的攻击[46]	DQN[1-2]、A2C[6]、PPO[39]	使用模仿学习提取的专家模型信息进行攻击	测试阶段	黑盒
	CopyCAT算法[47]	DQN[1-2]	使用预先计算的掩码对智能体的观测做出实时的攻击	测试阶段	白盒/黑盒
奖励攻击(见2.2)	基于对抗变换网络的对抗攻击[21]	DQN[1-2]	加入一个前馈的对抗变换网络使策略追求对抗奖励	测试阶段	白盒
	木马攻击[48]	A2C[6]	在训练阶段用特洛伊木马进行中毒攻击	训练阶段	白盒/黑盒
	翻转奖励符号攻击[49]	DDQN[3]	翻转部分样本的奖励值符号	训练阶段	白盒
环境攻击(见2.3)	路径脆弱点攻击[50]	DQN[1-2]	根据路径点Q值的差异与直线的夹角找出脆弱点	训练阶段	白盒
	通用优势对抗样本生成方法[20]	A3C[6]	在梯度上升最快的横断面上添加障碍物	训练阶段	白盒
	对环境模型的攻击[51]	DQN[1-2]、DDPG[38]	在环境的动态模型上增加扰动	测试阶段	黑盒
动作攻击(见2.4)	动作空间扰动攻击[52]	PPO[39]、DDQN[3]	通过奖励函数计算动作空间扰动	训练阶段	白盒
策略攻击(见2.5)	通过策略进行攻击[53]	PPO[39]	采用对抗智能体防止目标智能体完成任务	测试阶段	黑盒

对强化学习智能体的防御方法

- 鲁棒性学习：

  通过训练，提高模型抵抗干扰的能力。

  - 干扰来源：自然噪声扰动、对抗样本扰动
  - 防御方法：降噪、对抗训练

- 对抗检测：

  正常训练模型，在模型预测过程中检测识别对抗样本。

  - 直接检测输入
  - 根据输入对模型的影响进行检测

论文内容概述

Huang的论文最先对深度强化学习的攻击进行了探索：

- 基于FGSM的白盒攻击
- 基于迁移性的黑盒攻击

此后，Anay 对FGSM的攻击方法进行了改进，并用于对抗训练：

- 利用梯度下降确定方向，再抽样确定最佳步长

FGSM : Fast Gradient Sign Method

FGSM focuses on adversarial perturbations where each pixel of the input image is changed by no more than ϵ. Given a linear function $g(x) = w^\top x$, the optimal adversarial perturbation η that satisfies $\|\eta\|_\infty < \epsilon$ is

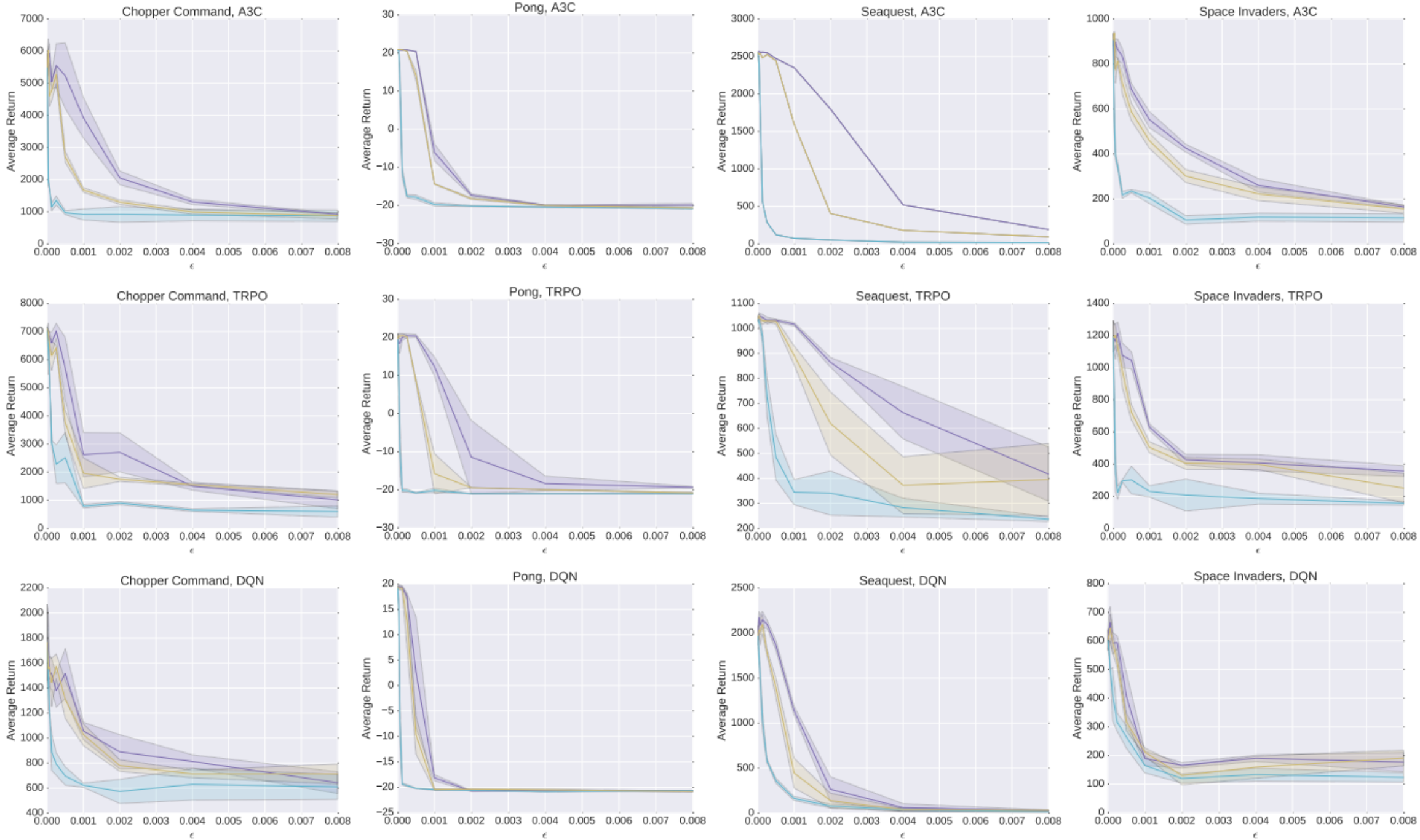
$$\eta = \epsilon \, \text{sign}(w), \tag{1}$$

since this perturbation maximizes the change in output for the adversarial example $\tilde{x}$, $g(\tilde{x}) = w^\top x + w^\top \eta$.

Given an image classification network with parameters θ and loss $J(\theta, x, y)$, where x is an image and y is a distribution over all possible class labels, linearizing the loss function around the input x results in a perturbation of

$$\eta = \epsilon \, \text{sign}(\nabla_x J(\theta, x, y)). \tag{2}$$

$$\eta = \begin{cases} \epsilon \, \text{sign}(\nabla_x J(\theta, x, y)) & \text{for constraint } \|\eta\|_\infty \leq \epsilon \\ \epsilon\sqrt{d} * \frac{\nabla_x J(\theta,x,y)}{\|\nabla_x J(\theta,x,y)\|_2} & \text{for constraint } \|\eta\|_2 \leq \|\epsilon \mathbf{1}_d\|_2 \\ \text{maximally perturb highest-impact dimensions with budget } \epsilon d & \\ & \text{for constraint } \|\eta\|_1 \leq \|\epsilon \mathbf{1}_d\|_1 \end{cases}$$


Chopper Command, A3C
Pong, A3C
Seaquest, A3C
Space Invaders, A3C
Chopper Command, TRPO
Pong, TRPO
Seaquest, TRPO
Space Invaders, TRPO
Chopper Command, DQN
Pong, DQN
Seaquest, DQN
Space Invaders, DQN
Average Return
ϵ

黑盒攻击场景细分

- Transferability Across Policies
  - 攻击者知道目标智能体所采用的算法，但不知道其训练的初始化参数，攻击者使用与目标相同的算法训练对抗攻击样本

- Transferability Across Training Algorithms
  - 攻击者对目标智能体一无所知，攻击者用各种算法训练对抗样本

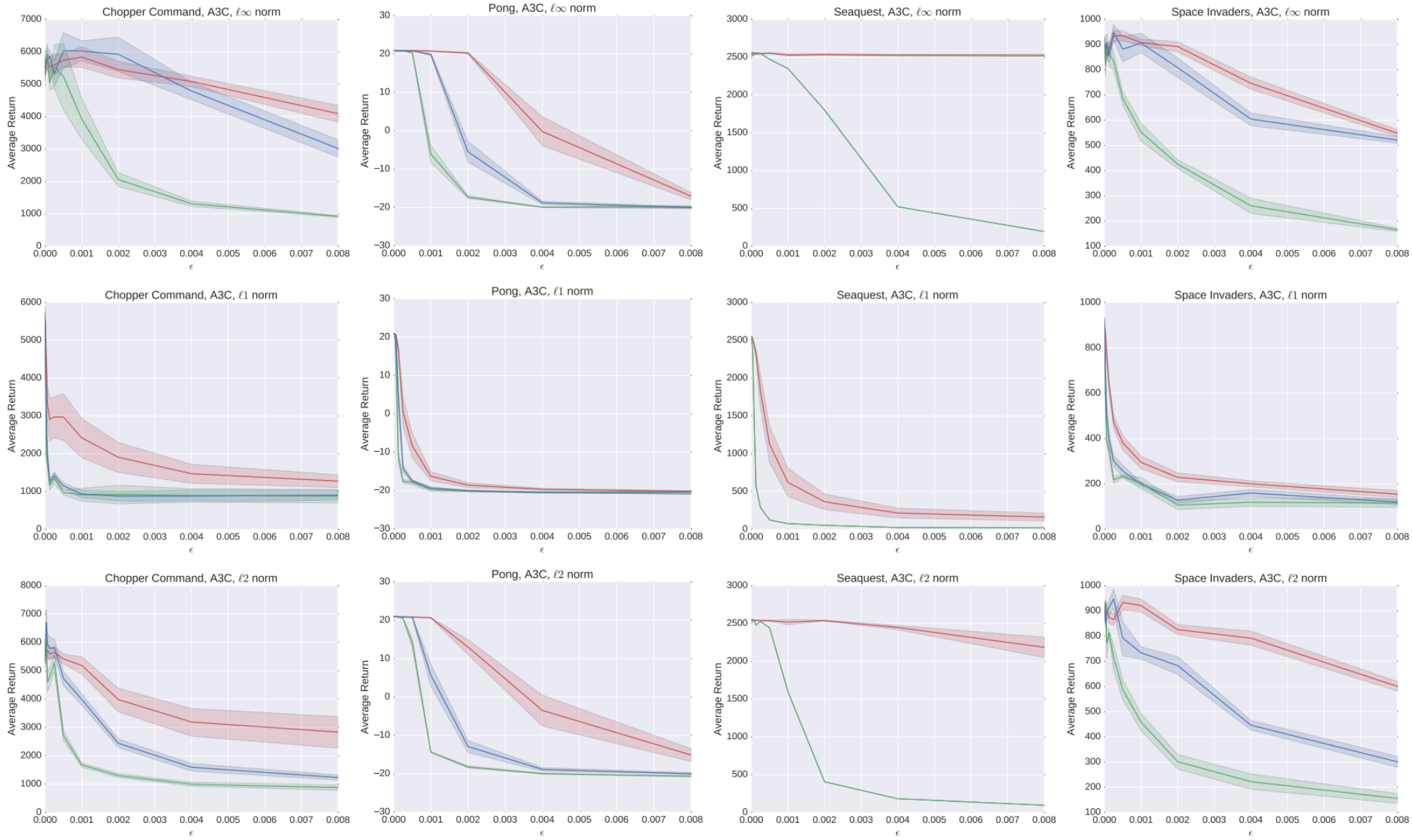


Figure 3: Transferability of adversarial inputs for policies trained with A3C. Type of transfer: algorithm policy none

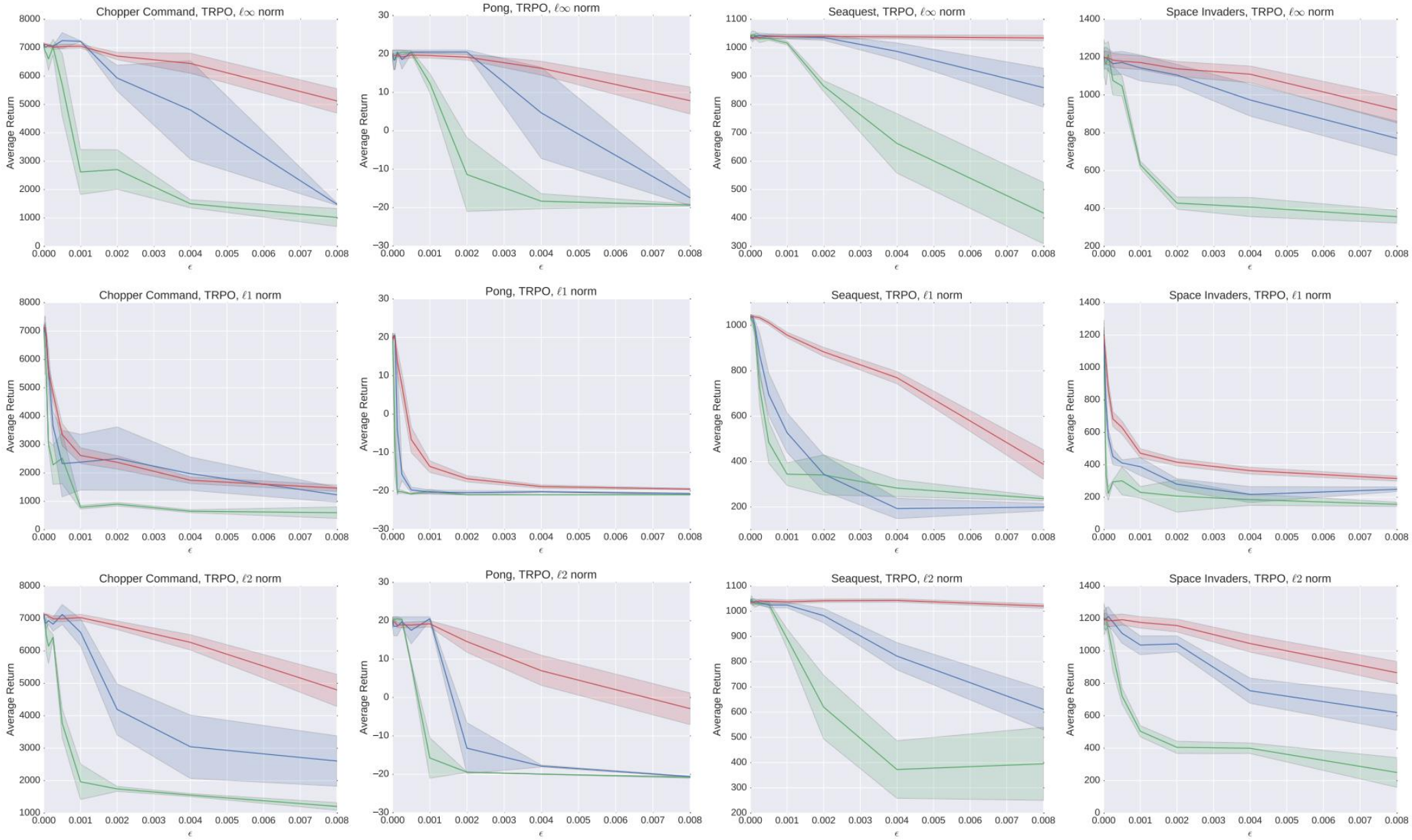


Figure 4: Transferability of adversarial inputs for policies trained with TRPO. Type of transfer: algorithm policy none

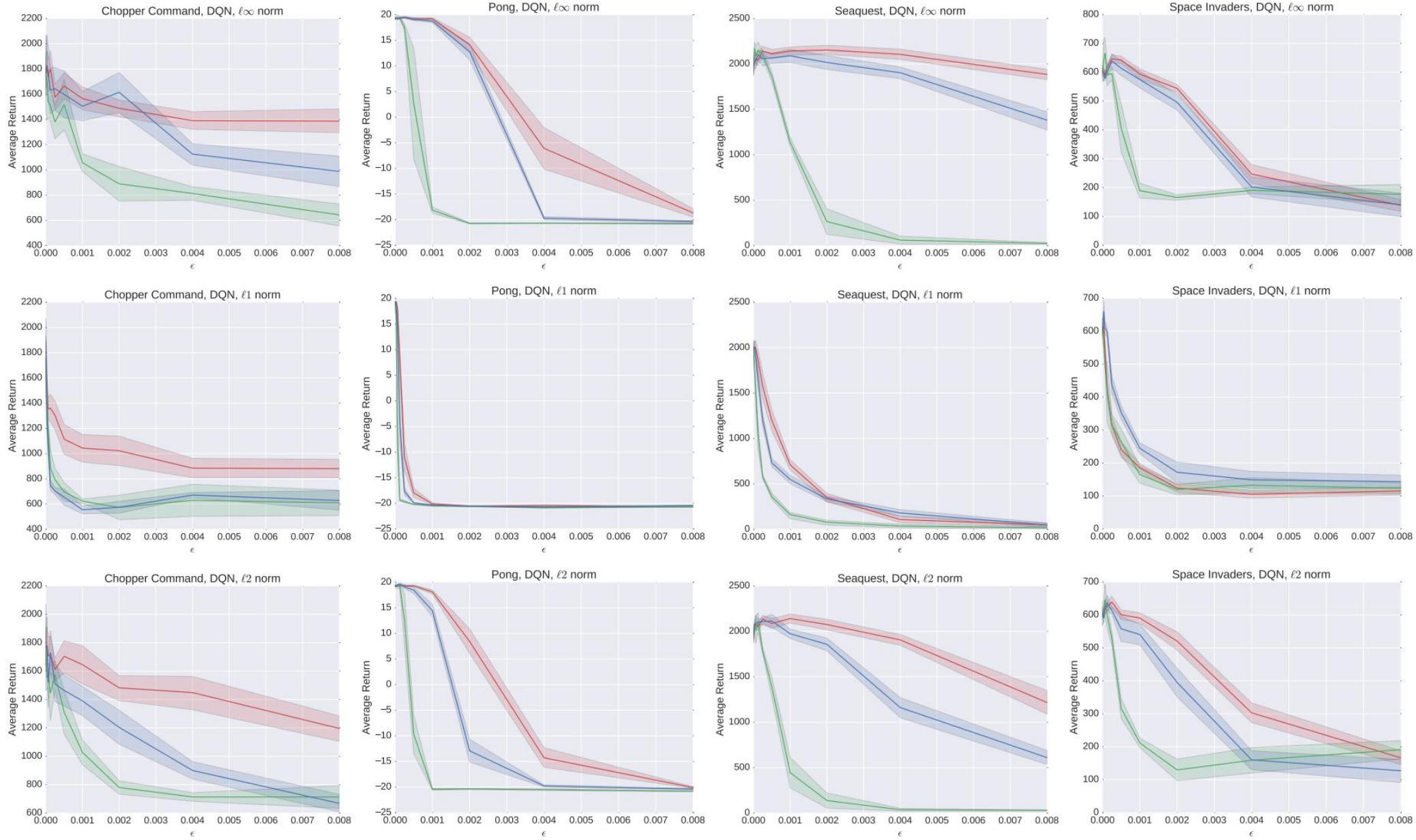


Figure 5: Transferability of adversarial inputs for policies trained with DQN. Type of transfer: algorithm policy none

对FGSM方法的改进

Algorithm 1 Naive attack (DDQN)

procedure NAIVE($Q^{target}, Q, s, \epsilon, n, \alpha, \beta$) ▷ Naive attack function takes Q network (Q), current state(s), adversarial attack magnitude constraint(ϵ), parameters of beta distribution(α, β) and number of times to sample noise(n) as input

$a^* = arg\max_a Q(s,a), Q^* = \max_a Q^{target}(s,a)$ ▷ Determine optimal action value function

for $i = 1 : n$ **do** ▷ Sample a few times

$n_i \sim beta(\alpha, \beta) - 0.5$ ▷ Sample noise

$s_i = s + \epsilon * n_i$ ▷ Possible adversarial state determined by sampled noise

$a_{adv} = arg\max_a Q(s_i, a)$ ▷ Determine optimal action in potential adversarial state

$Q^{target}_{adv} = Q^{target}(s, a_{adv})$ ▷ Determine the value of potential adversarial action corresponding to potential adversarial state for current state

if $Q^{target}_{adv} < Q^*$ **then** ▷ if the potential adversarial state leads to bad action

$Q^* = Q^{target}_{adv}$ ▷ Store the value function of that potential bad action

$s_{adv} = s_i$ ▷ Store possible adversarial state

else

do nothing

end if

end for

return s_{adv} ▷ Adversarial state

end procedure

Gradient based attack (DDQN)

$\pi^{target} = softmax(Q^{target})$

$grad = \nabla_s J(s, \pi^{target})$

$grad\_dir = \frac{\nabla_s J(s,\pi^{target})}{||\nabla_s J(s,\pi^{target})||}$

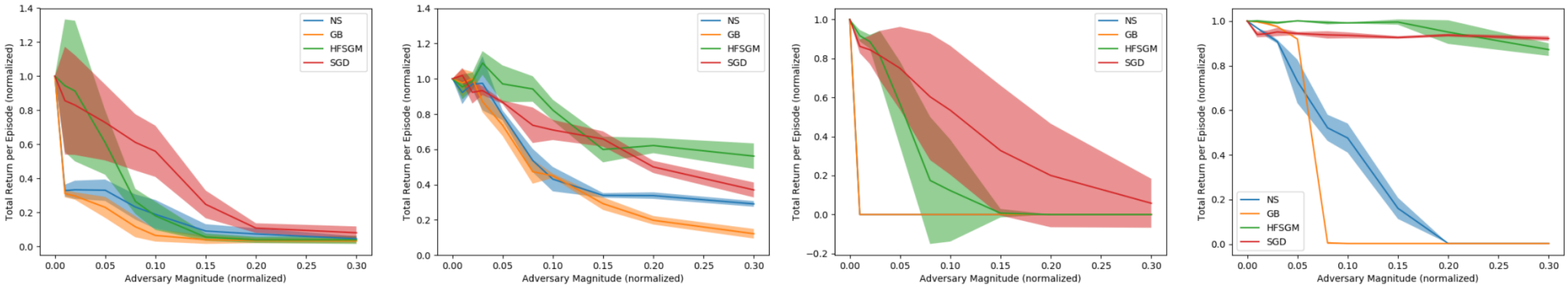
for $i = 1 : n$ **do**

$n_i \sim beta(\alpha, \beta)$

$s_i = s - n_i * grad\_dir$

$a_{adv} = arg\max_a Q(s_i, a)$

$Q^{target}_{adv} = Q^{target}(s, a_{adv})$



(a) DDQN Cart Pole (b) RBF Q Cart Pole (c) DDQN Mountain Car (d) RBF Q mountain car

Figure 2: Comparison of different attacks on DDQN and RBF based Q learning. Subfigure (a) and (b) shows adversarial attack on cart-pole environment with DDQN. Subfigures (c) and (d) show adversarial attack on mountain car environment. It can be observed that Gradient Based (GB) attack performs better than Naive Sampling (NS) which in turn outperform Stochastic Gradient Descent (SGD) as well as HFSGM (Huang et al. [2017]). RBF Q learning is relatively more resilient to adversarial attack than DDQN.

END