Malicious Attacks against Deep Reinforcement Learning Interpretations

Mengdi Huai[1], Jianhui Sun[1], Renqin Cai[1], Liuyi Yao[2], Aidong Zhang[1]
[1]University of Virginia, Charlottesville, VA, USA
[2]State University of New York at Buffalo, Buffalo, NY, USA
[1]{mh6ck, js9gu, rc7ne, aidong}@virginia.edu, [2]liuyiyao@buffalo.edu

ABSTRACT

The past years have witnessed the rapid development of deep reinforcement learning (DRL), which is a combination of deep learning and reinforcement learning (RL). However, the adoption of deep neural networks makes the decision-making process of DRL opaque and lacking transparency. Motivated by this, various interpretation methods for DRL have been proposed. However, those interpretation methods make an implicit assumption that they are performed in a reliable and secure environment. In practice, sequential agent-environment interactions expose the DRL algorithms and their corresponding downstream interpretations to extra adversarial risk. In spite of the prevalence of malicious attacks, there is no existing work studying the possibility and feasibility of malicious attacks against DRL interpretations. To bridge this gap, in this paper, we investigate the vulnerability of DRL interpretation methods. Specifically, we introduce the first study of the adversarial attacks against DRL interpretations, and propose an optimization framework based on which the optimal adversarial attack strategy can be derived. In addition, we study the vulnerability of DRL interpretation methods to the model poisoning attacks, and present an algorithmic framework to rigorously formulate the proposed model poisoning attack. Finally, we conduct both theoretical analysis and extensive experiments to validate the effectiveness of the proposed malicious attacks against DRL interpretations.

CCS CONCEPTS

• **Information systems → Data mining**; • **Computing methodologies → Machine learning**; • **Security and privacy**;

KEYWORDS

Deep reinforcement learning; model interpretation; adversarial attacks; poisoning attacks

ACM Reference Format:
Mengdi Huai, Jianhui Sun, Renqin Cai, Liuyi Yao, Aidong Zhang. 2020. Malicious Attacks against Deep Reinforcement Learning Interpretations. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3394486.3403089

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403089

1 INTRODUCTION

In recent years, there has been increasing interest in RL, a machine learning paradigm that has achieved great success in addressing challenging sequential decision-making problems [20]. The key components of RL include an agent and its environment, where the agent learns an optimal action selection policy by iteratively interacting with and receiving rewards from its environment. RL has served in a wide spectrum of applications, such as healthcare, autonomous navigation, and optimal control. In reality, many achievements of RL are due to its combination with deep learning. This combination, called DRL, is more capable of handling tasks with either high dimensional state space or complex task selection policy. Recently, various DRL algorithms have been developed, including deep Q-networks (DQN) [15], trust region policy optimization (TRPO) [18], and asynchronous advantage actor-critic (A3C) [5].

Although DRL techniques have shown superior performance in many real-world applications, their decision-making process is opaque and lacking transparency, which makes the inner workings of DRL models incomprehensible for human users. The "black box" nature of the DRL models may impede users from trusting the predicted results, especially when the model is used for making critical decisions (e.g., medical diagnosis and autonomous driving), because the consequences may be catastrophic if the predictions are acted upon blind faith. To address this problem, a plethora of DRL interpretation methods have been proposed to gain insight into the decision-making process of DRL agents [2, 5, 15, 21, 22, 24, 25]. These interpretation methods can provide explanations for particular predictions of DRL models and help humans understand the inner mechanisms. For example, at each time step of a sepsis patient's trajectory, doctors can use these interpretation methods to interpret the clinical decision making. However, an implicit assumption for these DRL interpretation methods is that they are performed in a reliable and secure environment, which may not be true in practice. As DRL interpretations play an increasingly critical role in many real-world applications, they are susceptible to a risk of being maliciously attacked. In this paper, we consider two representative types of attacks against DRL interpretations: *adversarial attack* and *model poisoning attack*.

Adversarial Attack. This type of attacks happens in the testing stage of DRL where the attacker tampers the input data after the victim DRL model is trained. Unlike deep supervised learning models, where decision is made instantaneously and independently, adversarial attacks to DRL models are extremely difficult to analyze quantitatively and defend effectively, as DRL models involve a temporally dependent sequential decision making process where states at different time-steps are perturbable. At each targeted victim time-step, the attacker's goal is to fool both the DRL model and the

corresponding DRL interpretation method through manipulating the current state observation that is communicated between the agent and the environment. For example, at a specific time step of a sepsis patient's trajectory, the attacker could add adversarial perturbation onto patient's clinical records, which not only causes the DRL model to produce wrong medical decision but also leads the adopted interpreter to give wrong interpretation results.

Model Poisoning Attack. Different from the adversarial attacks at the testing stage, the model poisoning attacks occur in the training stage of DRL. In this type of attacks, the attacker aims to dramatically degrade the performance of the DRL interpretation methods through manipulating the learned DRL model parameters, while maintaining the original performance of DRL model to ensure maximal stealthiness. This type of attacks is common in many real-world applications. Consider the example where due to the vast computational cost of training DRL models, the agent resorts to downloading the well-trained DRL model from an online model repository to complete its own tasks. However, during this process, the attacker could perform the model poisoning attack to manipulate the pre-trained DRL model to make the agent unwittingly download a maliciously re-trained DRL model.

Despite the prevalence of malicious attacks in real-world applications, there is no existing work studying the vulnerability of DRL interpretations to these attacks. Although there are some works addressing the adversarial vulnerability of the interpretation methods for supervised deep neural networks (DNN) based classification models [1, 4, 11, 23, 26], they cannot be directly applied to DRL interpretations due to the following unique features of DRL: First of all, they only focus on attacking a particular test instance in supervised classification settings, and ignore the agent's sequential decision-making process that consists of a continuous sequence of state-action predictions. If we directly adopt these per-instance attack methods and craft different perturbations to different states, the computation complexity will be extremely high. Secondly, a human imperceptible per-instance attack as in supervised deep learning models, may be highly noticeable for DRL agent as a tiny perturbation in one certain state may have unpredictable and apparent shift towards the whole future path. DRL is in fact goal-oriented, and it aims to learn sequences of actions that can lead the agent to achieve its goal. For example, in the autonomous driving scenarios, the ultimate goal of the agent is to successfully and safely reach the desired destination. If an attacker locally perturbs some states with per-instance perturbation methods without a global grasp of agent's end goal, the agent can easily detect the attack based on the deviation away from his desirable destination. Thus, the attacker who aims to attack the DRL models should be more cautious because he needs not only to guarantee the imperceptibility of local perturbations, but also to avoid compromising the end goal of the agent. Last but not least, an implicit assumption in the existing adversarial attack methods is that the attacker has the capability of manipulating the whole input data (e.g., the entire image). However, in practice, the attacker may be restricted to only manipulating a subset of each input data point (e.g., the bottom right region where digital watermark is), and hence this assumption may be impractical for the real-world physical attacks.

As for the model poisoning attacks, to the best of our knowledge, there is no existing work studying the vulnerability of interpretation models to such attacks. The challenge here is how to manipulate the pre-trained DRL model such that the attacker can dramatically alter the interpretation results without significantly hurting the performance of the DRL model. If the performance of the DRL model is significantly degraded after the manipulation, the model poisoning attack can be easily detected by evaluating on a holdout set, and then the manipulated DRL model will be immediately rejected by the agent.

To well understand the performance of DRL interpretations in malicious environment, in this paper, we study their vulnerability to the above two types of attacks. Specifically, we first propose an universal adversarial attack against DRL interpretations (**UADRLI**), based on which the attacker can add the crafted universal perturbation to the environment states on a maximum number of time steps while incurring minimal damage to the agent's end goal. In our design, the optimal attack strategy can be efficiently derived by solving an optimization problem even with the sequential and progressive nature of DRL taken into account. Additionally, we propose a model poisoning attack against DRL interpretations (**MPDRLI**), based on which the attacker can manipulate the pre-trained DRL model such that the attacker can dramatically alter the interpretation results without significantly hurting the efficacy of the original DRL model. With the proposed model poisoning manipulation, the interpretations can be successfully misled in multiple experimental settings. We also provide theoretical results indicating the change in overall efficacy of DRL model is strictly bounded, which guarantees the difficulty to detect such attacks. To summarize, our contributions are:

- First of all, we propose an universal adversarial attack against DRL interpretations (i.e., UADRLI), which aims to craft a single universal perturbation that can be applied identically (uniformly) on every time step. Based on the proposed UADRLI, the attacker can efficiently deceive downstream DRL interpretation methods via state perturbations.
- We also design a model poisoning attack against DRL interpretations (MPDRLI), based on which the attacker can secretly alter the interpretation results through providing the agent a strategically poisoned but equally effective pre-trained DRL model.
- Both theoretical analysis and extensive experimental results validate the effectiveness of the proposed malicious attacks against DRL interpretations.

2 PRELIMINARY

Deep Reinforcement Learning. In RL, an agent aims to learn an optimal behavior through trial-and-error by sequentially interacting with an environment, which is referred to as the Markov Decision Process (MDP) defined with a tuple $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$. In a MDP, the agent interacts with its environment in the following way: The agent starts by gathering an initial state $s_0 \in \mathcal{S}$ that describes the environment. At each time step t, the agent chooses an action $a_t \in \mathcal{A}$ according to some policy π based on the current state $s_t \in \mathcal{S}$ (i.e., $a_t = \arg\max_{a \in \mathcal{A}} \pi(s_t, a)$), and progresses to a new state $s_{t+1} \in \mathcal{S}$ according to the transition dynamics $\mathcal{P}$. Additionally, the agent receives a scalar reward $r(s_t, a_t) = \mathcal{R}(s_t, a_t)$, provided by the reward function $\mathcal{R}$, which judges the quality of

its decision. This sequential decision making process produces a sequence of state-action pairs $\mathbb{T} = \{(s_t, a_t)\}_{t=0}^{T}$, where T is the timestep that the environment terminates. The return is computed as $R = \sum_{t=0}^{T} \gamma^t r(\boldsymbol{s}_t, a_t)$, where $\gamma \in [0, 1]$ is a discount factor indicating how much the agent values an immediate reward compared to a future reward. The agent's goal is to find a policy π^* that maximizes the expected value of the total reward from all states

$$\pi^* = \arg\max_{\pi} \{\mathbb{E}[\sum_{t=0}^{T} \gamma^t r(\boldsymbol{s}_t, a_t)]\}, \quad (1)$$

where $\mathbb{E}$ denotes the expectation over all possible trajectories generated by policy π. DRL is the combination of deep learning and RL, and is proposed to overcome the challenges in learning control policies from high-dimensional raw input data and large state and action spaces in traditional RL environments. In DRL, we represent the policy π with a deep neural network that is parameterized by $\boldsymbol{\Theta}$ (i.e., the weights of the policy network). To find out the optimal policy parameters, many different DRL algorithms have been proposed, including DQN [15], TRPO [18], and A3C [5].

Interpretation Methods for Deep Reinforcement Learning. Currently, many works have been proposed to make DRL more transparent. In general, existing interpretation methods for DRL can be grouped into two categories: intrinsic interpretability and post-hoc interpretability. The latter case does not require modifying model architectures or parameters, thereby leading to higher prediction accuracy. In this paper, we mainly consider post-hoc interpretations. Formally, for the given state-action pair $(\boldsymbol{s}_t, a_t)$ and policy π, the post-hoc interpreter $\mathcal{I}$ can generate the feature importance scores $\mathcal{I}(\boldsymbol{s}_t, a_t; \pi)$, which measures how important each feature of the state $\boldsymbol{s}_t$ is, in determining the corresponding action a_t under policy π. In the following, we describe several widely-used post-hoc interpretation methods for DRL, all of which aim to generate feature importance scores (also called saliency maps) to show the relevancy of each feature for the prediction.

- **Gradient saliency.** This method [5] is a generic interpretation method that combines gradient information with class activation maps to visualize the importance of each feature. The map is computed as $\mathcal{I}(\boldsymbol{s}_t, a_t; \pi) = \frac{\partial \pi(\boldsymbol{s}_t, a_t)}{\partial \boldsymbol{s}_t}$, and quantifies how sensitive the action prediction score (i.e., $\pi(\boldsymbol{s}_t, a_t)$) is with respect to the small changes of input features.
- **Jacobian saliency.** Wang et al. [21] extend gradient-based saliency maps to DRL by computing the Jacobian of the output logits with respect to a stack of input images. Specifically, to visualize the salient part of the image as seen by the value stream, they compute the absolute value of the Jacobian of the predicted state value with respect to the input frame.
- **Object saliency.** Iyer et al. [9] use template matching, a common computer vision technique, to detect objects within an input image and measure saliency through changes in Q-values for masked and unmasked objects.
- **Perturbation saliency.** Greydanus et al. [5] use saliency maps to provide explanations for the DRL agent's behaviors over temporally extended sequences. Specifically, they generate saliency maps by perturbing the original input image using a Gaussian blur of the image and measure changes in policy from removing information from a region.

Note that all of the above interpretation methods for DRL focus on instance-level interpretability, which means the different interpretation results would be given on different state-action pairs in one episode, and provide the importance score of each feature.

3 ADVERSARIAL ATTACK AGAINST DRL INTERPRETATIONS

In this section, we first introduce the threat model and then develop an optimization framework to formalize our universal adversarial attack against DRL interpretations (i.e., UADRLI). After that, we present the theoretical analysis for the proposed universal attack.

3.1 Threat Model

Following the line of work on adversarial attacks [1, 4, 11, 23, 26], we here assume a white-box setting, which is a conservative and realistic assumption. The attacker in this setting tries to evade the system by manipulating malicious states during the testing phase. The attacker cannot change the DRL algorithm used for the training of the agent, and cannot change the architecture of the policy networks. The attacker can only change the state observations that are communicated between the agent and the environment. The attacker's goal is to deceive both the trained DRL model and its adopted interpretation method.

3.2 Formalization of Universal Adversarial Attack

In the adversarial attack settings, when we design the adversarial attack against DRL interpretations, we need to take the unique characteristics of DRL into account. Firstly, the attacker should handle the sequentiality of DRL, and this sequential decision-making process produces a sequence of state-action pairs $\{(\boldsymbol{s}_t, a_t)\}_{t=0}^{T}$. If we directly adopt existing methods on supervised learning attacks and craft state-dependent adversarial perturbations, the computation complexity will be significantly increased due to the generation of a large amount of perturbations different with each other. Additionally, when generating state-dependent perturbations, the attacker has to query the victim DRL model at test time. Instead, we propose to add the state-agnostic perturbations. Specifically, we aim to craft a single universal adversarial perturbation (dubbed as $\boldsymbol{\delta}$), which can be identically (uniformly) applied on every observed state without accessing the target victim DRL model at test time. We further restrict the attacker to only manipulating pixels within a small region of the image-the attacker may choose the location of the area, but cannot perturb pixels outside this selected image area. Formally, for any given observed state $\boldsymbol{s}_t$, the attacker crafts the corresponding adversarial state $\tilde{\boldsymbol{s}}_t$ as follows

$$\tilde{\boldsymbol{s}}_t = A(\boldsymbol{s}_t, k_t * \boldsymbol{M}, \boldsymbol{\delta}) = (1 - k_t * \boldsymbol{M}) \odot \boldsymbol{s}_t + (k_t * \boldsymbol{M}) \odot \boldsymbol{\delta}, \quad (2)$$

where $k_t \in \{0, 1\}$, $\odot$ denotes the matrix element-wise product, and $\boldsymbol{\delta}$ is the universal perturbation to be generated. Here, $\boldsymbol{M}$ is a predefined binary mask matrix representing the position and shape of the area that can be attacked. Note that $k_t \in \{0, 1\}$ denotes whether at time step t the perturbation should be applied ($k_t = 1$) or not ($k_t = 0$). Specifically, if $k_t = 1$, $A(\boldsymbol{s}_t, k_t\boldsymbol{M}, \boldsymbol{\delta}) = (1-\boldsymbol{M})\odot\boldsymbol{s}_t+\boldsymbol{M}\odot\boldsymbol{\delta}$. Otherwise, $A(\boldsymbol{s}_t, k_t\boldsymbol{M}, \boldsymbol{\delta}) = (1 - (0 * \boldsymbol{M})) \odot \boldsymbol{s}_t + (0 * \boldsymbol{M}) \odot \boldsymbol{\delta} = 1 \odot \boldsymbol{s}_t = \boldsymbol{s}_t$. Secondly, we should make sure that the universal

adversarial perturbation (i.e., $\boldsymbol{\delta}$) that causes the wrong predictions is imperceptible. We use parameter ϵ to control the magnitude of the universal perturbation $\boldsymbol{\delta}$. Specifically, given a state $\boldsymbol{s}_t$ of the system, the attacker can only select a perturbed state $\tilde{\boldsymbol{s}}_t$ as

$$\tilde{\boldsymbol{s}}_t \in \{\tilde{\boldsymbol{s}}_t \in \mathcal{S} : d(\boldsymbol{s}_t, \tilde{\boldsymbol{s}}_t) \leq \epsilon\}, \tag{3}$$

where $d(\boldsymbol{s}_t, \tilde{\boldsymbol{s}}_t) = \|\boldsymbol{s}_t - \tilde{\boldsymbol{s}}_t\|_\infty = \|\boldsymbol{\delta}\|_\infty$. Thirdly, when added to any clean state $\boldsymbol{s}_t$, the universal adversarial perturbation will cause the policy to select a different action at this state and the interpretation results to be wrong at the same time. Note that the attacked image region is the real reason why the policy alters decision. Hence, the attacker should mislead the interpreter to highlight other image regions that are not perturbed. In addition to fooling the interpreter for DRL, when adding $\boldsymbol{\delta}$ to state $\boldsymbol{s}_t$ at time step t, the attacker also wants to mislead the agent to take any wrong action (instead the original optimal action), that is,

$$\arg\max_{a \in \mathcal{A}} \pi(\boldsymbol{s}_t, a) \neq \arg\max_{a \in \mathcal{A}} \pi(\tilde{\boldsymbol{s}}_t, a), \tag{4}$$

where $\tilde{\boldsymbol{s}}_t$ is calculated based on Eq. (2). Last but not least, since DRL is goal-oriented, the attack will be easily detected if the attacker perturbs the observed state at every time step. The reason is that when the attacker perturbs every observed state, the final accumulated reward will be largely compromised. On the attacker's side, he also wants to maximize his expected utility. In this paper, we consider the case where the attacker wants to maximize the total number of attacked time steps (i.e., $\sum_{t=0}^{T} k_t$). On the other hand, to avoid being detected, the attacker should also make sure that the end goal of the entire DRL task is not significantly compromised. In practice, the end goal of the agent is formalized in terms of the accumulated reward in the long run. Specifically, the attacker should make sure that the difference of the accumulated reward before and after the attack should be small, and the difference is defined as $(\sum_{t=0}^{T} \mathbb{E}_\pi[\gamma^t r(\tilde{\boldsymbol{s}}_t, \tilde{a}_t)] - \sum_{t=0}^{T} \mathbb{E}_\pi[\gamma^t r(\boldsymbol{s}_t, a_t)])^2$, where $\tilde{a}_t = \arg\max_{a \in \mathcal{A}} \pi(\tilde{\boldsymbol{s}}_t, a)$. Now, the problem is how can the attacker find an effective attack strategy $\{k_0, \cdots, k_T\}$ with the corresponding single universal perturbation $\boldsymbol{\delta}$, which not only maximizes the attacker's utility (i.e., $\sum_{t=0}^{T} k_t$) but also incurs minimal damage to the agent's end goal.

Based on the above arguments, the attacker's goal is to add *smallest (imperceptible)* universal perturbation to the environment states in a *maximum* number of steps while incurring *minimal* damage to the agent's end goal. With this goal in mind, for the given episode that consists a sequence of state-action pairs $\{(\boldsymbol{s}_t, a_t)\}_{t=0}^{T}$ and the threat model, at high level, we formulate the proposed adversarial attack using the following optimization framework

$$\begin{aligned}
\min_{\boldsymbol{\delta}, \{k_t \in \{0,1\}\}_{t=0}^{T}} & (\sum_{t=0}^{T} \mathbb{E}_\pi[\gamma^t r(\tilde{\boldsymbol{s}}_t, \tilde{a}_t)] - \sum_{t=0}^{T} \mathbb{E}_\pi[\gamma^t r(\boldsymbol{s}_t, a_t)])^2 \\
& + \lambda_1 \sum_{t=0}^{T} \exp(\mathcal{I}(\tilde{\boldsymbol{s}}_t, \tilde{a}_t; \pi) \odot \boldsymbol{M}) - \lambda_2 \sum_{t=0}^{T} k_t \\
s.t. \quad & \forall t \in [T], \tilde{\boldsymbol{s}}_t = A(\boldsymbol{s}_t, k_t * \boldsymbol{M}, \boldsymbol{\delta}) \\
& \forall t \in [T], \tilde{a}_t = \arg\max_{a \in \mathcal{A}} \pi(\tilde{\boldsymbol{s}}_t, a), \\
& \|\boldsymbol{\delta}\|_\infty \leq \epsilon, \\
& \forall t \in \mathcal{T}_1, \arg\max_{a \in \mathcal{A}} \pi(\boldsymbol{s}_t, a) \neq \arg\max_{a \in \mathcal{A}} \pi(\tilde{\boldsymbol{s}}_t, a),
\end{aligned} \tag{5}$$

where $\mathcal{T}_1 = \{t : k_t = 1\}$ denotes the set of time steps being attacked, and λ_1 and λ_2 are two regularization parameters. The first loss term is utilized to enforce that the actual ultimate accumulated reward does not change significantly. The second loss term is used to decrease the importance scores of the pixels that are within the attacked image region. The third loss term is introduced to maximize the attacker's utility (i.e., the number of attacked time steps). The two hyper-parameters (i.e., λ_1 and λ_2) balance the three factors. The third constraint allows the attacker to manipulate all the states that the agent perceives within an ϵ budget, and hence ensures that the universal perturbation (i.e., $\boldsymbol{\delta}$) is imperceptible. The last constraint forces that the action predictions are wrong.

3.3 Optimization

In this section, we discuss how to solve the optimization problem described in Eq. (6). However, it is very challenging to directly solve the above optimization problem as it involves too many variables. To address this challenge, we propose to convert it into two sub-problems, and then solve them in two separate steps. Specifically, in the first step, we aim to figure out the universal adversarial perturbation $\boldsymbol{\delta}$ by solving a sub-optimization problem. In the second step, we solve the problem of identifying the optimal attack strategy $\{k_0, \cdots, k_T\}$. Below, we elaborate the two steps in greater detail.

Step 1: Generating the Universal Adversarial Perturbation. In this step, we focus on how to generate the universal adversarial perturbation (i.e., $\boldsymbol{\delta}$), which can be applied identically on every time step. As aforementioned, when generating the universal adversarial perturbation $\boldsymbol{\delta}$, the attacker should satisfy the following requirements: Firstly, when applying the universal adversarial perturbation (i.e., $\boldsymbol{\delta}$) to each observed state, the attacker should make sure that not only the predicted action is altered but also the corresponding interpretations are wrong. Additionally, we should note that the attacker is restricted to only manipulating pixels within a small region of the input image. Based on the above two restrictions, for the given unattacked trajectory $\{(\boldsymbol{s}_t, a_t)\}_{t=0}^{T}$, we formulate the following optimization problem

$$\begin{aligned}
\min_{\boldsymbol{\delta}} & \sum_{t=0}^{T} \frac{1}{T+1} \pi(\tilde{\boldsymbol{s}}_t, a_t) + \lambda_1 \sum_{t=0}^{T} \frac{1}{T+1} \exp(\mathcal{I}(\tilde{\boldsymbol{s}}_t, \tilde{a}_t; \pi) \odot \boldsymbol{M}) \\
s.t. \quad & \forall t \in [T], \tilde{\boldsymbol{s}}_t = A(\boldsymbol{s}_t, \boldsymbol{M}, \boldsymbol{\delta}), \\
& \forall t \in [T], \tilde{a}_t = \arg\max_{a \in \mathcal{A}} \pi(\tilde{\boldsymbol{s}}_t, a), \\
& \|\boldsymbol{\delta}\|_\infty \leq \epsilon,
\end{aligned} \tag{6}$$

where $A(\boldsymbol{s}_t, \boldsymbol{M}, \boldsymbol{\delta}) = (1 - \boldsymbol{M}) \odot \boldsymbol{s}_t + \boldsymbol{M} \odot \boldsymbol{\delta}$. The first loss term is used to force the agent to take an arbitrary action (instead of the original optimal action). The second term is used to fool the interpretation results. The first constraint enforces the attacker to only manipulate a small region of the input data. The last constraint aims to find the sufficiently imperceptible universal perturbation that leads to the wrong action prediction and interpretations desired by the adversary. To derive the universal perturbation $\boldsymbol{\delta}$, we can solve the above optimization problem using the projected gradient algorithm.

Step 2: Identifying Attack Points. Note that after Step 1, the attacker can generate the universal perturbation $\boldsymbol{\delta}$ that can be identically applied to every time step. However, as aforementioned, if the attacker perturbs the observed state at every time step, the

launched universal attack will be easily detected due to the significant decrease in the end reward. Hence, in this step, we discuss how to identify the optimal attack strategy $\{k_0, \cdots, k_T\}$ by strategically selecting a set of time steps. With this identified attack strategy, the attacker can maximize his attack utility while avoiding being detected. To derive the attack strategy, at each time step, the attacker first computes the variance of the Q value as follows

$$Var(Q(\boldsymbol{s}_t)) = \frac{1}{|\mathcal{A}|-1}\sum_{i=1}^{|\mathcal{A}|}(Q(\boldsymbol{s}_t, a_i) - \frac{1}{|\mathcal{A}|}\sum_{j=1}^{|\mathcal{A}|}Q(\boldsymbol{s}_t, a_j))^2, \quad (7)$$

where $\mathcal{A}$ is the action space of the MDP, and $|\mathcal{A}|$ is the number of actions. Then, according to the above calculated variance, the attacker decides whether he should perturb $\boldsymbol{s}_t$. Based on Lemma 3 (in Appendix), we know that when attacking states with low variance, the attacker will get more reward in expectation. Hence, to avoid being detected, the attacker should attack the states with low variance to incur low decrease in the accumulated reward.

3.4 Theoretical Analysis

In this section, we theoretically quantify the influence of our proposed universal attack on the accumulated reward collected by the agent throughout the game. To do so, we first characterize the environment under attack as a new MDP[1] (denoted as $\mathcal{M}_1$ or $\mathcal{M}_2$, depending on detailed attack setup), which is different from the original MDP $\mathcal{M}$ in its transition probability and immediate reward function. Then we have the following theorem.

Theorem 1. *Let V^*, V_1^*, and V_2^* be optimal value functions for $\mathcal{M}$, $\mathcal{M}_1$, and $\mathcal{M}_2$, respectively. Note that the value function is equal to the expected total reward for an agent starting from a particular state. Suppose $\mathcal{M}$, $\mathcal{M}_1$, and $\mathcal{M}_2$ have bounded immediate rewards, i.e., $\max_{s\in\mathcal{S}, a\in\mathcal{A}}|\mathcal{R}(\boldsymbol{s}, a)| \le R$. Let $TV(\mathcal{P}, \mathcal{Q})$ be the total variation distance between two probability measures $\mathcal{P}$ and $\mathcal{Q}$ on $\mathcal{S}$. Let $\|f - g\|_\infty = \max_{s\in\mathcal{S}, a\in\mathcal{A}}|f(\boldsymbol{s}, a) - g(\boldsymbol{s}, a)|$. Suppose $\mathcal{M}$ has transition and immediate reward models which are continuous on $\mathcal{S}$, i.e., $\forall \boldsymbol{s} \in \mathcal{S}, a \in \mathcal{A}$, $\|\mathcal{P}(\cdot|\boldsymbol{s}, a) - \mathcal{P}_1(\cdot|\boldsymbol{s}, a))\|_1 \le L\epsilon$, and $|\mathcal{R}(\boldsymbol{s}, a) - \mathcal{R}_2(\boldsymbol{s}, a)| \le l\epsilon$, for some constant L and l. Then, we have*

$$\begin{aligned}
&\|V^* - V_1^*\|_\infty \\
&\quad\le \frac{2\gamma R}{(1-\gamma)^2}\max_{s\in\mathcal{S}, a\in\mathcal{A}} TV(\mathcal{P}(\cdot|\boldsymbol{s}, a), \mathcal{P}_1(\cdot|\boldsymbol{s}, a)) \le \frac{\gamma RL}{(1-\gamma)^2}\epsilon, \\
&\|V^* - V_2^*\|_\infty \\
&\quad\le \frac{2\gamma R\max_{s\in\mathcal{S}, a\in\mathcal{A}} TV(\mathcal{P}(\cdot|\boldsymbol{s}, a), \mathcal{P}_2(\cdot|\boldsymbol{s}, a)) + (1-\gamma)\|\mathcal{R} - \mathcal{R}_2\|_\infty}{(1-\gamma)^2} \\
&\quad\le \frac{\gamma RL + (1-\gamma)l}{(1-\gamma)^2}\epsilon. \qquad (8)
\end{aligned}$$

The above theorem upper bounds the change in the optimal total accumulated reward if all time steps in the episode of game playing are perturbed. Note that in our proposed adversarial attack paradigm, we control the total number of attacked time steps in one episode. Therefore, the result from Theorem 1 on value function, which is essentially the sum of discounted rewards from all time steps, is insufficient. Accordingly, we define $\mathcal{T}$-step value function (in Appendix) to formally measure the influence of the attacker on the accumulated reward collected only from the perturbed $\mathcal{T}$ steps. Then, based on this, we can derive the following theorem.

Theorem 2. *Suppose $\mathcal{M}$, $\mathcal{M}_1$, and $\mathcal{M}_2$ satisfy the same assumptions as in Theorem 1. V^*, V_1^*, and V_2^* are optimal value functions for $\mathcal{M}$, $\mathcal{M}_1$, and $\mathcal{M}_2$, respectively. π^*, π_1^*, and π_2^* are optimal policies for $\mathcal{M}$, $\mathcal{M}_1$, and $\mathcal{M}_2$, respectively. Denote $\max(L\epsilon, l\epsilon)$ to be δ. Suppose π^* is executed on $\mathcal{M}_i$ for $\mathcal{T}$ steps, we denote $\mathcal{T}$-step value function under policy π^* as $\mathcal{V}_i^{\pi^*}(\boldsymbol{s}, \mathcal{T}) = \mathbb{E}[\sum_{t=0}^{\mathcal{T}-1}\gamma^t\mathcal{R}_i(\boldsymbol{s}_t, \pi^*(\boldsymbol{s}_t))|\boldsymbol{s}_0 = \boldsymbol{s}]$. We denote the maximum possible $\mathcal{T}$-step return by $\mathcal{G}_\mathcal{T} = \max_{s\in\mathcal{S}}\mathcal{V}^{\pi^*}(\boldsymbol{s}, \mathcal{T})$. For all $\omega \ge 0$, if $\epsilon \le \frac{1}{\max(L,l)}(\frac{\omega}{12|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T}})^2$, we have $|\mathcal{V}^{\pi^*}(\boldsymbol{s}, \mathcal{T}) - \mathcal{V}_i^{\pi^*}(\boldsymbol{s}, \mathcal{T})| \le \omega$, where $i \in \{1, 2\}$.*

From Theorem 2, we can see that when the agent unconsciously executes the original optimal policy in the adversarially perturbed environment $\mathcal{M}_1/\mathcal{M}_2$, since the agent is not allowed to re-train the policy network, the fluctuation measured by $\mathcal{T}$-step value function the agent may experience, is well bounded if the perturbation the attacker imposes is small, which validates our previous argument on stealthiness of our proposed attack.

4 MODEL POISONING ATTACK AGAINST DRL INTERPRETATIONS

In this section, we firstly describe the threat model considered in the poisoning attack settings. Then, we present an algorithmic framework to rigorously design a model poisoning attack against DRL interpretations.

4.1 Threat Model

Here, we describe the threat model considered in our poisoning attack. Different from traditional data poisoning attacks where the attacker injects fake samples into the training dataset before the training process of the victim model begins, we do not assume that the attacker has full knowledge of the original training data. Instead, we assume that the attacker in our setting only has access to the pre-trained DRL model. The attacker's goal is to manipulate the parameters of the pre-trained DRL model such that the interpretation results are significantly altered. Consider the aforementioned example where due to the limited computational resources, the agent resorts to downloading the pre-trained DRL model provided by a third party. During this process, the attacker could pose potential threats to the system. To contrast with traditional data poisoning attacks [13, 14], we here call our attack the model poisoning attack since the attacker directly manipulates the pre-trained model parameters that are generated in the training phase instead of the training dataset. In practice, model poisoning attacks are more ubiquitous than data poisoning attacks as the attacker may not always have access to the training database that is typically secured by professional staff, while ordinary agent may acquire poisoned model from the attacker disguised as third-party model provider.

4.2 Formalization of Model Poisoning Attack

In this section, we introduce our designed model poisoning attack against DRL interpretations (i.e., MPDRLI). Note that in our model poisoning attack setting, the attacker directly manipulates the model parameters instead of the training data. However, if the

[1]Due to space constraints, both the formal definitions of $\mathcal{M}_1$ and $\mathcal{M}_2$, and the proof of the theorems are deferred to Appendix.

performance of the re-trained DRL model is significantly degraded, the model poisoning attack can be easily detected by evaluating on the validation games, and then the re-trained DRL model will be immediately rejected by the agent. Hence, when the attacker manipulates the trained DRL model, he should guarantee that he can dramatically alter the interpretation results without significantly hurting the performance of the original DRL model. To address this challenge, we propose to design our model poisoning attack against DRL interpretations by fine-tunning the pre-trained DRL model with the objective function that combines the ordinary loss of the original DRL model with a penalty term that involves the interpretation results. Without loss of generality, in the following, we use the DQN algorithm [15] as a representative example of DRL to present the proposed model poisoning attack (i.e., MPDRLI). Note that the proposed MPDRLI is model-agnostic and can generalize to other DRL algorithms naturally.

Note that a deep Q network estimates the Q-value function by leveraging a multi-layered neural network. The input for the network is the current state, and the output is the corresponding Q-value for each of the action. Specifically, for each given state-action pair (i.e., state s_t and action a_t), the deep Q-Network predicts the corresponding Q-Value $Q(s_t, a_t; \Theta)$ through a forward pass, where Θ are the parameters of the deep Q network. The value $Q(s_t, a_t; \Theta)$ is an estimate of the expected future reward that can be obtained from (s_t, a_t). The corresponding policy for a DQN is obtained by choosing the action with the maximum Q-value for each state. The deep Q-network parameters (i.e., Θ) can be derived by minimizing the following mean-squared Bellman error

$$J(\Theta) = \mathbb{E}[(r(s_t, a_t) + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \Theta^-) - Q(s_t, a_t; \Theta))^2],$$

where Θ^- represents the parameters of the target network, and the parameters Θ of the online network are updated by sampling gradients from minibatches of past transition tuples. The above mean squared error measures the squared difference between the target Q value (i.e., $r(s_t, a_t) + \gamma \max_{a \in \mathcal{A}} Q(s_t, a; \Theta^-)$) and the current Q output (i.e., $Q(s_t, a_t; \Theta)$).

Here, we consider the case where the attacker wants to secretly alter the model parameters, such that the agent cannot figure out what features are really most important for the current action prediction with targeted interpretation method. In our model poisoning attack settings, the attacker has no knowledge of the training dataset $\mathcal{D}_{tr}$ but he can collect a substitute dataset $\mathcal{D}'_{tr}$ by iteratively running the targeted model. Let $p_{t,k}(\Theta^*)$ denote the set of pixels that had the top k highest saliency map values with interpreter $\mathcal{I}$ of the original clean DQN model (parameterized by Θ^*), for the state-action pair (s_t, a_t). Note that to avoid being detected, the attacker should maintain the performance of the retrained DRL model, while only focusing on attacking the interpretation results. In order to achieve the attack goals, based on the pre-trained model parameters Θ^* and the substitute dataset $\mathcal{D}'_{tr}$, the attacker can manipulate the original clean DQN model as follows

$$\min_{\tilde{\Theta}} \mathcal{L}(\tilde{\Theta}) = \mathbb{E}[(r(s_t, a_t) + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \tilde{\Theta}^-) - Q(s_t, a_t; \tilde{\Theta}))^2] + \lambda_3 * \frac{1}{T+1} \sum_{t=0}^{T} \sum_{j \in p_{t,k}(\Theta^*)} \exp(\mathcal{I}(s_t, a_t; \tilde{\Theta})), \quad (9)$$

where Θ^* is the model parameters of the original unattacked DQN model, λ_3 is a trade-off parameter, and the penalty term is designed to reduce the interpretation scores of the pixels that originally had the top k highest values. By differentiating the above loss function with respect to $\tilde{\Theta}$, we can get the following gradient

$$\frac{\partial \mathcal{L}(\tilde{\Theta})}{\partial \tilde{\Theta}} = \mathbb{E}[(r + \gamma \max_{a \in \mathcal{A}} Q(s_t, a; \tilde{\Theta}^-) - Q(s_t, a_t; \tilde{\Theta})) \frac{Q(s_t, a_t; \tilde{\Theta})}{\partial \tilde{\Theta}}] + \lambda_3 * \frac{1}{T+1} \sum_{t=0}^{T} \sum_{j \in p_{t,k}(\Theta^*)} (\exp(\mathcal{I}(s_t, a_t; \tilde{\Theta})) * \frac{\partial \mathcal{I}(s_t, a_t; \tilde{\Theta})}{\partial \tilde{\Theta}}). \quad (10)$$

Then, based on the above, we can re-train the DQN model by using the projected gradient descent method [15]. Note that the original parameters Θ^* are used as the initialized parameters.

Discussion. In the above, we consider how to reduce the interpretation scores of the pixels that originally have the top k highest values. In practice, we can also make the interpretations always say that some particular region of the input (e.g., boundary or corner of the image), is important regardless of the input.

5 EXPERIMENTS

In this section, we firstly introduce the experimental setup in Section 5.1. Then, we conduct experiments to validate the effectiveness of the proposed universal adversarial attack against DRL interpretations (i.e., UADRLI) in Section 5.2. Lastly, in Section 5.3, we verify the effectiveness of the proposed model poisoning attack against DRL interpretations (i.e., MPDRLI).

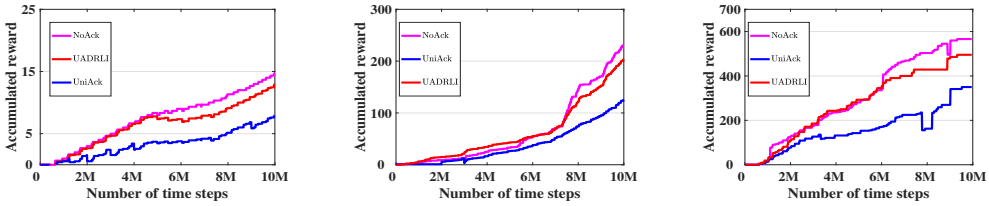
5.1 Experimental Setup

Model Setting. Our experimental implementation of the environments builds on OpenAI gym's control environments with the Atari physics simulator. In experiments, we train agents on Pong, Breakout, and SpaceInvaders by using two state-of-the-art DRL algorithms (i.e., A3C and DQN). We choose these three games because each of them poses a different set of challenges and the two adopted DRL algorithms have historically exceeded human-level performance on them. For evaluation, the game's randomness seed is reset for every episode. We also adopt two representative DRL interpreters, i.e., the Jacobian and gradient saliency.

Table 1: The setting of parameters.

Parameter	Value	Parameter	Value
learning rate	1e-4	λ_1	1.0
discount factor (γ)	0.99	k	706
λ_3	1.0	-	-

Network Architecture and Parameter Setting. For the adopted A3C algorithm, all of the Atari agents have the same recurrent architecture. The input at each time step is a preprocessed version of the current frame, and the preprocessing operations include gray-scaling, down-sampling by a factor of 2, cropping the game space to an 80×80 square and normalizing the values to $[0, 1]$. This input is processed by 4 convolutional layers (each with 32 filters, kernel sizes of 3, strides of 2, and paddings of 1), followed by an LSTM layer with 256 hidden units and a fully connected layer with $|\mathcal{A}|$+1, where $|\mathcal{A}|$ denotes the dimension of action



(a) Pong (b) Breakout (c) SpaceInvader

Figure 1: Performance comparison of adversarial attacks on the accumulative reward.

space. For the adopted DQN algorithm, the network architecture is a convolutional neural network with 3 convolution layers and a fully-connected hidden layer. Specifically, the first hidden layer convolves 32 filters of 8×8 with stride 4 with the input image and applies a rectifier nonlinearity. The second hidden layer convolves 64 filters of 4×4 with stride 2, again followed by a rectifier nonlinearity. The third hidden layer convolves 64 filters of 3×3 with stride 1 followed by a rectifier. The final fully-connected layer consists of 512 rectifier units. Here, the input RGB frame (i.e., the observed state) is rescaled to 84×84. The output layer is a fully-connected linear layer with a single output for each valid candidate action. The setting of parameters is given in Table 1.

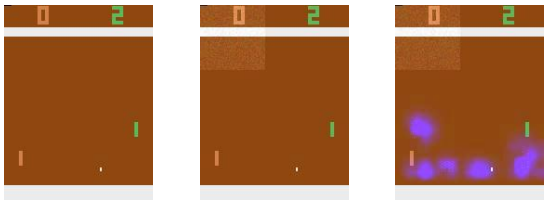
Baselines. For the proposed universal attack, in experiments, we adopt two baselines: Firstly, we adopt the uniform adversarial attack as the baseline, denoted as **UniAck**, which is a direct extension of the traditional adversarial attacks on DRL. In UniAck, we apply the generated universal perturbation to the observed state at each time step. Additionally, we also compare the agent's performance under our proposed adversarial attacks with that under no adversarial attack, denoted as **NoAck**. For the proposed model poisoning attack, since there is no existing work addressing the vulnerability of DRL to poisoning attacks, we adopt the no model poisoning attack baseline (dubbed as **NoPAck**).

5.2 Experiments for Adversarial Attack

In this section, we evaluate the performance of the proposed universal adversarial attack (i.e., UADRLI). Unless otherwise specified, in this experiment, we adopt the Jacobian saliency and restrict the attacker to only manipulating the pixels at the top-left corner of the input image, and set the size of the attacked image area as 40×40. Additionally, given an episode that consists of an alternating sequence of state-action pairs, for the proposed UADRLI, we only attack 10% of these state-action pairs that have the lowest Q value variance. In contrast, for the baseline UniAck, we attack 10% of these state-action pairs that have the largest Q value variance.

Performance of Adversarial Attack on the Discounted Accumulative Reward. Next, we compare the performance of the proposed UADRLI with that of the two baselines by averaging the total reward accumulated by the target agent. Here, for the proposed UADRLI, the universal perturbation $\boldsymbol{\delta}$ is crafted with $\epsilon = 0.12$. For an episode of game playing, we only attack 10% of the state-action pairs that have the lowest Q value variance. In contrast, for the adopted baseline UniAck, we attack 10% of the state-action pairs that have the largest Q value variance. The experimental results on the three adopted games are shown in Figure 1, where the y-axis is the accumulated reward and the x-axis is the number of time steps. The reference line in the figure is the purple line which corresponds to the reward function under no attack (NoAck). From this figure, we can see that the proposed UADRLI can reach the similar effect of the original unattacked DRL model. In contrast, for the adopted baseline UniAck that attacks the time steps where the variance of the corresponding states are high, it suffers from the most severe reduction in accumulated reward, which is also in accordance to the conclusion of Lemma 3 (in Appendix) that attacking the states with low variance incurs low decrease in the accumulated reward. In sum, regardless of which game the agent plays, the proposed UADRLI indeed incurs minor decrease in the policy's performance.

Visualization. Next, we visually demonstrate the effectiveness of the proposed UADRLI. To better visualize the experimental results, in this experiment, we set the value of ϵ as 0.12, and set the size of the attacked image region as 20×20. In practice, we can set ϵ as a much smaller value to make the crafted universal perturbation $\boldsymbol{\delta}$ more imperceptible. In Figure 2, we plot the visualization results on the Pong game. In this figure, the leftmost (i.e., Figure 2a) is the original input image, the second one (i.e., Figure 2b) is the adversarial image that is derived by adding the crafted universal perturbation $\boldsymbol{\delta}$ to its original unattacked state (i.e, Figure 2a), and the rightmost (i.e, Figure 2c) highlights the most important features that are identified by the adopted Jocabian interpreter. By adding the universal perturbation, the trained agent, who should have taken the "down" action, take the "up" action instead. And the added perturbations at the top-left corner are the the real reason for the wrong action prediction. However, from Figure 2c, we can observe that the crafted universal perturbation can successfully fool the adopted interpreter. That is to say, the adopted interpreter cannot identify the attacked image region. These results show that the proposed UADRLI not only leads the trained agent to make wrong action judgement but also can avoid being detected.

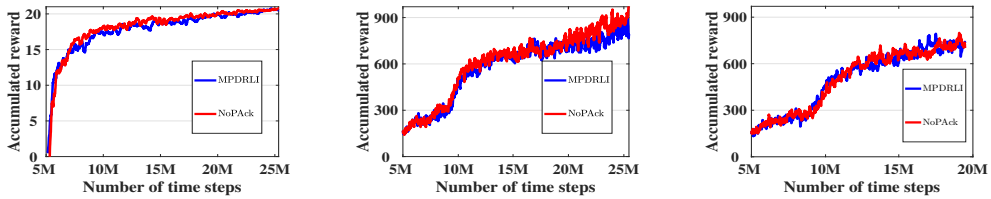


(a) (b) (c)

Figure 2: Visualization results for the Pong game.

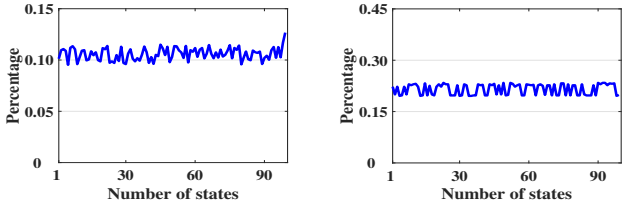
5.3 Experiments for Model Poisoning Attack

In this section, we evaluate the performance of the proposed model poisoning attack (i.e., MPDRLI). The adopted DRL algorithm is



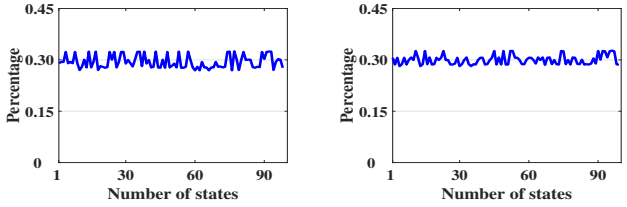
(a) Pong (b) Breakout (c) SpaceInvader

Figure 3: Performance of the proposed model poisoning attack on the accumulative reward.



(a) Pong (b) Breakout

Figure 4: Percentage of identified features when $k = 706$.



(a) Pong (b) Breakout

Figure 5: Percentage of identified features when $k = 1,058$.

DQN, and the adopted interpreter is the gradient saliency. In the following experiment, unless otherwise specified, we set $\lambda_3 = 1.0$ and $k = 706$, and there are $84 * 84$ features in total.

Performance on the Accumulated Discounted Reward. First of all, we compare the performance of the proposed MPDRLI with that of the adopted baseline (i.e., NoPAck) on the accumulated reward. The experimental results are reported in Figure 3, where x-axis denotes the number of time steps and y-axis represents the accumulated reward. From this figure, we can see that the proposed MPDRLI achieves similar performance as that of the adopted no attack baseline (i.e., NoPAck). These experimental results verify that the proposed model poisoning attack incurs minor decrease to the performance of the original DRL model. In this way, the proposed poisoning attack can remain undetected (i.e., stealthy). In practice, if the performance of the poisoned DRL model is significantly decreased, the agent can easily detect the launched attacks by just checking the accumulated reward.

Performance on Altering the Interpretation Results. We then evaluate the effectiveness of the proposed MPDRLI in terms of altering the interpretation results. In this experiment, we first let the agent play a game with the original DRL model, and then select a sequence of state-action pairs from this entire game episode. Next, for each selected state-action pair, we first use the adopted interpreter to identify the k highest ranked features that are crucial for explaining the prediction decision made by the original DRL model. Then, for this state-action pair, we select the k highest ranked features that explain the decision made by the poisoned DRL model. After that, we count the number of the features in the intersection between the two selected feature subsets. The lower the number of features in the intersection, the better the performance of the proposed MPDRLI. Based on this count, we can calculate the percentage of features in the intersection over the number of features originally identified by the interpreter. The averaged experimental results are reported in Figure 4. Here, we vary the number of the selected state-action pairs from 1 to 99. Take Figure 4a as an example. From this figure, we can see that the percentage of the features in the intersection is only around 0.10. In other words, the interpretation results generated from the poisoned DRL model are significantly different from that of the original clean model. Hence, the proposed MPDRLI can guarantee that the top-ranked features cannot be identified by the agent. In Figure 5, we also show the experimental results when the value of k is set as $1,058$, which means that the attacker attacks 15% of all the features that have the largest feature importance scores. From this figure, we can also derive the same observation that the interpretation results generated from the poisoned DRL model are significantly different from that of the original clean DRL model.

6 RELATED WORK

Adversarial Attacks against Deep Reinforcement Learning. Recent studies [3, 7, 8, 16, 17] show that DRL algorithms are unavoidably susceptible to adversarial perturbations. [7] makes use of white-box assumptions and proposes an attack method where the attacker attacks every time step by applying the FGSM. [8] designs a targeted controlling attack where the attacker can manipulate the policy by adding the imperceptible noise to the observations of the environment. [16] proposes adversarial attacks that lead the agent into increased probability of taking worst possible actions. [3] verifies the transferability of adversarial examples across different DQN models. [17] unveils how little it takes to deceive an DRL policy by considering three restrictive settings. However, these methods only focus on attacking specific states using traditional per-instance techniques, and ignore the end goal of the entire DRL task. In contrast, [12, 19] consider how to significantly deteriorate the agent's end reward. However, the agent can easily identify the adversarial attack by simply comparing the consequent end goal with his own desired one. Additionally, all of the above works only focus on how to craft state-dependent perturbations, which is computationally intractable in long state sequences. They also make an implicit assumption that the attacker has the ability of

manipulating the whole input data. Lastly, they do not study the vulnerability of DRL interpretations to the security threats.

Interpretation Models for Deep Reinforcement Learning. Based on the interpretation stages, existing DRL interpretation works can be generally divided into the following two categories: intrinsic and post-hoc. The latter case does not require modifying the model architectures or parameters, thereby leading to higher prediction accuracy [6]. Motivated by this, considerable works [2, 5, 15, 21, 22, 24, 25] have been proposed to provide post-hoc explanations for explaining the model's output for a given input. For example, [22] takes a closer look at a slightly modified version of Grad-CAM in the context of deep RL on Atari games. However, all of these interpretation works assume a secure and reliable environment, and do not consider the vulnerability of DRL interpretations to the malicious attacks.

Adversarial Attacks against Deep Learning Interpretations. Very recently, some works [1, 4, 11, 23, 26] are beginning to study the vulnerability of the interpretation methods for deep neural networks. For example, [4] demonstrates that explanation maps can be sensitive to small perturbations in the image. Their results can be thought of as untargeted manipulations, i.e., perturbations to the image which lead to an unstructured change in the explanation map. However, these works only focus on how to design adversarial attacks against supervised deep neural network interpretations, and cannot be directly applied to DRL due to its unique characteristics (e.g., the sequentiality of decision-making process and the end-goal oriented property). Furthermore, these works only consider the adversarial attack, and do not consider the poisoning attack.

7 CONCLUSIONS

To the best of our knowledge, we are the first to study the vulnerability of DRL interpretations to the malicious attacks. More specifically, in this paper, we firstly present an universal adversarial attack against DRL interpretations (i.e., UADRLI), from which the attacker can add the crafted universal perturbation uniformly to the environment states in a maximum number of steps to incur minimal damage to the agent's end goal. Then, we design a model poisoning attack against DRL interpretations (i.e., MPDRLI), based on which the attacker can significantly alter the interpretation results while incurring minor damage to the performance of the original DRL model. Both theoretical analysis and extensive experimental results are provided to demonstrate the effectiveness of our proposed malicious attacks against DRL interpretations. Our analysis provides valuable insights on malicious attacks against DRL interpretations which will be useful to the researchers who will study the approaches to defend such malicious attacks.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and helpful suggestions. This work is supported in part by the US National Science Foundation under grants IIS-1924928, IIS-1938167 and OAC-1934600. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity checks for saliency maps. In *NeurIPS*. 9505–9515.
[2] Akanksha Atrey, Kaleigh Clary, and David Jensen. 2019. Exploratory Not Explanatory: Counterfactual Analysis of Saliency Maps for Deep Reinforcement Learning. *arXiv preprint arXiv:1912.05743* (2019).
[3] Vahid Behzadan and Arslan Munir. 2017. Vulnerability of deep reinforcement learning to policy induction attacks. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*. Springer, 262–275.
[4] Amirata Ghorbani, Abubakar Abid, and James Zou. 2019. Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
[5] Sam Greydanus, Anurag Koul, Jonathan Dodge, and Alan Fern. 2017. Visualizing and understanding atari agents. *arXiv preprint arXiv:1711.00138* (2017).
[6] Mengdi Huai, Di Wang, Chenglin Miao, and Aidong Zhang. 2020. Towards Interpretation of Pairwise Learning. In *Thirty-fourth AAAI Conference on Artificial Intelligence*.
[7] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. 2017. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284* (2017).
[8] Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. 2019. Targeted Attacks on Deep Reinforcement Learning Agents through Adversarial Observations. *arXiv preprint arXiv:1905.12282* (2019).
[9] Rahul Iyer, Yuezhang Li, Huao Li, Michael Lewis, Ramitha Sundar, and Katia Sycara. 2018. Transparency and explanation in deep reinforcement learning neural networks. In *Proc. of the AAAI/ACM Conference on AI, Ethics, and Society*.
[10] Michael Kearns and Satinder Singh. 2002. Near-Optimal Reinforcement Learning in Polynomial Time. *Mach. Learn.* (2002).
[11] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. 2019. The (un) reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer, 267–280.
[12] Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. 2017. Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748* (2017).
[13] Chenglin Miao, Qi Li, Lu Su, Mengdi Huai, Wenjun Jiang, and Jing Gao. 2018. Attack under Disguise: An Intelligent Data Poisoning Attack Mechanism in Crowdsourcing. In *Proc. of the 2018 World Wide Web Conference*. 13–22.
[14] Chenglin Miao, Qi Li, Houping Xiao, Wenjun Jiang, Mengdi Huai, and Lu Su. 2018. Towards data poisoning attacks in crowd sensing systems. In *Proc. of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 111–120.
[15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.
[16] Anay Pattanaik, Zhenyi Tang, Shuijing Liu, Gautham Bommannan, and Girish Chowdhary. 2018. Robust deep reinforcement learning with adversarial attacks. In *Proc. of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. 2040–2042.
[17] Xinghua Qu, Zhu Sun, Pengfei Wei, Yew-Soon Ong, and Abhishek Gupta. 2019. Minimalistic Attacks: How Little it Takes to Fool a Deep Reinforcement Learning Policy. *arXiv preprint arXiv:1911.03849* (2019).
[18] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *ICML*. 1889–1897.
[19] Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma, Yan Zheng, Kangjie Chen, and Yang Liu. 2020. Stealthy and efficient adversarial attacks against deep reinforcement learning. *arXiv preprint arXiv:2005.07099* (2020).
[20] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.
[21] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. 2015. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581* (2015).
[22] Laurens Weitkamp, Elise van der Pol, and Zeynep Akata. 2018. Visual rationalizations in deep reinforcement learning for atari games. In *Benelux Conference on Artificial Intelligence*. Springer, 151–165.
[23] Kaidi Xu, Sijia Liu, Pu Zhao, Pin-Yu Chen, Huan Zhang, Quanfu Fan, Deniz Erdogmus, Yanzhi Wang, and Xue Lin. 2018. Structured adversarial attack: Towards general implementation and better interpretability. *arXiv preprint arXiv:1808.01664* (2018).
[24] Liu Yuezhang, Ruohan Zhang, and Dana H Ballard. 2018. An Initial Attempt of Combining Visual Selective Attention with Deep Reinforcement Learning. *arXiv preprint arXiv:1811.04407* (2018).
[25] Tom Zahavy, Nir Ben-Zrihem, and Shie Mannor. 2016. Graying the black box: Understanding dqns. In *ICML*. 1899–1908.
[26] Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang. 2020. Interpretable deep learning under fire. In *29th USENIX Security Symposium (USENIX Security 20)*.

8 APPENDIX

Before presenting the proof of Theorem 1 and 2, we firstly introduce the assumptions, the definition and the concept of the attacked MDPs, which are used in the latter proof.

Assumption 1 (MDP Regularity)**.** Suppose MDP $\mathcal{M} =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ has finite state space, finite action space, and bounded reward function, i.e., $|\mathcal{S}| \leq \infty$, $|\mathcal{A}| \leq \infty$, and $\|\mathcal{R}\|_\infty \leq R$, where R is the upper bound of reward function.

Assumption 2 (Transition/Reward Continuity)**.** Suppose two MDPs: $\mathcal{M}' =< \mathcal{S}, \mathcal{A}, \mathcal{P}', \mathcal{R}', \gamma >$ and $\mathcal{M}'' =< \mathcal{S}, \mathcal{A}, \mathcal{P}'', \mathcal{R}'', \gamma >$ share the same state space and action space. $\mathcal{P}''(\cdot|s,a) = \mathcal{P}'(\cdot|\tilde{s},a)$ and $\|s - \tilde{s}\| \leq \epsilon$. Suppose there exists a constant δ such that for any $s \in \mathcal{S}$ and $a \in \mathcal{A}$, $\|\mathcal{P}'(\cdot|s,a) - \mathcal{P}''(\cdot|s,a)\|_1 \leq \delta$ and $|\mathcal{R}'(s,a) - \mathcal{R}''(s,a)| \leq \delta$. In some literature, $\mathcal{M}''$ is also said to be δ-approximation of $\mathcal{M}'$.

Definition 1 ($\mathcal{T}$-step value function)**.** Suppose MDP $\mathcal{M} =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ satisfies Assumption 1. Following the definition of value function under policy π: $V^\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t = \pi(s_t))|s_0 = s]$ in previous section, we define $\mathcal{T}$-step value function to be the truncation of the first $\mathcal{T}$ discounted returns. Specifically, $\mathcal{V}^\pi(s,\mathcal{T}) = \mathbb{E}[\sum_{t=0}^{\mathcal{T}-1} \gamma^t \mathcal{R}(s_t, \pi(s_t))|s_0 = s]$. Note the expectation is taken over all possible paths the agent might follow starting from s and of fixed length $\mathcal{T}$. Further, we define the optimal $\mathcal{T}$-step value function $\mathcal{V}^*(s,\mathcal{T}) = \max_\pi \mathcal{V}^\pi(s,\mathcal{T})$. The optimal value function could be viewed as the limit case of optimal $\mathcal{T}$-step value function, i.e., $\mathcal{V}^*(s) = \lim_{\mathcal{T}\to\infty} \mathcal{V}^*(s,\mathcal{T})$. Finally, we denote the maximum possible $\mathcal{T}$-step return by $\mathcal{G}_\mathcal{T} = \max_{s\in\mathcal{S}} \mathcal{V}^*(s,\mathcal{T})$.

Definition 2 (The attacked MDPs)**.** Here, we define the following two MDPs, which are different from original MDP $\mathcal{M} =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ attempted to be attacked in either transition probability or immediate reward, indicating the new environments under malicious attacks.

- $\mathcal{M}_1 =< \mathcal{S}, \mathcal{A}, \mathcal{P}_1, \mathcal{R}_1, \gamma >$, where $\mathcal{P}_1(\cdot|s,a) = \mathcal{P}(\cdot|\tilde{s},a)$, and $\mathcal{R}_1(\cdot|s,a) = \mathcal{R}(\cdot|s,a)$. $\mathcal{P}_1$ characterizes the system dynamics where the next state follows the distribution $\mathcal{P}(\cdot|\tilde{s},a)$ given the current state is s and the selected action is a, since the current state has been crafted to $\tilde{s}$.
- $\mathcal{M}_2 =< \mathcal{S}, \mathcal{A}, \mathcal{P}_2, \mathcal{R}_2, \gamma >$, where $\mathcal{P}_2(\cdot|s,a) = \mathcal{P}(\cdot|\tilde{s},a)$, and $\mathcal{R}_2(s,a) = \mathcal{R}(\tilde{s},a)$. The system transition model $\mathcal{P}_2$ is exactly the same as $\mathcal{P}_1$, while the immediate reward model $\mathcal{R}_2$ is different. The reward the agent instantaneously collects from the environment is $\mathcal{R}(\tilde{s},a)$ in $\mathcal{M}_2$ given that the current state is s and the selected action is a. We argue that both $\mathcal{R}_2$ and $\mathcal{R}_1$ are ubiquitous in real-world settings, depending on whether the environment from which the agent collects reward is aware of the state crafting. $\mathcal{M}_1$ characterizes the scenario where the attack is imperceptible to the system or the agent obtains reward based on his own assessment of the current state and action. On the contrary, $\mathcal{M}_2$ describes the setting where the immediate reward is evaluated externally and the attack is noticeable to the performance evaluation system.

Finding optimal malicious attack is equivalent to solving the optimal policy for $\mathcal{M}_1$ and $\mathcal{M}_2$. In Theorem 1, we bound the difference of optimal value functions between $\mathcal{M}_i$ and $\mathcal{M}$, where $i = 1, 2$. The importance of these bounds is to help us understand, to what extent the malicious attack affects the long term reward and how perceptible the attack could be to the participating agent. In Theorem 2, we further study, if the attack takes place only in a finite number of states, the difference of $\mathcal{T}$-step value functions between $\mathcal{M}_i$ and $\mathcal{M}$, where $i = 1, 2$.

8.1 Proof of Theorem 1

Proof. Recall Bellman's Equation for optimal value function is $V^*(s) = \max_{a\in\mathcal{A}} \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\mathcal{R}(s,a) + \gamma V^*(s')]$. Let $\mathcal{F}$ be the function class mapping from $\mathcal{S}$ to $\mathbb{R}$. Define Bellman's operator $\mathcal{B} : \mathcal{F} \to \mathcal{F}$ of MDP $\mathcal{M} =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ to be $\mathcal{B} \circ V^*(s) = \max_{a\in\mathcal{A}} \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\mathcal{R}(s,a) + \gamma V^*(s')]$.

$$\begin{aligned}\left|V^*(s) - V_1^*(s)\right| &= |\mathcal{B}\circ V^*(s) - \mathcal{B}\circ V_1^*(s)| \\ &= \max\{\mathcal{B}\circ V^*(s) - \mathcal{B}\circ V_1^*(s), \mathcal{B}\circ V_1^*(s) - \mathcal{B}\circ V^*(s)\} \\ \overset{1}{\leq} \max_{a\in\mathcal{A}} &|\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\mathcal{R}(s,a) + \gamma V^*(s')] - \mathbb{E}_{s'\sim\mathcal{P}_1(\cdot|s,a)}[\mathcal{R}_1(s,a) + \gamma V_1^*(s')]|.\end{aligned}$$

Suppose $\hat{a} \triangleq \arg\max_{a\in\mathcal{A}} \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\mathcal{R}(s,a) + \gamma V^*(s')]$, and $\hat{a}_1 \triangleq \arg\max_{a\in\mathcal{A}} \mathbb{E}_{s'\sim\mathcal{P}_1(\cdot|s,a)}\left[\mathcal{R}_1(s,a) + \gamma V_1^*(s')\right]$. The reason for inequality **1** is: if $V^*(s) \geq V_1^*(s)$, $\mathcal{B}\circ V^*(s) - \mathcal{B}\circ V_1^*(s) \leq \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,\hat{a})}[\mathcal{R}(s,\hat{a}) + \gamma V^*(s')] - \mathbb{E}_{s'\sim\mathcal{P}_1(\cdot|s,\hat{a})}\left[\mathcal{R}_1(s,\hat{a}) + \gamma V_1^*(s')\right]$; while if $V^*(s) < V_1^*(s)$, $\mathcal{B}\circ V_1^*(s) - \mathcal{B}\circ V^*(s) \leq \mathbb{E}_{s'\sim\mathcal{P}_1(\cdot|s,\hat{a}_1)}\left[\mathcal{R}_1(s,\hat{a}_1) + \gamma V_1^*(s')\right] - \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,\hat{a}_1)}[\mathcal{R}(s,\hat{a}_1) + \gamma V^*(s')]$. In both cases, the inequality **1** holds.

$$\begin{aligned}\left|V^*(s) - V_1^*(s)\right| &\leq \max_{a\in\mathcal{A}} |\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\gamma V^*(s')] - \mathbb{E}_{s'\sim\mathcal{P}_1(\cdot|s,a)}[\gamma V_1^*(s')]| \\ \overset{2}{\leq} &\underbrace{\max_{a\in\mathcal{A}} |\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\gamma V^*(s')] - \mathbb{E}_{s'\sim\mathcal{P}_1(\cdot|s,a)}[\gamma V^*(s')]|}_{I} \\ &+ \underbrace{\max_{a\in\mathcal{A}} |\mathbb{E}_{s'\sim\mathcal{P}_1(\cdot|s,a)}[\gamma V^*(s')] - \mathbb{E}_{s'\sim\mathcal{P}_1(\cdot|s,a)}[\gamma V_1^*(s')]|}_{II}.\end{aligned}$$

Inequality **2** follows from the triangle inequality. It is easy to get $II \leq \gamma\|V^* - V_1^*\|_\infty$. We now derive the bound for I.

$$\begin{aligned} I &= \max_{a\in\mathcal{A}} \Big|\sum_{s'\in\mathcal{S}} (\mathcal{P}(s'|s,a) - \mathcal{P}_1(s'|s,a))V^*(s')\Big| \cdot \gamma \\ &\overset{3}{\leq} \max_{a\in\mathcal{A}} \|\mathcal{P}(s'|s,a) - \mathcal{P}_1(s'|s,a)\|_1 \max_{s\in\mathcal{S}} V^*(s) \cdot \gamma \\ &\overset{4}{\leq} \max_{a\in\mathcal{A}} 2 \cdot TV(\mathcal{P}(\cdot|s,a), \mathcal{P}_1(\cdot|s,a)) \cdot \max_{s\in\mathcal{S}} V^*(s) \cdot \gamma \\ &\overset{5}{\leq} \max_{a\in\mathcal{A}} 2 \cdot TV(\mathcal{P}(\cdot|s,a), \mathcal{P}_1(\cdot|s,a)) \cdot \frac{R}{1-\gamma} \cdot \gamma. \end{aligned}$$

Inequality **3** follows from Hölder's inequality. Let $\mathcal{S}_0 \triangleq \{s' \in S : \mathcal{P}(s'|s,a) \geq \mathcal{P}_1(s'|s,a)\}$. We have $\|\mathcal{P}(s'|s,a) - \mathcal{P}_1(s'|s,a)\|_1 = \sum_{s'\in\mathcal{S}_0}(\mathcal{P}(s'|s,a) - \mathcal{P}_1(s'|s,a)) + \sum_{s'\in\mathcal{S}\setminus\mathcal{S}_0}(\mathcal{P}_1(s'|s,a) - \mathcal{P}(s'|s,a)) = 2\mathcal{P}(\mathcal{S}_0|s,a) - 2\mathcal{P}_1(\mathcal{S}_0|s,a) \leq 2 \cdot TV(\mathcal{P}(\cdot|s,a), \mathcal{P}_1(\cdot|s,a))$. Therefore, inequality **4** holds. Recall $V(s) \triangleq \mathbb{E}[\sum_{t\geq 0} \gamma^t \mathcal{R}(s_t, a_t)] \leq \mathbb{E}[\sum_{t\geq 0} \gamma^t R] \leq \frac{R}{1-\gamma}$. Hence, inequality **5** holds. With all these combined, we have the following

$$\begin{aligned}\|V^* - V_1^*\|_\infty &= \max_{s\in\mathcal{S}} \left|V^*(s) - V_1^*(s)\right| \\ \leq \max_{a\in\mathcal{A}, s\in\mathcal{S}} &2 \cdot TV(\mathcal{P}(\cdot|s,a), \mathcal{P}_1(\cdot|s,a)) \cdot \frac{R}{1-\gamma} \cdot \gamma + \|V^* - V_1^*\|_\infty \cdot \gamma.\end{aligned}$$

Reorganizing the last inequality, we have: $\|V^* - V_1^*\|_\infty \leq \frac{2\gamma R}{(1-\gamma)^2} \max_{s\in\mathcal{S}, a\in\mathcal{A}} TV(\mathcal{P}(\cdot|s,a), \mathcal{P}_1(\cdot|s,a))$. Recall that $\mathcal{P}$ and $\mathcal{P}_1$ are $L\epsilon$ approximate. Therefore, together with inequality 3, we have

$$\begin{aligned} I &\leq \max_{a\in\mathcal{A}} \|\mathcal{P}(s'|s,a) - \mathcal{P}_1(s'|s,a)\|_1 \max_{s\in\mathcal{S}} V^*(s)\cdot\gamma \\ &\leq L\epsilon \cdot \frac{R}{1-\gamma}\cdot\gamma. \end{aligned}$$

With II combined, we have $\|V^* - V_1^*\|_\infty \leq \frac{\gamma RL}{(1-\gamma)^2}\epsilon$, which completes the first part of proof.

$$\begin{aligned} &\left|V^*(s) - V_2^*(s)\right| = |\mathcal{B}\circ V^*(s) - \mathcal{B}\circ V_2^*(s)| \\ &\underset{6}{\leq} \max_{a\in\mathcal{A}} |\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\gamma V^*(s')] - \mathbb{E}_{s'\sim\mathcal{P}_2(\cdot|s,a)}[\gamma V_2^*(s')]| \\ &\quad + \max_{a\in\mathcal{A}} |\mathcal{R}(s,a) - \mathcal{R}_2(s,a)| \\ &\leq \max_{a\in\mathcal{A}} |\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[\gamma V^*(s')] - \mathbb{E}_{s'\sim\mathcal{P}_2(\cdot|s,a)}[\gamma V_2^*(s')]| + l\epsilon. \end{aligned}$$

Suppose $\hat{a} \triangleq \arg\max_{a\in\mathcal{A}} \mathcal{R}(s,a)$, and $\hat{a}_2 \triangleq \arg\max_{a\in\mathcal{A}} \mathcal{R}_2(s,a)$. If $\max_{a\in\mathcal{A}} \mathcal{R}(s,a) \geq \max_{a\in\mathcal{A}} \mathcal{R}_2(s,a)$, $|\max_{a\in\mathcal{A}} \mathcal{R}(s,a) - \max_{a\in\mathcal{A}} \mathcal{R}_2(s,a)| \leq \mathcal{R}(s,\hat{a}) - \mathcal{R}_2(s,\hat{a})$; otherwise, $|\max_{a\in\mathcal{A}} \mathcal{R}(s,a) - \max_{a\in\mathcal{A}} \mathcal{R}_2(s,a)| \leq \mathcal{R}(s,\hat{a}_2) - \mathcal{R}_2(s,\hat{a}_2)$. In both cases, $|\max_{a\in\mathcal{A}} \mathcal{R}(s,a) - \max_{a\in\mathcal{A}} \mathcal{R}_2(s,a)| \leq \max_{a\in\mathcal{A}} |\mathcal{R}(s,a) - \mathcal{R}_2(s,a)|$. Together with inequality 1 and triangle inequality, inequality 6 holds. The rest is the same as the first part of theorem. □

8.2 Proof of Theorem 2

Proof. Let ψ be any path of states starting from s under policy π^* of length $\mathcal{T}$, i.e., $\psi = (s_0^\psi = s, s_1^\psi, s_2^\psi, ..., s_{\mathcal{T}-1}^\psi)$. Let Ψ be the set of all possible such kind of paths. Further, we define $V^{\pi^*}(\psi) = \sum_{t=0}^{\mathcal{T}-1} \gamma^t \mathcal{R}(s_t^\psi, \pi^*(s_t^\psi))$, and $\mathbb{P}^{\pi^*}[\psi] = \prod_{t=0}^{\mathcal{T}-2} \mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi))$. $V^{\pi^*}(\psi)$ is the total discounted reward the agent could receive along path ψ under policy π^* in $\mathcal{M}$, and $\mathbb{P}^{\pi^*}[\psi]$ is the probability of the agent taking path ψ under policy π^* in $\mathcal{M}$. Analogously, we have $V_i^{\pi^*}(\psi)$ and $\mathbb{P}_i^{\pi^*}[\psi]$.

The proof is a modification of Lemma 4 in [10]. According to the definition of $\mathcal{T}$-step value function under policy π^*, we know $\mathcal{V}^{\pi^*}(s,\mathcal{T}) = \sum_{\psi\in\Psi} \mathbb{P}^{\pi^*}[\psi] V^{\pi^*}(\psi)$. We define the set of all θ-small paths, $\Psi_1 = \{\psi \in \Psi : \exists t, \mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)) \leq \theta\}$. Naturally, the set of paths where all transition probabilities under π^* is larger than θ is $\Psi_2 = \Psi \setminus \Psi_1$. Thus, we have $\mathcal{V}^{\pi^*}(s,\mathcal{T}) = \sum_{\psi\in\Psi_1} \mathbb{P}^{\pi^*}[\psi] V^{\pi^*}(\psi) + \sum_{\psi\in\Psi_2} \mathbb{P}^{\pi^*}[\psi] V^{\pi^*}(\psi)$.

We have $|\mathcal{P}_i(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)) - \mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi))| \leq \delta$, following from the continuity assumption. For any path $\psi \in \Psi_2$, for all $0 \leq t \leq \mathcal{T}-2$, it is not difficult to see $\mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)) + \frac{\delta}{\theta}\theta \leq \mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)) + \frac{\delta}{\theta}\mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi))$, and $\mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)) - \frac{\delta}{\theta}\theta \geq \mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)) - \frac{\delta}{\theta}\mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi))$. Therefore, we have

$$\begin{aligned} &(1-\frac{\delta}{\theta})\mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)) \leq \mathcal{P}_i(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)) \\ &\leq (1+\frac{\delta}{\theta})\mathcal{P}(s_{t+1}^\psi | s_t^\psi, \pi^*(s_t^\psi)). \end{aligned}$$

Furthermore, we can derive that for any $\psi \in \Psi_2$, $(1-\frac{\delta}{\theta})^{\mathcal{T}} \mathbb{P}^{\pi^*}[\psi] \leq \mathbb{P}_i^{\pi^*}[\psi] \leq (1+\frac{\delta}{\theta})^{\mathcal{T}} \mathbb{P}^{\pi^*}[\psi]$.

Now let us study the property of path $\psi \in \Psi_1$. $\sum_{\psi\in\Psi_1} \mathbb{P}_i^{\pi^*}[\psi] V^{\pi^*}(\psi)$ is upper bounded by $\theta \cdot |\mathcal{S}| \cdot \mathcal{T}\mathcal{G}_\mathcal{T}$ and $\sum_{\psi\in\Psi_1} \mathbb{P}^{\pi^*}[\psi] V_i^{\pi^*}(\psi)$ is upper bounded by $(\theta+\delta)\cdot|\mathcal{S}|\cdot\mathcal{T}\mathcal{G}_\mathcal{T}$. Thus, we have $|\sum_{\psi\in\Psi_1} \mathbb{P}_i^{\pi^*}[\psi] V_i^{\pi^*}(\psi) - \sum_{\psi\in\Psi_1} \mathbb{P}^{\pi^*}[\psi] V^{\pi^*}(\psi)| \leq (\delta+2\theta)|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T}$.

For any path ψ, $|V^{\pi^*}(\psi) - V_i^{\pi^*}(\psi)| \leq \mathcal{T}\delta$. With Ψ_1 and Ψ_2 combined, we have

$$\begin{aligned} &\mathcal{V}_i^{\pi^*}(s,\mathcal{T}) \triangleq \sum_{\psi\in\Psi_1} \mathbb{P}_i^{\pi^*}[\psi] V_i^{\pi^*}(\psi) + \sum_{\psi\in\Psi_2} \mathbb{P}_i^{\pi^*}[\psi] V_i^{\pi^*}(\psi) \\ &\leq \{\sum_{\psi\in\Psi_1} \mathbb{P}^{\pi^*}[\psi] V^{\pi^*}(\psi) + (\delta+2\theta)|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T}\} + \sum_{\psi\in\Psi_2} \mathbb{P}_i^{\pi^*}[\psi] V_i^{\pi^*}(\psi) \\ &\leq \{V_{\Psi_1}^{\pi^*}(s,\mathcal{T}) + (\delta+2\theta)|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T}\} + \sum_{\psi\in\Psi_2} (1+\frac{\delta}{\theta})^{\mathcal{T}} \mathbb{P}^{\pi^*}[\psi](V^{\pi^*}(\psi) + \mathcal{T}\delta) \\ &\leq (1+\frac{\delta}{\theta})^{\mathcal{T}}(V_{\Psi_1}^{\pi^*}(s,\mathcal{T}) + V_{\Psi_2}^{\pi^*} + \mathcal{T}\delta) + (\delta+2\theta)|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T} \\ &\leq (1+\frac{\delta}{\theta})^{\mathcal{T}}(\mathcal{V}^{\pi^*}(s,\mathcal{T}) + \mathcal{T}\delta) + (\delta+2\theta)|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T}. \end{aligned}$$

Similarly, we could also get inequality in the other direction $\mathcal{V}_i^{\pi^*}(s,\mathcal{T}) \geq (1-\frac{\delta}{\theta})^{\mathcal{T}}(\mathcal{V}^{\pi^*}(s,\mathcal{T})) - (\delta+2\theta)|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T}$. Here, $V_{\Psi_1}^{\pi^*}(s,\mathcal{T})$ and $V_{\Psi_2}^{\pi^*}(s,\mathcal{T})$ denote the part of $\mathcal{T}$-step value function which only include paths in Ψ_1 and Ψ_2, respectively.

Let us set θ to be $\sqrt{\delta}$. Note that we could assume $\delta \leq 1$ without loss of generality since we could always rescale the reward function to $[0,1]$. For some $\omega \geq 0$, if $\delta \leq (\frac{\omega}{12|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T}})^2$, $(1+\frac{\delta}{\theta})^{\mathcal{T}}\mathcal{V}^{\pi^*}(s,\mathcal{T})$ is no greater than $\frac{\omega}{8} + \mathcal{V}^{\pi^*}(s,\mathcal{T})$, and $(1+\frac{\delta}{\theta})^{\mathcal{T}}\mathcal{T}\delta \leq \frac{\omega}{8}$. Combined with $(\delta+2\theta)|\mathcal{S}|\mathcal{T}\mathcal{G}_\mathcal{T} \leq \frac{\omega}{4}$, we could get $|\mathcal{V}^{\pi^*}(s,\mathcal{T}) - \mathcal{V}_i^{\pi^*}(s,\mathcal{T})| \leq \omega$, where $i \in \{1,2\}$, independent of $s \in \mathcal{S}$. □

8.3 Lemma 3

Lemma 3. *Let the state and state-action value be $V(s)$ and $Q(s,a)$ respectively, and let the observed state with higher variance of Q value be state s_{t_1} and the observed state with smaller variance of Q value be s_{t_2}. The variance is taken over different actions. Let π denote the current policy. Then, we have the following*

$$\mathbb{E}_\pi[\sum_{t=0}^{T} \gamma^t r_t | do(s_{t_1} = \hat{s}_{t_1})] \leq \mathbb{E}_\pi[\sum_{t=0}^{T} \gamma^t r_t | do(s_{t_2} = \hat{s}_{t_2})],$$

where $do(s_{t_1} = \tilde{s}_{t_1})$ means the observed state at time step t_1 is perturbed from s_{t_1} to $\tilde{s}_{t_1}$ by using the adversarial perturbation, and $do(s_{t_2} = \tilde{s}_{t_2})$ means that the observed state at time step t_2 is changed from s_{t_2} to $\tilde{s}_{t_2}$ by utilizing the adversarial perturbation.