Privacy Attacks in Reinforcement Learning with Sensitive Rewards

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Kritika Prakash
20161039
kritika.prakash@research.iiit.ac.in

International Institute of Information Technology
(Deemed to be University)
Hyderabad - 500 032, INDIA
June 2022

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "**Privacy Attacks in Reinforcement Learning with Sensitive Rewards**" by **Kritika Prakash**, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____
Date

_____
Advisor: Dr. Praveen Paruchuri

_____
Co-advisor: Dr. Sujit Gujar

To my family and my friends

Acknowledgments

I would like to begin with thanking my advisors, Dr. Praveen Paruchuri and Dr. Sujit Gujar, for their continued guidance and support. They have been my anchor through this entire research journey. I am grateful to Praveen Sir for giving me an opportunity to be a part of his group, and for everything that I have learned from him, including but not limited to: how to think about problems, assessing their impact, facing adversity with patience, calmness, and optimism. Sujit Sir introduced me to the field of Differential Privacy - a field I would eventually develop a passion for. I'm grateful for the numerous tea-time conversations. He taught me how to dig deeper and get to the bottom of problems and solutions. I hope this is the beginning of a life-long mentorship with both my advisors. In the process of working with Fiza Husain on this project, I found a caring and wonderful friend. This work would not have been possible without her dedication and our discussion and debugging sessions over many late night video calls during a pandemic. My first research project was with Aditya Srinivas, and I had so much fun with him on the project. I am glad that we got to work together and have many conversations. I had the best time being a part of the Machine Learning Lab, and I learnt a lot from my lab-mates Tarun Gupta, Manisha Padala, Moin Moti, and more. I would like to thank Prof. P. J. Narayanan, Prof. Jayanthi, Prof. Manish, Prof. Jawahar, and Prof. Radhika for welcoming me to IIIT-H as a Lateral Entry student, supporting me throughout my education, and for being the best role-models for life. I am indebted to Andrew Trask, my mentor and my friend. Working on open-source research with him and many amazing folks at OpenMined has been elemental in building an intuition in Differential Privacy. He helped me gain immense confidence and experience, and helped find many wonderful opportunities for me. Over the course of my dual degree at IIIT Hyderabad, I discovered a family away from home. I'm grateful for all the wonderful memories and the journey I experienced with my amazing friends Roopal, Karthik, Haard, Tejaswi, Prakrati, Pooja, Hemanth, and a few more. They supported me in my weakest moments and celebrated with me in my achievements. I found a sister in Roopal, the person with the kindest heart - I hope we go on many more adventures together. I am thankful to Siddhartha for inspiring me, being there for me through everything, and magically making every day more beautiful and better. Finally, I wish to thank my family - my parents, Rashmi Khare and Om Prakash, and my younger brother Aryaman, for always being there with me through all the ups and downs, and enabling me to pursue science and research. I have felt lost and challenged many times in this journey, but I persisted through all the uncertainty with the support of everyone in my life, and I am glad that despite everything, this thesis has reached completion.

Abstract

Reinforcement Learning (RL) enables agents to learn how to perform various tasks from scratch. In domains like autonomous driving, recommendation systems, domestic service robots, and more, the need for privacy of individuals interacting with such systems has become evident. We aim to investigate the quality of solutions that claim to protect the reward functions that contain sensitive personal information, from being exploited. Optimal RL policies learned could cause a privacy breach if the policies memorize any part of the private reward. Differential Privacy (DP) - a popular technique used for computational privacy, empowers us to provide mathematically quantifiable privacy guarantees for queries made to "privatised" algorithms. Our work introduces the reward reconstruction attack that tries to reverse engineer the reward from a private policy using Inverse RL algorithms. This attack is the key part of the novel Privacy-Aware Inverse RL Analysis (PRIL) framework. We assume that the entire training process and the optimal policy is released publicly.

We study the set of existing differentially-private RL policies derived from various RL algorithms including Value Iteration, Deep Q Networks (DQNs), and Vanilla Proximal Policy Optimization (PPO). We propose the new PRIL analysis framework, that performs reward reconstruction as an adversarial attack on private policies that the agents may deploy. For this, we introduce the novel reward reconstruction attack, wherein we seek to reconstruct the original reward from a privacy-preserving policy using an Inverse RL algorithm. An adversary must do poorly at reconstructing the original reward function if the agent uses a tightly private policy. Using the PRIL framework, we empirically test the effectiveness of the privacy guarantee offered by the private algorithms on multiple instances of OpenAI Gym environments of varying complexities, specifically the discrete state-space *Frozen Lake* domain and the continuous state-space *Mountain Car* domain, employing a different Linear Programming based Inverse RL method for each domain.

From our experiments, we observe that there is no indication of the policy improving at reconstructing the reward with a relaxation in the budget - thus rendering all strategies ineffective at providing true reward privacy. Based on the analysis performed, we infer that there exists a gap between the current standard of privacy offered and the standard of privacy needed to protect reward functions in RL. We quantify the extent to which each private policy protects the reward function by measuring distances between the original and reconstructed rewards. We calculate the distance between the original reward and the reconstructed reward via various distance metrics. The larger the reward distance, the better the private agent's ability to protect the sensitive reward function using which it was trained. We present

the trends in test-time performance utility of private agents upon varying the level of privacy guaranteed (by altering the privacy budget during training). We also formally discover the differentially-private Bellman Update algorithm along with its proof of sensitivity. This is one of the first few works in this sub-field, and we hope that it is pursued with more rigour in the future.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Recent advancements in reinforcement learning (RL) have found widespread application in many real-world domains. Often, these domains are built from rich data sources or real-world environments, which could contain sensitive information of many individuals. This is evident in domains such as autonomous driving, recommendation systems, trading, industrial assembly, and domestic service robots. For example, a recommendation system agent for an online shopping platform not only tracks the purchases made, but also how long the user hovers over an item that he/she did not purchase. Another example is when an autonomous driving agent not only learns the dynamics behind driving, but also identifies people and predicts their behaviours on the streets, and how to respond to such situations. The reward function in these environments is often sensitive, as it is built on people's private information. RL agents trained in such environments should not expose the private information of individuals. We therefore, need to use privacy-preserving methods to protect the rewards from being memorized in the agent's policy in such a manner that the agent's utility is not compromised.

Recent works use *Differential Privacy* (DP) [14] to make the agent's policy quantifiably private, resultantly obtaining a rigorous privacy guarantee. Like many other areas in Artificial Intelligence (AI), RL has also started adopting DP to establish a mathematical way of guaranteeing data privacy in RL environments. However, an important question is: does the privacy guarantee offered by the private policies translate well into protecting the reward function? If not, how can we understand the gap and the steps we need to take in order to achieve meaningful privacy? What role does the type of reward function, algorithms (both RL and DP algorithms), and environment dynamics play? In other words, does privacy in policy translate well to privacy in reward? To investigate this, we first build autonomous agents whose aim is to learn private RL policies using existing privacy techniques. The intent is to reach a goal state that helps maximize the agent's expected reward in discrete and continuous state space environments. We then investigate the true level of reward privacy offered by the existing state-of-the-art privacy techniques for RL algorithms.

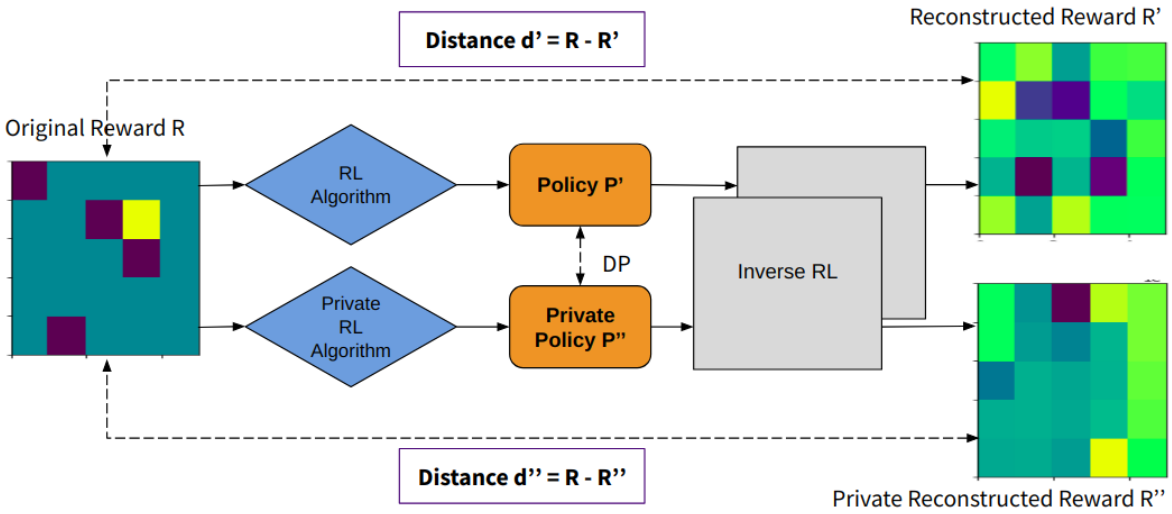

Figure 1.1 PRIL: Privacy-Aware Inverse RL analysis framework

1.2 Overview

We evaluate the private policies by estimating an adversary's ability to reconstruct the original reward function from the agent's learned policy. The field of Inverse RL [38] arose to solve this exact problem of extracting reward signal given the observed, optimal behaviour. Given the "reverse engineering" nature of inverse RL, the reconstructed reward can be used as an adversarial attack on environments with protected rewards. Building on this key intuition, we propose the PRIL analysis framework that first performs the reward reconstruction attack, and then computes its similarity to the original private reward via various distance metrics to test the strength of the attack.

We apply this framework over a set of privacy preserving techniques:

1. Bellman update DP
2. Rényi-DP in deep learning (DL)
3. Functional noise DP

These are combined with a set of RL algorithms:

1. Value Iteration (VI)
2. Deep Q Network (DQN)
3. Vanilla Proximal Policy Optimization (PPO)

We use the following Inverse RL methods to perform the reward reconstruction attack:

1. Inverse RL in Finite State Spaces

2. Inverse RL from Sampled Trajectories in Large State Spaces

We perform experiments on two RL domains:

1. Frozen Lake
2. Mountain Car

We build privacy into the agents from multiple perspectives: a DL perspective, an RL perspective, and a Deep RL perspective. Through experiments, we show that there exists a gap between the privacy offered via the current private RL methods, and the privacy needed to protect reward functions. We also present the privacy-utility trade-off achieved by each policy. We show that privacy in policy does not translate to privacy in reward, as the reconstruction error is independent of the ϵ-DP budget. DP based policies are unsuccessful at protecting the sensitive reward function due to a gap in privacy. Our experiments demonstrate that there is a need to further inspect the effectiveness of DP policies to protect sensitive reward functions. It is a serious privacy threat if an adversary is able to infer the rewards in spite of using a private policy.

1.3 Contributions

In summary, our key contributions are:

1. We study and analyze the existing set of privacy techniques for RL.
2. We introduce a novel privacy attack in RL: the reward reconstruction attack and the supporting PRIL framework.
3. We empirically evaluate the performance of various private deep RL policies within our framework.
4. We identify and quantify the gap between the privacy offered in policy and the privacy needed in reward.
5. We formally discover the differentially-private Bellman Update algorithm along with its proof of sensitivity.

1.4 Outline

The next chapter (2) discusses the related work. Chapter 3 gives us a background on RL, DP, and Inverse RL. Chapter 4 explains the PRIL framework and the basic building blocks of our experiments. Chapter 5 discusses the experiments, results, empirical analysis and a discussion on our findings. Chapter 6 concludes the work. Chapter 7 highlights the future work. And finally, chapter 7.1 presents the relevant publications, followed by the references.

All source code and experiment data are made publicly available [1]. An extended version of our AAAI 2022 conference paper with more details on the experiments and proofs is made available on ArXiv[2] [44].

[1]Link to code and data: `https://github.com/magnetar-iiith/PRIL`
[2]Link to ArXiV paper: `https://arxiv.org/abs/2112.05495`

Chapter 2

Related Work

2.1 Reinforcement Learning

The field of Reinforcement Learning [28, 5, 6] has grown exponentially over the past few years. More and more insightful work is being published, and as a result, better and safer real-world RL based systems are being built. RL is being applied to financial portfolio management, defeating world-champions in games like Go [52, 51] and chess, controlling power stations, making robots walk and perform human-like actions, and more. In this setting, an autonomous agent is tasked with learning the optimal behaviour in an uncertain environment in order to maximize the total gain. The agent learns from experience - by interacting with its environment. More specifically, it needs to find the best action to take in each state (policy). The state transition dynamics are modelled as a sequential decision making problem with the help of an MDP. Multiple RL strategies try to find a balance between exploration and exploitation. The agent explores so as to be able to find more rewarding actions, and it exploits (the policy) to maximize its gain over time. RL algorithms are broadly classified as Model-based and Model-free algorithms, based on whether the future value updates take into account whether a model's predictions are used to learn the controller. In this work, we focus on Model-free algorithms, such as Deep Q Networks, and Policy Gradient networks [55] - such as Proximal Policy Optimization.

2.2 Inverse Reinforcement Learning

The idea behind Inverse Reinforcement Learning [67] is to be able to learn the reward function for environments where only the optimal behaviour is known. In practice, the reward function can be very hard to specify and exhaustive to tune for large and complex problems, and this inspires the development of IRL. IRL can be considered to be a branch of Imitation Learning [23, 25] and Apprenticeship Learning [2], where the aim is to learn from demonstrations, such as an optimal policy. We use value functions as estimates of true gain achievable over time from each state, or each state-action pair. An optimal policy captures the best decisions that can be made in every state. Different policies also represent different unique strategies followed by domain experts. IRL is a reverse procedure of RL problems.

The original IRL solution algorithms [38] are Linear Programming formulations, where the constraints represent the optimal conditions. In our work, we use IRL in the cases of finite-state MDPs with known optimal policy, and infinite-state MDPs with unknown optimal policy but demonstrations are provided (as sampled trajectories). The latter is the more practical scenario. One key intuition is to find a reward function that shows that the optimal policy is far superior to all other sub-optimal policies. This is done to navigate the space of solutions, as it is an under-constrained problem, implying that many solutions would be optimal. More recent works employ techniques such as Information Theory and Deep Learning in the quest of the reward function.

2.3 Differential Privacy

Anonymity is not enough! The need for computational privacy became evident in 2006, when Netflix announced its $ 1 Million prize challenge for the best collaborative filtering [53, 29, 21, 22] algorithm to predict user ratings. They released an anonymous version of their dataset. In 2007, researchers from UT Austin de-anonymized the dataset [37] by cross-referencing it with the publicly available IMDB dataset. Various privacy methods have been used in the past, such as, distributed computation, encrypted computation, data swapping [49], k-anonymity [56], rule hiding [58], and anonymization [39]. But none of them were able to give us a rigorous guarantee or a method of quantifying the privacy loss. We needed a mathematical guarantee on the "process" which helps us quantify and upper bound our loss of privacy. Differential Privacy [15] uses a key intuition: uncertainty in the process means uncertainty for the attacker. Here, the attacker could be an adversary performing a membership inference attack [50], a data reconstruction attack, a linkage attack [57], etc. Differential Privacy is a system for publicly sharing information about a data system which masks individual contributions while retaining the bigger picture, by strategically adding some random noise to the process. Since it provides a worst-case guarantee against an arbitrarily powerful adversary, it does not require explicit adversary modelling. As the privacy loss introduced is quantifiable, it becomes easy to compose this loss across multiple queries. It is also easier to compute, in contrast to other cryptography based methods. Randomized response [62] is the simplest and most intuitive example of a differentially private query mechanism. Note that queries with a DP guarantee are immune to post-processing. No amount of actions taken on the output of a DP query can leak any more privacy than is quantified by the privacy loss of the query. It is being applied to applications such as healthcare, web browsers, contact tracing, Federated Learning systems, Census bureau, Deep Learning systems, and more. When Differential Privacy is used for training in Deep Learning [1, 10, 8, 63], a privacy accountant is attached to the ML optimizer, known as the "Privacy Engine". This accountant tracks detailed information such as higher order moments of the privacy loss random variable. It aims to improve the training efficiency and the privacy-utility trade-off. We make use of these techniques for our work.

2.4 Privacy Preserving Reinforcement Learning

Previous works on privacy-preserving RL make the use of DP. [59] shows how to achieve joint-DP in episodic RL via the novel private upper confidence bound and private Q-learning algorithms, where each training episode comes from a different environment. It provides bounds on the total regret and probably approximately correct guarantees. [60] makes the use of functional noise added to the value function to make Q-learning private in continuous spaces, such that neighbouring reward functions are indistinguishable. They claim that it protects the reward information from being exploited by methods such as inverse reinforcement learning. [7] propose the first differentially private algorithms for RL for the task of policy evaluation in the MDP setting using Monte-Carlo methods, along with a rigorous privacy and utility analysis. They show that the cost of privacy diminishes with increase in training batch sizes. [66] presents the first differentially private algorithm for off-policy evaluation along with a theoretical analysis of the privacy-preserving properties. [20] introduces a private way of using multi-party contextual bandits. They do so via a combination of a differentially private mechanism and secure multi-party computation along with an epsilon-greedy strategy for exploration. One of their key results is that at an optimal level of privacy, the private algorithm performs better than its non-private counterpart. [45] creates cryptographic solutions to ensure privacy guarantees in distributed RL systems in a multi-agent RL setting. However, the privacy in question is between individual agent's knowledge, instead of the actual training data. For the actor-critic class of algorithms, [31] presents a differentially private critic, and [48] presents a differentially private actor. Overall, the best performance comes from the DP-actor agent. One key observation by [48] is that the DP-actor agent had a larger advantage not just to the DP-critic strategy, but also to the strategy where both the actor and the critic network were trained in a differentially private manner (DP-both).

2.5 Generalization in Reinforcement Learning

Often, improving the generalization ability implies that the model is better equipped to dealing with noisy data. Specifically in the case of RL, an agent with strong generalization ability should be able to apply an optimal policy learnt in an environment to a different environment with a similar goal. For example, consider the agent trained to perform well across different environments in the OpenAI "CoinRun" domain introduced in [13], trying to solve the problem of over-fitting in deep RL. [12] introduces 16 procedurally generated gym environments with the aim of improving the generalization ability of deep RL agents. [32] introduces a simple randomization technique for improving the generalization ability across tasks. It uses various unseen visual patterns via convolutional neural networks (CNNs) for randomization of inputs along with a feature matching loss function. The aim is for agents to train on a broad range of low-level features in high-dimensional state spaces such as raw images. Overall, there is some correlation between making the RL policies more private and generalizable.

2.6 Adversarial Attacks in Reinforcement Learning

In RL, a significant amount of work has been done to perform adversarial attacks that target the quality of the learned policy. [19] propose a novel threat model of natural adversarial observations produced by an adversarial policy taking actions in a shared environment. They show that the adversary wins by confusing the victim, rather than learning a strong policy. They demonstrate the existence of their adversarial policy via simulated zero-sum robotics games. They also observed that deep RL strategies are not robust to small perturbations in the input images. [24] introduce another threat model that allows the adversary to add small perturbations to raw input images during training that do not interfere with human perception, with the goal of degrading test-time performance and testing for robustness. Resultantly, the introduce the Fast Gradient Sign Method (FGSM) attack in this context. They also contrast between the white box and black box methods of adversary modelling. [30] is an in-depth study on adversarial attacks in Deep RL policies, evaluating the effectiveness of various attacks. They also provide more efficient ways of performing the FGSM and Value Function (VF) adversarial attack. [11] is a similar study which focuses on RL attacks in the domain of security critical applications. Building on this, there has been work to make RL policies robust to such attacks. [40] propose the RADIAL framework that improves the robustness of Deep RL policies against norm-bounded adversarial perturbations in the inputs evaluated using the greedy worst-case reward. However, very few works have looked into privacy attacks in RL. [41] shows that agents can memorize the environment and its private transition dynamics, by performing privacy attacks using genetic algorithms and candidate inference. It is the first work to conduct studies on the privacy leakage problem in Deep RL. The adversary's task is to recover the transition dynamics of the environment from a well trained policy. [16] introduce an adversarial inverse RL (AIRL) algorithm based on an adversarial reward learning formulation to improve robustness wrt changes in the environment dynamics in large-scale domains.

2.7 Summary

While these previous works tackle building privacy in policies using Differential Privacy, they do not investigate its impact on the underlying private data i.e., the reward function used to learn the policies. Many adversarial attacks such as the membership inference attack [50], linkage attack [15], and data-reconstruction attack have been used to evaluate the level of privacy attainable by a data-analysis system. We introduce a new privacy attack that targets the private reward function. Our proposed attack - the reward-reconstruction attack is a special case of the data-reconstruction attack. We use inverse RL to learn the reward, and to assess the quality of private RL algorithms. Possibly, robustness of a policy could play a significant role in it's ability to preserve privacy.

Chapter 3

Background and Preliminaries

In this chapter, we give a background on the basics of Reinforcement Learning (RL), Inverse RL, and Differential Privacy (DP).

3.1 Reinforcement Learning

In this section, we give a brief background on the basics of Reinforcement Learning [54] within the scope of our work. We focus on environments with finite action spaces.

3.1.1 Non-deterministic discrete (finite) state space environments

Let $M = (S, A, P, R, \gamma, S_0)$ represent the MDP environment. Here, S is the set of finite discrete states, A is the set of finite actions, $P(s, a, s')$ is the transition probability of reaching state s' by taking an action $a \in A$ in state s, where $s, s' \in S$. $R(s)$ is the reward the agent receives in state $s \in S$, γ is the discount factor for future rewards, and S_0 is the initial state distribution over S. State value $V(s)$ is the value of expected return (sum of future discounted rewards) starting with state s. State-Action value $Q(s, a)$ denotes the value of taking action a in state s (following some policy π).

3.1.2 Deterministic continuous state space environments

Similar to the MDP above, in this case, the states come from a continuous space, such as R^n. Correspondingly, the transition dynamics are represented parametrically as $P(s, a, s') = f(s, a, s')$. Since the environment is deterministic, every state-action pair results in a single destination state every time.

3.1.3 RL Policies

The goal of an RL agent is to learn a policy that maximizes the expected cumulative reward. We use the following classes of RL algorithms to learn the optimal policies and value functions for our experiments:

1. *Value Iteration (VI)* [43]: VI computes an optimal state value function for an MDP. The method uses Bellman updates to converge to the optimal values.

2. *Deep Q Network (DQN)* [35]: DQN is a model-free off-policy deep RL approach that uses a deep neural network for the Q-function. It uses a batch of past experiences (replay memory) to train the agent to learn the optimal policy.

3. *Proximal Policy Optimization (PPO)* [47]: PPO is a first-order optimization based on-policy policy-gradient algorithm that uses the actor-critic approach to find the best policy. The actor model learns to take the best action in an observed state by improving upon the feedback given by the critic model - that takes state as an input, and finds a value function estimating future rewards with the help of a deep neural network. We use Generalized Advantage Estimation [46](GAE) to calculate the Advantage of taking an action in a state.

3.2 Inverse Reinforcement Learning

Inverse RL [38] is a method of extracting a reward function, given the observed, optimal behaviour in an environment. We use two methods shared in the original Inverse RL paper for the reward reconstruction attack in our experiments - IRL for finite state spaces for the Frozen Lake domain, and IRL from sampled trajectories for large state spaces for the Mountain Car domain.

3.2.1 Inverse RL in finite state spaces

In the case of an MDP environment with discrete and finite state and action spaces with known transition dynamics and a deterministic policy, we use the method of inverse RL in finite-state spaces to reconstruct the private reward function by solving a linear programming (LP) [26] formulation that makes the given policy optimal by a large margin (as compared to other sub-optimal policies). Since this is an under-constrained problem, we choose the reward with the smallest L_1-norm. The LP formulation is as follows:

$$\begin{aligned} \text{maximize} \quad & \Sigma_{i=1}^{N} \min_{a_2,\ldots,a_k} \{(P_{a_1}(i) - P_a(i))(I - \gamma P_{a_1})^{-1} R\} - \lambda ||R||_1 \\ s.t. \quad & (P_{a_1} - P_a)(I - \gamma P_{a_1})^{-1} R \geq t_i, \; \forall a \in A \setminus a_1 \\ & |R_i| \leq R_{max}, i = 1, \ldots, N \end{aligned}$$

3.2.2 Inverse RL from sampled trajectories for large state spaces

In the case of an MDP environment with a continuous state space, a discrete action space, unknown transition dynamics, and a deterministic policy's trajectories, we use the method of Inverse RL from

sampled trajectories for large (continuous) state spaces. In this method, we work with a linear function approximator as the reward function:

$$R(s) = \alpha_1\phi_1(s) + \alpha_2\phi_2(s) + \cdots + \alpha_d\phi_d(s)$$

where the "phi" functions are fixed, known, bounded basis Gaussian functions and the "alpha" coefficients are the unknown parameters that we learn. From the simulated trajectories, we estimate value functions as follows:

$$V_i^{\pi}(s_0) = \phi_i(s_0) + \gamma\phi_i(s_1) + \gamma^2\phi_i(s_2) + \dots$$
$$V^{\pi}(s_0) = \alpha_1 V_1^{\pi}(s_0) + \cdots + \alpha_d V_d^{\pi}(s_0)$$

We optimize to maximise the value difference between the optimal policy and sub-optimal policies trained iteratively on the predicted reward function as follows:

$$\begin{aligned} \text{maximize} \quad & \Sigma_{i=1}^{k}\, p(V^{\pi *}(s_0) - V_i^{\pi_i}(s_0)) \\ s.t. \quad & |\alpha_i| \leq 1,\ i = 1, \dots, d \\ & p(x) = x\ , if x \geq 0, 2x \text{ otherwise} \end{aligned}$$

The output is the optimal values for the "alpha" coefficients such that the reconstructed reward function optimally emulates the original reward function.

3.3 Differential Privacy

DP [14] is considered to be the golden standard of computational privacy. It allows us to quantify the degree of privacy achievable by a mechanism. It is built on the concept of adjacent databases. In the context of our work, the RL agents learn optimal policies by exploring the environment and taking in rewards as a feedback for their actions. We aim to iteratively make the mechanism private during training [33] [61]. Since we care about the privacy of the rewards, we say that two reward functions are adjacent if the maximum L_2 norm of their point-wise difference is upper bounded by 1.0.

Definition 1 *(ϵ, δ)-DP: A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ with domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies (ϵ, δ)-differential privacy if for any two adjacent inputs $d, d' \in \mathcal{D}$ and for any subset of outputs $\mathcal{S} \subseteq \mathcal{R}$ it holds that*

$$Pr[\mathcal{M}(d) \in \mathcal{S}] \leq e^{\epsilon} Pr[\mathcal{M}(d') \in \mathcal{S}] + \delta$$

Definition 2 *α-Rènyi Divergence: For two probability distributions P and Q defined over $\mathbf{R}$, the Rènyi divergence of order $\alpha > 1$ is*

$$D_{\alpha}(P||Q) = \frac{1}{\alpha - 1} \log_e E_{x \sim Q} \left(\frac{P(x)}{Q(x)} \right)^{\alpha}$$

where P(x) and Q(x) are the respective probability densities of P and Q at x.

An algorithm is said to have (α, ϵ) Rènyi DP [34] if for any two neighbouring databases, it holds that the Rènyi divergence (D_α) of order α between outputs of the algorithm is less than e^ϵ.

Definition 3 (α, ϵ)*-Rènyi DP: A randomized mechanism* $f : \mathcal{D} \to \mathcal{R}$ *is said to have* (ϵ)*-Rènyi differential privacy of order* α*, or* (α, ϵ)*-RDP for short, if for any adjacent* $d, d' \in \mathcal{D}$ *it holds that*

$$D_\alpha(f(d)||f(d')) \leq e^\epsilon$$

3.3.1 L_2-Sensitivity

The L_2-sensitivity of a query function $f : \mathbb{N}^{|\chi|} \to \mathbb{R}^k$ is:

$$\Delta f = \max_{x,y \in \mathbb{N}^{|\chi|}} \{ \, ||f(x) - f(y)||_2 \, \} \text{ where } ||x - y||_2 = 1$$

It is the maximum value of the L_2-norm of the difference between $f(x)$ and $f(y)$. The sensitivity of a function captures the magnitude by which a single individual's data can change the output in the worst case. This directly determines the amount of noise that needs to be added to the data to successfully mask the participation of any individual. DP noise mechanisms use sensitivity as a parameter to obtain the noise distributions. One of the key challenges with applying DP algorithms to real-world use cases is discovering the sensitivities of various queries.

3.3.2 Gaussian Mechanism

The Gaussian mechanism privatizes queries by adding noise sampled from a Gaussian distribution to it. The uni-variate Gaussian distribution $N(\mu, \sigma^2)$ with mean μ and standard deviation σ has the following probability density function:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{3.1}$$

The Gaussian Mechanism is as follows:

Definition 4 *Gaussian Mechanism: Let* $f : \chi^n \to \mathbb{R}^k$*. The Gaussian Mechanism is defined as*

$$M(X) = f(X) + (Y_1, \ldots, Y_k)$$

, where each Y_i *is an independent random variable of type* $N(0, 2\ln(1.25/\delta)\Delta_2^2/\epsilon^2)$.

The Gaussian Mechanism is (ϵ, δ)-differentially private.

3.4 Differentially Private Reinforcement Learning

We use the following private RL methods in our experiments:

1. *Bellman update DP*: In this method, noise is added locally to the Bellman update step of VI, such that it satisfies the definition of ϵ-DP [14].

2. *Rényi-DP in DL*: This is a natural relaxation of DP that we use for multiple DP methods - DQN-DP-SGD, PPO-DP-SGD, and more [1], [42]. In the case of differentially private deep learning [27], we assume that we share the entire training process (including every loss gradient update step for the techniques) publicly, (worst-case privacy guarantee). We do so by attaching a Privacy Engine to the optimizer, which adds noise to loss gradients during every training step, which ensures that the output sensitivity to perturbations in the input data is under control.

3. *Functional noise DP in Q-Learning*: In this, functional noise is iteratively added to the value function in the training process. The aim is to protect the value function [60].

Abbreviation	Full Form
DP	Differential Privacy
RL	Reinforcement Learning
MDP	Markov Decision Process
DL	Deep Learning
LP	Linear Programming
RDP	Rènyi Differential Privacy
PAC	Probably Approximately Correct
PRIL	Privacy-Aware Inverse RL
VI	Value Iteration
DQN	Deep Q Network
PPO	Proximal Policy Optimization
DP-Bellman	Private Bellman update
DP-SGD	DP-SGD optimizer + ReLU
DP-Shoe	DP-SGD optimizer + tan-h
DP-Adam	DP-Adam optimizer + ReLU
DP-FN	Private Functional Noise engine
VI-DP-Bellman	VI + DP-Bellman
DQN-DP-SGD	DQN + DP-SGD
DQN-DP-Shoe	DQN + DP-Shoe
DQN-DP-Adam	DQN + DP-Adam
DQN-DP-FN	DQN + DP-FN
PPO-DP-SGD	PPO + DP-SGD actor
PPO-DP-Shoe	PPO + DP-Shoe actor
PPO-DP-Adam	PPO + DP-Adam actor

Table 3.1 List of acronyms used

No.	Policy Class	RL Algorithm	Privacy Technique
1	VI-DP-Bellman	Value Iteration	Bellman Update DP
2	DQN-DP-SGD	Deep Q Network	DP-SGD
3	DQN-DP-Shoe	Deep Q Network	DP-Shoe
4	DQN-DP-Adam	Deep Q Network	DP-Adam
5	DQN-DP-FN	Deep Q Network	DP-FN
6	PPO-DP-SGD	Vanilla PPO	DP-SGD actor
7	PPO-DP-Shoe	Vanilla PPO	DP-Shoe actor
8	PPO-DP-Adam	Vanilla PPO	DP-Adam actor

Table 3.2 List of policy classes used for experiments

Chapter 4

PRIL: Privacy-Aware Inverse RL Analysis Framework

4.1 Reward Reconstruction Attack

We introduce a novel case of the data-reconstruction attack - the reward reconstruction attack for RL, as we wish to protect the reward function from adversaries. We assume that the adversary has knowledge of the environment and the learned private policy. Using this information, the adversary tries to reverse engineer the reward function. While many methods can be used to do so, we focus on the inverse RL technique in this body of work, as it seems to be the best tool at our disposal. Using Inverse RL, we perform the reward reconstruction attack, to determine how effective a private policy is at protecting the reward function.

4.2 Privacy-Aware Inverse RL Analysis Framework

The PRIL framework takes as input the original reward function R, an RL policy P', and a private RL policy P'' trained using the same algorithm. Using the inverse RL algorithm, we predict the reconstructed rewards, R' and R'', from P' and P'' respectively. Once we perform the reward reconstruction attack, we compute (a variety of) distances between the reconstructed reward and the original reward. It then computes the distances $d'(R', R)$ and $d''(R'', R)$, and compares them. The larger the distance, the stronger the RL policy's privacy guarantee (in protecting the reward function). We use multiple distance metrics, including - *L_1 norm, L_2 norm, L_∞ norm, and number of sign changes*.

4.3 RL Policies

Overall, we measure the performance of 8 private algorithms across three algorithm classes - *VI, DQN, and PPO*:

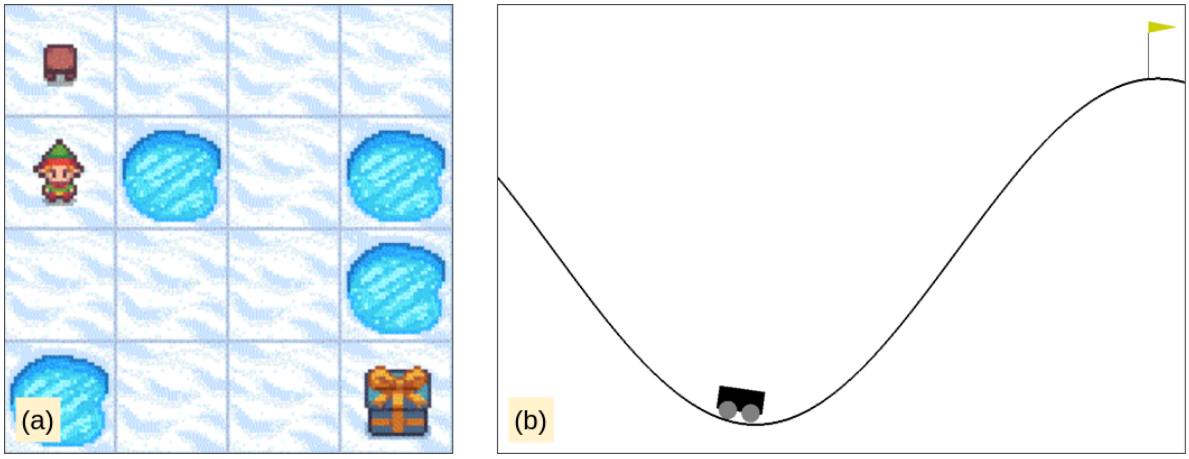

Figure 4.1 (a) Frozen Lake and (b) Mountain Car domains

For VI, we evaluate the performance of VI-DP-Bellman (private Bellman update via local DP) as well as non-private VI. Convergence threshold for value iteration $= 1.0 \times 10^{-10}$. Alternatively, we enforce the number of iterations to be less than 10000.

For DQN, we evaluate the performance of the following cases (with and without privacy):

1. *DQN-DP-SGD*: DP-SGD optimizer + ReLU activations
2. *DQN-DP-Adam*: DP-Adam optimizer + ReLU activations
3. *DQN-DP-Shoe*: DP-SGD optimizer + tan-h activations
4. *DQN-DP-FN*: DQN + functional noise

For PPO, we evaluate the performance of the following cases (with and without privacy in the actor network):

1. *PPO-DP-SGD*: DP-SGD optimizer + ReLU activations
2. *PPO-DP-Adam*: DP-Adam optimizer + ReLU activations
3. *PPO-DP-Shoe*: DP-SGD optimizer + tan-h activations

The intuition behind the DP-Shoe set of strategies comes from [42] titled "Making the shoe fit: : Architectures, initializations, and tuning for learning with privacy". Their key result states that SGD optimizer is strictly better than Adam optimizer, and that using tan-h activations are strictly better than using ReLu activations in the case of differentially private training. Hence, we named the DP-SGD optimizer with tan-h activations as the DP-Shoe method. This is also the reason why we do not perform experiments on the DP-Adam + tan-h combination.

For PPO policy class, we assume that only the actor network will be using a DP optimizer (and not the critic network). We based this on the result from [48], which shows that DP-Actor outperforms DP-Critic and DP-Both (where both the networks are made private using DP).

4.4 Metrics

4.4.0.1 Reward distance as a measure of privacy guarantee

: We used five reward distance metrics for our experiments - *absolute difference,* L_1 *distance,* L_2 *distance,* L_∞ *distance, and the sign change count*. The idea is to measure the similarity between the original reward function and the recovered reward function. This is a measure of the degree of privacy of a policy - the larger the reward distance, the more private the policy is. The metrics are calculated as follows:

1. Absolute difference: The absolute of the difference between the rewards $|R_1 - R_2|$ (used for scalar rewards).

2. L_1 distance: Normalize the rewards R, R' using L_1-norm, and then take the L_1 distance across the 2 vectors.

3. L_2 distance: Normalize the rewards R, R' using L_2-norm, and then take the L_2 distance across the 2 vectors.

4. L_∞ distance: Normalize the rewards R, R' using L_∞-norm, and then take the L_∞ distance across the 2 vectors.

5. Sign change count: Measure the number of sign changes from R to R'.

Since each distance metric is in a different space, all the distances evaluated together allow us to get a deeper insight into the reward reconstruction mechanism, and the optimality and privacy of policies.

4.4.0.2 Policy return as a measure of agent utility

: We measure how much utility (average reward return) the learned private policies achieve by observing how they perform during test-time, by calculating the average discounted returns over multiple trajectories played by the agent following the policy.

We evaluate the *privacy-utility trade-off* by simultaneously measuring the average returns of the private policy over multiple sample trajectories during test time (as shown in Figure **??**).

4.5 Domains

In this section, we introduce the domains that we perform all our experiments on - the Frozen Lake domain and the Mountain Car domain 4.1.

4.5.1 Frozen Lake

Frozen Lake is a non-deterministic discrete state and action space OpenAI Gym [9] toolkit. In all the Frozen Lake environments, the agent controls its movement and navigates in a grid-world. Additionally, the movement direction of the agent is uncertain and is only partially dependent on the direction chosen. The agent is rewarded for finding the most rewarding walkable path to the goal state. The grid-world environment has five possible states - safe (S), frozen (F), hole (H), high-reward (A) and goal (G). The agent has four possible actions - up, down, left and right. Half of the 24 environments are of a grid-size 5x5, and the remaining half are of a grid-size 10x10. The agent moves around the grid until it reaches the goal state. If it falls into the hole, it has to start from the beginning and is given a low reward. The process continues until it eventually reaches the goal state. Rewards are discounted with time at a rate of $\gamma = 0.99$.

4.5.2 Mountain Car

Mountain Car [36] is a deterministic continuous state space and discrete action space OpenAI Gym toolkit. In this environment, a car agent is placed at the bottom of a sinusoidal valley, and it's aim is to reach the top of the mountain to it's right as quickly as possible. The agent can take three actions: accelerate to the left, to the right, or not accelerate at all. The states are formulated as a tuple: (current position, current velocity). The position (x) of the car can range from $x = [-1.2, 0.6]$ and the velocity (v) ranges from $v = [-0.1, 0.1]$. The force the agent can exert is .007 and the gravity experienced is 0.0025 The starting position lies in the range of $x = [-0.6, 0.4]$, at the bottom of the valley with a velocity of 0. The top of the mountain is located at a position of $x = 0.5$, and so the goal state is defined as the position being $x \geq 0.5$ and the velocity being $v \geq 0$. The car can take at most 100 steps in an episode. Rewards are discounted with time at a rate of $\gamma = 0.99$. We propose three environments within this domain, each having a unique reward structure as follows:

1. R1: For every step, the agent's reward is -1, except when it reaches the goal-state. In that case, the agent's reward is 0.
2. R2: For every step, the agent's reward is 0, except when it reaches the goal-state. In that case, the agent's reward is 1.
3. R3: This reward function directly uses the height of the car, as calculated using the sinusoidal equation taking as input the position. $R3(x, v) = sin(3 * x) * 0.45 + 0.55$.

4.6 Experimental Setup

We will now discuss our overall experimental setup. We perform our experiments on 24 custom environments (as shown in Figure 4.2) in the *Frozen Lake domain*, and on 3 custom environments in the *Mountain Car domain*.

No.	Policy Class	l_2-Sensitivity
1	VI	1.05
2	DQN	1.0
3	PPO	1.0

Table 4.1 L_2 - Sensitivities of policy classes

No.	Policy Class	Epochs	Iterations	Episodes
1	VI	-	10000	5
2	DQN	15	200	5
3	PPO	15	200	5

Table 4.2 Frozen Lake: DL hyper-parameters for testing

4.6.1 Frozen Lake Implementation

4.6.1.1 Environments

: We build 24 custom Frozen Lake environments using the Open AI gym toolkit. Our method of creating grid-world environment maps aimed for diversity in spatial rewards and richness in reward structure. We built maps that would lead to interesting policies - while some policies would be very specific (like learning a single route), other policies would have multiple optimal routes. We also show that the absolute reward values themselves do not dictate the policy - it's their values taken relative to their surrounding reward values that influences the policy.

4.6.1.2 Inverse RL technique

: We use an LP solver to solve the objective functions of Inverse RL. We use a reference implementation [3] for finite-state space inverse RL that makes the use of cvxopt [4] to solve the LP formulation.

4.6.1.3 Private RL Policies

: We demonstrate our experiments on all 8 policy classes: VI-DP-Bellman, DQN-DP-SGD, DQN-DP-Shoe, DQN-DP-Adam, DQN-DP-FN, PPO-DP-SGD, PPO-DP-Shoe, and PPO-DP-Adam. We build the Deep RL experiments using TensorFlow 2.4, and add privacy using TensorFlow Privacy. We build VI-DP-Bellman private algorithm from scratch, and modified the publicly available code provided for the DQN-DP-FN strategy [60] for discrete state spaces.

Policy	$\epsilon = 0.1$	$\epsilon = 0.105$	$\epsilon = 0.2$	$\epsilon = 0.5$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = \infty$
VI	2080.08	1886.69	520.02	83.20	20.80	5.20	0.83	0.21	0
DQN	94229	150	22.75	9.89	5.38	3.03	1.55	1.0	0
PPO	94229	150	22.75	9.89	5.38	3.03	1.55	1.0	0

Table 4.3 Frozen Lake: Standard deviations

4.6.1.4 Training Specifications

: We use Linux OS based servers for training all the agents with a total of 8 GPUs and 8 CPUs. All experiments spanned across 9 privacy budgets, 24 environments, 8 policy classes, repeating each experiment 10 times (to account for the randomness stemming from private noise mechanisms and DL optimization). The total run-time for the entire set of experiments was 3 weeks.

4.6.1.5 DL Hyper-parameters

: In this part, we provide a detailed record of all DL hyper-parameters (such as learning rate, convergence criteria, and more). We also list the l_2 sensitivities, and Gaussian noise standard deviations used corresponding to each privacy budget ϵ, along with the formal privacy guarantees offered.

- Learning rate $= 0.15$
- Mini-batch size = 50
- Number of micro-batches = 5 (each of size 10)
- Wind factor = 0.0001, resulting in non-determinism in the environment
- For reproducibility of the experiments, we fixed a random seed = 0.

4.6.1.6 Privacy Budget Specifications

:

1. We used the Rényi-DP privacy engine from the TensorFlow Privacy library to compute standard deviations (sigmas) for our chosen ϵ budgets, with an additive δ budget of 10^{-5} (standard default).
2. We choose 9 different privacy budgets: $\epsilon \in \{0.1, 0.105, 0.2, 0.5, 1, 2, 5, 10, \infty\}$ (including the no-privacy case where $\epsilon = \infty$). The way we choose these budgets is to have a diverse order of magnitudes within a reasonable range of budget values. We include 0.1 and 0.105 on purpose - to show the increased sensitivity of the private algorithm with slight changes to the budget at lower

No.	Policy Class	Epochs	Iterations	Episodes
2	DQN	10	100	10
3	PPO	10	100	10

Table 4.4 Mountain Car: DL hyper-parameters for testing

Policy	$\epsilon = 0.1$	$\epsilon = 0.2$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 2.0$	$\epsilon = 5.0$	$\epsilon = 10.0$	$\epsilon = \infty$
DQN	151	22.5	9.8	5.35	3.04	1.54	0.989	0
PPO	151	22.5	9.8	5.35	3.04	1.54	0.989	0

Table 4.5 Mountain Car: Standard deviations

values. We observed a budget lower bound of 0.1 - adding anymore noise results in a budget of 0. We present the list of standard deviations used for each ϵ budget for each policy class in table 4.6.

3. For all the DQN and PPO policy classes, we assume an l_2-sensitivity of 1.0 - as all the private deep learning techniques allow for us to scale-down the data, to bring it to unit sensitivity. We present the list of l_2-sensitivities for each policy class in table 4.6.

4.6.1.7 Evaluation

: We evaluate the reward distance in this domain via the following metrics: L_1 distance, L_2 distance, L_∞ distance, and sign change counts across the reward vectors. We evaluate the utility by measuring average discounted returns during test-time.

4.6.2 Mountain Car Implementation

4.6.2.1 Environments

: We build 3 custom Mountain Car environments, each with a unique reward function (R1, R2, R3) using the Open AI Gym toolkit. Our method of creating environments with unique reward functions aims to understand the effect of negative rewards and discounting in the first two cases, and the effect of the height information in the third case. We show that the absolute reward values themselves do not dictate the policy - it's their values taken relative to their surrounding reward values that influences the policy.

4.6.2.2 Inverse RL technique

: We use an LP solver to solve the objective functions of Inverse RL. We implement the algorithm from scratch for Inverse RL from sampled trajectories for large (continuous) state spaces using a linear

function approximator. We make use of the cvxopt library to solve the LP formulation. We use $d = 10$ as the dimensionality for the linear approximation. We choose ϕ functions to be Gaussians with a standard deviation of $\sigma = 0.0666$, and means

$$\vec{\mu} = [-1.2,\ -1.0,\ -0.8,\ -0.6,\ -0.4,\ -0.2,\ 0.0,\ 0.2,\ 0.4,\ 0.6].$$

The goal (optimal) α vector values for each environment are as follows:

1. R1: $\vec{\alpha} = [-1, -1, -1, -1, -1, -1, -1, -1, -1, 0]$
2. R2: $\vec{\alpha} = [0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1]$
3. R3: $\vec{\alpha} = [0.749,\ 0.486,\ 0.246,\ 0.112,\ 0.131,\ 0.296,\ 0.55,\ 0.804,\ 0.969,\ 0.988]$, as generated from the original sinusoidal reward function.

For each update round, we estimate the value function by averaging it over the rewards obtained from $m = 2$ trajectories. The algorithm builds over $n = 3$ policies: it starts with a random policy, and iterates over two more policies produced sequentially.

4.6.2.3 Private RL Policies

: We demonstrate our experiments on 6 policy classes: DQN-DP-SGD, DQN-DP-Shoe, DQN-DP-Adam, PPO-DP-SGD, PPO-DP-Shoe, and PPO-DP-Adam. We use the same infrastructure as mentioned in the Frozen Lake implementation section. We use the values for noise multiplier as shared in table 4.6.1.7.

4.6.2.4 Training Specifications

: We use Linux OS based servers for training all the agents with a total of 4 GPUs and 8 CPUs. All experiments spanned across 8 privacy budgets, 3 environments, 6 policy classes, repeating each experiment 10 times (to account for the randomness stemming from private noise mechanisms and DL optimization). The total run-time for the entire set of experiments was 1 week.

4.6.2.5 DL Hyper-parameters

: Below are listed the key values of hyper-parameters used for training:

- Learning rate = 0.01
- L_2 norm clip = 1.0
- Noise multiplier = -.001
- Number of micro-batches = 4

- Epsilon at start = 0.3
- Epsilon decay rate = 0.02
- Min epsilon = 0.02
- No. of epochs of training = 10
- No. of episodes = 10
- No. of iterations = 100

4.6.2.6 Privacy Budget Specifications

:

1. We used the Rényi-DP privacy engine from the TensorFlow Privacy library to compute standard deviations (sigmas) for our chosen ϵ budgets, with an additive δ budget of 10^{-5} (standard default).
2. We choose 8 different privacy budgets: $\epsilon \in \{0.1, 0.2, 0.5, 1, 2, 5, 10, \infty\}$ (including the no-privacy case where $\epsilon = \infty$).
3. For all the DQN and PPO policy classes, we assume an l_2-sensitivity of 1.0 - as all the private deep learning techniques allow for us to scale-down the data, to bring it to unit sensitivity.

4.6.2.7 Evaluation

: We evaluate the reward distance in this domain by calculating the absolute difference of the averaged total discounted reward achieved for each policy class. We additionally evaluate the distance between the optimal and predicted *alpha* reward coefficient vectors obtained as output from the Inverse RL algorithm for sampled trajectories. We do so via the following metrics: L_1 distance, L_2 distance, L_∞ distance, and sign change counts across the reward vectors. We evaluate the utility by measuring average discounted returns during test-time.

4.6.3 VI-DP-Bellman Strategy

Value Iteration with output DP (global DP) allows us to publish only the final state values. However, if we want the entire learning process and each update to be public, we need to add noise to the Bellman update step. The key step is to add Gaussian Noise to the Bellman update step as follows:

$$V(s) = \max_a \Sigma_{s'} T(s, a, s')[r(s, a) + \gamma V(s')] + \mathcal{N}(0, \sigma) \tag{4.1}$$

Here, the σ parameter of the Gaussian noise distribution (standard deviation) is computed from the privacy budget ϵ and the sensitivity of f: δf, using the Rényi DP guarantee for Gaussian noise mechanisms. We now calculate the sensitivity of the Bellman update.

4.6.3.0.1 Sensitivity Proof The sensitivity of the state-value update via the Bellman equation is upper bounded by $(n+1)/n$, where $n = |S|$ is the number of states in the discrete environment. We consider two reward functions r, r' to be neighbouring reward functions if $||r - r'||_2 \leq 1$. For the $r(s)$ reward structure, the gain can be represented as:

The gain at time-step t from all future rewards until time-step $t + T + 1$ is:

$$G_t = \Sigma_{j=0}^{T} \gamma^j r_{t+j+1} \tag{4.2}$$

The state-value function $V_\pi(s)$ of an MDP is the expected return starting from state s, and then following policy π.

$$V_\pi(s) = E_\pi[G_t|s = s_t] = E_\pi[\Sigma_{j=0}^{T} \gamma^j r_{t+j+1}|s = s_t] \tag{4.3}$$

Bellman equation for optimal value function:

$$V(s) = \max_a \Sigma_{s'} P(s, a, s')[r(s, a) + \gamma V(s')] \tag{4.4}$$

Let us consider two Value functions V_1, V_2, each corresponding to a different reward function r_1, r_2 for the same environment such that $||r_1 - r_2||_2 \leq 1$.

$$V_1(s) = \max_a \Sigma_{s'} P(s, a, s')[r_1(s, a) + \gamma V(s')] \tag{4.5}$$

$$V_2(s) = \max_a \Sigma_{s'} P(s, a, s')[r_2(s, a) + \gamma V(s')] \tag{4.6}$$

Let the difference value function be $\nabla V = V_1 - V_2$ and reward function be $\nabla r = r_1 - r_2$.

$$\begin{aligned} \nabla V(s) = &\max_{a_1} \Sigma_{s'} P(s, a_1, s')[r_1(s, a_1) + \gamma V_1(s')] \\ &- \max_{a_2} \Sigma_{s''} P(s, a_2, s'')[r_2(s, a_2) + \gamma V_2(s'')] \end{aligned} \tag{4.7}$$

We can assume that V_1 is estimated by following policy π_1 learnt from r_1, and similarly, V_2 is estimated by following policy π_2 learnt from r_2. Then,

$$\begin{aligned} \nabla V(s) = &\Sigma_{a_1} \pi_1(a_1|s) \Sigma_{s'} P(s, a_1, s')[r_1(s, a_1) + \gamma V_1(s')] \\ &- \Sigma_{a_2} \pi_2(a_2|s) \Sigma_{s''} P(s, a_2, s'')[r_2(s, a_2) + \gamma V_2(s'')] \end{aligned} \tag{4.8}$$

Upon re-arranging the terms,

$$\begin{aligned} \nabla V(s) = \Sigma_a \Sigma_{s'} P(s, a, s')[&\pi_1(a|s)(r_1(s, a) + \gamma V_1(s')) \\ &- \pi_2(a|s)(r_2(s, a) + \gamma V_2(s'))] \end{aligned} \tag{4.9}$$

$$\begin{aligned} \nabla V(s) = \Sigma_a \Sigma_{s'} P(s, a, s')[&(\pi_1(a|s) r_1(s, a) - \pi_2(a|s) r_2(s, a)) \\ &+ (\gamma \pi_1(a|s) V_1(s') - \gamma \pi_2(a|s) V_2(s'))] \end{aligned} \tag{4.10}$$

To simplify,

$$\begin{aligned}\nabla V(s) = \Sigma_a \Sigma_{s'} P(s,a,s')[(\pi_1 r_1 - \pi_2 r_2) \\ + \gamma(\pi_1 V_1 - \pi_2 V_2)]\end{aligned} \tag{4.11}$$

We know that $0 \leq P(s,a,s'), \pi(a|s), \gamma \leq 1$. Therefore, we can always say that $\Sigma_a \pi(a|s) r(s,a) \leq \max_a r(s,a)$. In our case, we use $r(s)$, so $\max_a r(s,a) = r(s)$. Similarly, we can use the same logic to resolve $P(s,a,s')$ and γ in the expression. So, we can upper bound $\nabla V(s)$ as follows:

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'}(\max V_1 - \min V_2) \tag{4.12}$$

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'} \max(V_1 - V_2) \tag{4.13}$$

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'} \max \nabla V(s') \tag{4.14}$$

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'} ||\nabla V(s')||_\infty \tag{4.15}$$

Since l_∞-norm is always $\leq l_2$-norm,

$$\begin{aligned}\nabla V(s) &\leq (r_1 - r_2) + \Sigma_{s'} ||\nabla V(s')||_\infty \\ &\leq (r_1 - r_2) + \Sigma_{s'} ||\nabla V(s')||_2\end{aligned} \tag{4.16}$$

Taking l_2-norm of the entire equation,

$$\begin{aligned}||\nabla V(s)||_2 \leq ||((r_1 - r_2) \\ + \Sigma_{s'} ||\nabla V(s')||_2)||_2\end{aligned} \tag{4.17}$$

Let $(r_1 - r_2)(s) = \nabla r(s)$, and $\Sigma_{s'} \nabla V(s') = \nabla W$. Notice that ∇W is independent of (any) state s. Then,

$$\begin{aligned}&||((r_1 - r_2) + \Sigma_{s'} ||\nabla V(s')||_2)||_2 \\ &= ((\nabla r(s) + \nabla W)^2)^{0.5} \\ &= \nabla r(s) + \nabla W\end{aligned} \tag{4.18}$$

Thus,

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'} \nabla V(s') \tag{4.19}$$

If we sum the above term across all states,

$$\begin{aligned}\Sigma_s \nabla V(s) \leq \Sigma_s (r_1 - r_2)(s) \\ + |S| \times (\Sigma_{s'} \nabla V(s'))\end{aligned} \tag{4.20}$$

$$\begin{aligned}\Sigma_s \nabla V(s) \leq \Sigma_s (r_1 - r_2)(s) \\ + |S| \times (\Sigma_{s'} \nabla V(s'))\end{aligned} \tag{4.21}$$

Now the LHS is the same as the second term in the RHS. So, by shifting terms around,

$$\Sigma_s \nabla V(s') \leq \frac{\Sigma_{s \in S}(\nabla r)(s)}{|S-1|} \tag{4.22}$$

Considering $||\nabla r||_2 \leq 1$, we can generalize that

$$||\Sigma_{s \in S}(\nabla r)(s)||_2 \leq |S|.1$$

Therefore,

$$\Sigma_s \nabla V(s') \leq \frac{|S|}{|S-1|} \tag{4.23}$$

Or,

$$||V_1 - V_2||_2 \leq \frac{|S|}{|S-1|}$$

.

Therefore, we are able to show that the l_2-sensitivity of the Bellman Value update for Value Iteration is $\nabla V \leq \frac{|S|}{|S-1|}$, where $|S|$ is the number of states. For our domains, the largest sensitivity was $\frac{25}{24} = 1.041$, which we upper bounded with a value of 1.05.

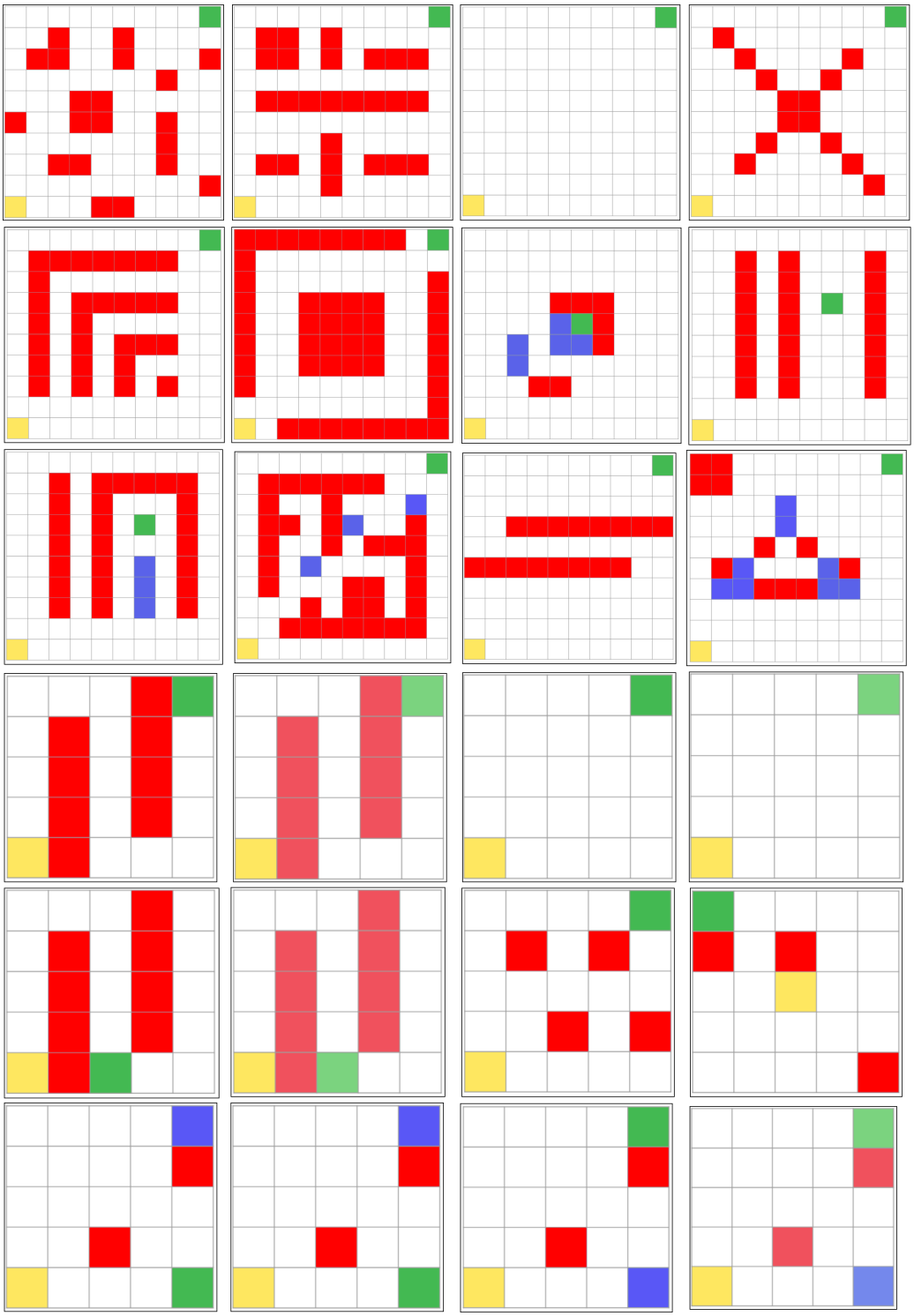

Figure 4.2 All 24 Frozen Lake environments used. Here, green: goal (G), red: frozen (F), yellow: start state, white: safe (S), and blue: high-reward (A).

Chapter 5

Results and Analysis

To the best of our knowledge, this is the first time such an attack and evaluation is conducted - hence the lack of pre-existing baselines. For each private RL algorithm, we consider the non-private version of the RL policy (privacy budget $\epsilon = \infty$) as the baseline for reward reconstruction. The more private the policy is, the larger the reward distance should be (between the original reward and the reconstructed reward).

5.0.1 Frozen Lake Results and Analysis

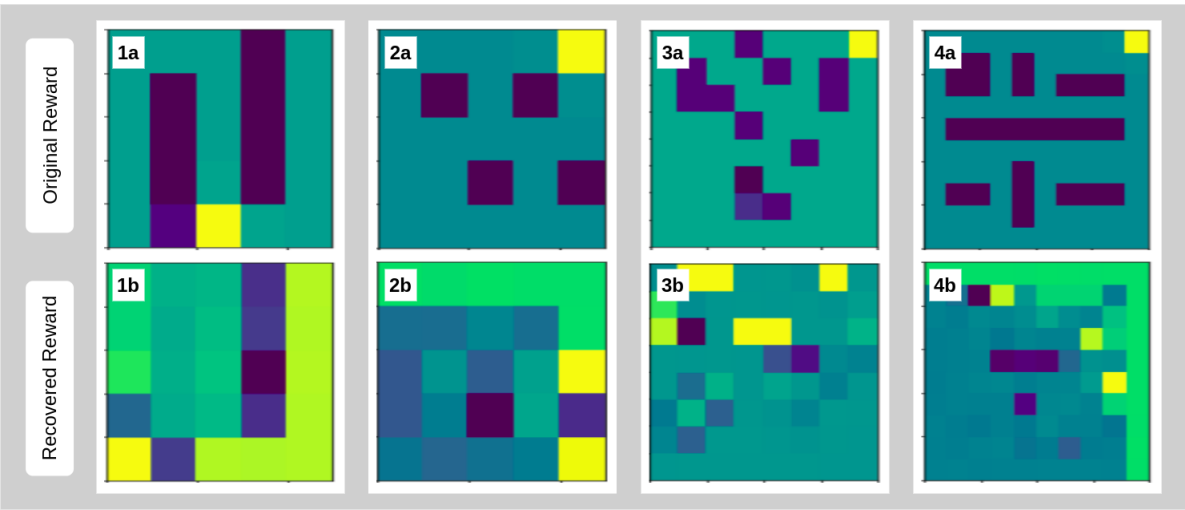

Figure 5.1 Frozen Lake: (a) Original and (b) reconstructed reward heat-maps for two 5x5 environments (1a, 2a) and for two 10x10 environments (3a, 4a), reconstructed using the DP-SGD algorithm with a budget = 0.2.

Figure 5.1 shows the heat-maps and reward structures of 4 different MDPs (row 1) and their corresponding reconstructed rewards (row 2) using the VI-DP-Bellman algorithm. From 1a and 1b we can

observe that the agent is able to clearly detect the obstacle bar on the right. From 2a and 2b we can infer that the agent finds a straight line path along the edges that is rewarding. But it also wrongly identifies some states as rewarding, even though they might be dead-ends (bright yellow bottom-right corner). Perhaps, this could be because the agent is able to survive the cost of the state directly above it, if it means that the agent can easily reach the goal state in a short amount of time. In 3a and 3b we learn that even in such a rich environment, the agent is able to reconstruct the reward structure from an implementation of the policy. It can clearly identify rewarding blocks within the entire maze. And finally, in 4a and 4b, the agent learns to stick to the edges to maximize reward early-on - which is why its evaluation of the central region is poor, and privacy preserving.

Figure 5.2 presents the variation in reward distances (y-axis) (of each type: L_1, L_2, L_∞, and sign change counts; from the original reward) with an increase in the ϵ privacy budget (x-axis) (9 discrete values) including the no-privacy case at the very end ($\epsilon = \infty$). The first row shows results averaged over the 12 Frozen Lake environments of grid size 5x5, whereas the second row shows the results corresponding to the environments of grid size 10x10. Each graph shows this relationship for all 8 private algorithms: DQN-DP-SGD, DQN-DP-Shoe, DQN-DP-Adam, DQN-DP-FN, PPO-DP-SGD, PPO-DP-Shoe, PPO-DP-Adam, and VI-DP-Bellman. The graphs show that there is no clear indication of any private strategy improving at reconstructing the reward (w.r.t. all distances) with a relaxation in the privacy budget - thus, rendering all strategies ineffective at being a truly meaningful private strategy. The reward reconstruction distance(s) are independent from the privacy budgets across all policy classes. We observe the same lack of trend across both rows, i.e., both for the 5x5 results in row 1 and 10x10 results in row 2.

For each graph in figure 5.2, the DQN-DP-FN strategy is giving straight lines. This implies that this strategy is extremely privacy agnostic in this domain. We made sure that there were no issues with the implementation itself.

Figure 5.3 presents violin plots for specific instances of PPO-DP-Shoe and VI-DP-Bellman techniques on an MDP of size 5x5 and 10x10 respectively, averaged over multiple runs. The x-axis shows all the states whereas the y-axis shows violin plots of the value of the reconstructed reward. It shows that across multiple runs of our experiments, we are able to bound the reconstructed reward's variance (for each state) by a term of the order of 10^{-19}. We observe similar results in most other cases as well. These rewards are learnt by an adversary via the PRIL framework.

Figure 5.4 presents the trade-off between the amount of utility (expected return: y-axis) and the degree of privacy (ϵ budget: x-axis) achieved by a private RL privacy. Graph 1 gives us the average trade-off for 5x5 environments, and graph 2 - for the 10x10 environments. Almost all algorithms exhibit comparable performance - with the exception of DQN-DP-FN, which performs significantly worse.

Figure 5.5 shows that DQN algorithms give us the strongest reward privacy, followed by PPO algorithms, followed by VI algorithms - that do a very poor job at protecting the reward - despite having very similar utilities (from figure 5.4).

5.0.2 Mountain Car Results and Analysis

Figure 5.6 presents the variation in reward distance (absolute reward difference) with an increase in the privacy budget for 8 discrete budget values, including the no-privacy case at the extreme right end ($\epsilon = \infty$). Graph (a) corresponds to environment with reward R1, and similarly, graph (b) - reward R2, and graph (c) - reward R3. Each graph shows this relationship for the 6 private algorithms: DQN-DP-SGD, DQN-DP-Shoe, DQN-DP-Adam, PPO-DP-SGD, PPO-DP-Shoe, and PPO-DP-Adam. The graphs show that there is no clear indication of any private strategy improving at reconstructing the reward with a relaxation in the budget, thus rendering all strategies ineffective at being a truly meaningful private strategy. The reward reconstruction distance(s) are independent from the privacy budgets across all policy classes.

Similarly, figure 5.7 presents the variation in alpha distances (of each type: L_1, L_2, L_∞ distances, and sign change counts) from the optimal alpha coefficient vectors. These graphs consistently show that there are no clearly observable trends, implying that the level of privacy has no effect on the policy learnt and the reward function reconstructed from these private policies.

Figure 5.8 presents the trade-off between the amount of utility (expected return: y-axis) and the degree of privacy (ϵ budget: x-axis) achieved by a private RL privacy, corresponding to each unique Mountain Car environment. Almost all algorithms exhibit comparable performance for each environment.

Figure 5.9 shows that even upon aggregating, the DQN and PPO classes of RL algorithms exhibit comparable performance via reward utility, neither being better than the other on the whole.

5.1 Summary

Based on the experiments performed, we can say that there is a considerable gap between the privacy provided by the existing private methods (in policy), and the level of privacy needed to protect the reward function from the inverse RL attack. We can also infer that techniques using Deep RL methods are able to learn the policy in a more general manner, as compared to non-deep methods. We address the need for better privacy techniques for RL algorithms that can effectively protect the reward function. We hope that our work inspires a deeper theoretical understanding of the limits to minimizing the gap, as well as its consequences in real-world applications. Besides thoroughly testing our code, we perform an extensive span of experiments. Contrasting our results with the baseline (no privacy), we find the reward distances to be quite similar. We therefore, believe that the source of the privacy gap is not experimental error.

Our survey of research papers that experiment on Frozen Lake shows that the commonly used grid sizes are $\{4 \times 4, 8 \times 8\}$ while we experimented with slightly larger grid sizes - $\{5 \times 5, 10 \times 10\}$. We expect to observe a similar (or worse) privacy gap upon further increase of grid size since the reward would be richer in information, and the DP sensitivity is independent of the grid size. Similarly, upon

expanding the Mountain Car domain to incorporate much richer geographies, we expect the privacy gap to only widen.

Figure 5.2 Frozen Lake: Reward distance vs privacy budget graphs for all strategies: 1,2: L_1 distance, 3,4: L_2 distance, 5,6: L_∞ distance, 7,8: Sign change counts. 1,3,5,7: averaged over 5x5 grid sized environments, 2,4,6,8: averaged over 10x10 grid sized environments

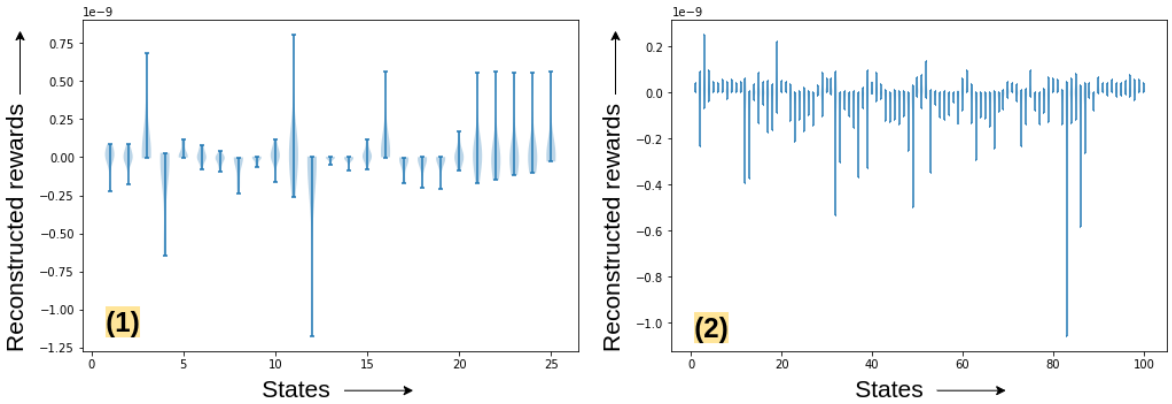

Figure 5.3 Frozen Lake: Variance Violin plots for (1): PPO-DP-Shoe on an MDP of grid-size 5x5 with a privacy budget $\epsilon = 0.5$ showing variance over 10 runs; (2): VI-DP-Bellman on an MDP of grid-size 10x10 with a privacy budget $\epsilon = 0.5$ showing variance over 20 runs

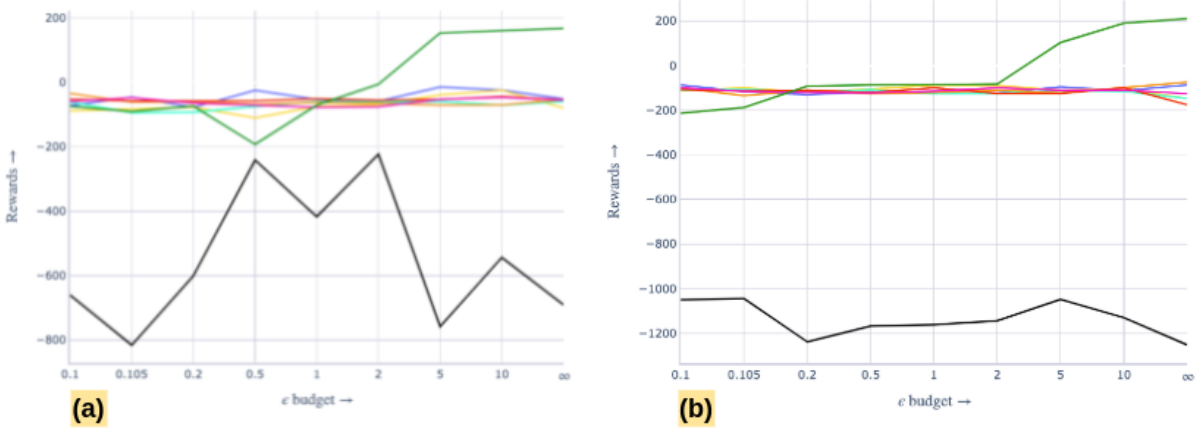

Figure 5.4 Frozen Lake: Utility (test-time return) vs privacy trade-off for all policies averaged across grid-sizes 5x5 (a) and 10x10 (b). Legend is the same as that in figure 5.2

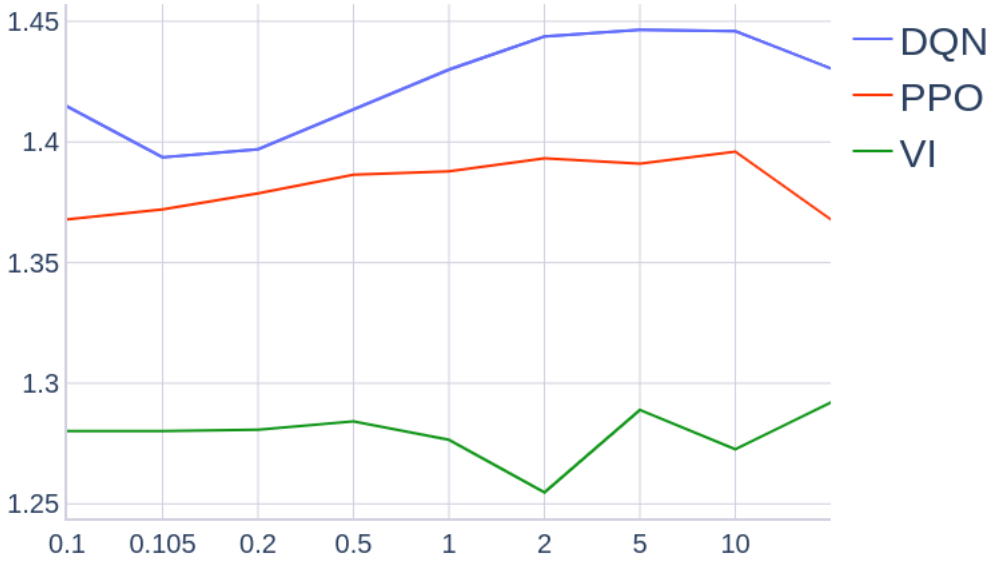

Figure 5.5 Frozen Lake: Aggregated L_2 distances across all environments and policy variants of the three main classes of RL algorithms: DQN, PPO, and VI.

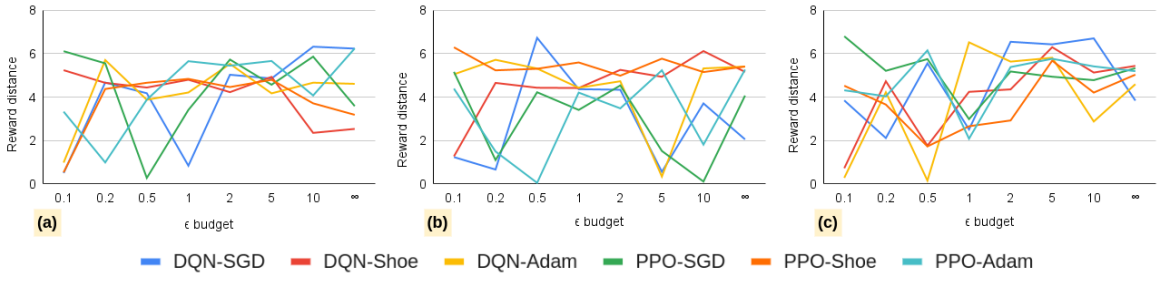

Figure 5.6 Mountain Car: Absolute reward difference (distance) vs privacy budget graphs for all policies corresponding to each unique environment (a - R1, b - R2, c - R3).

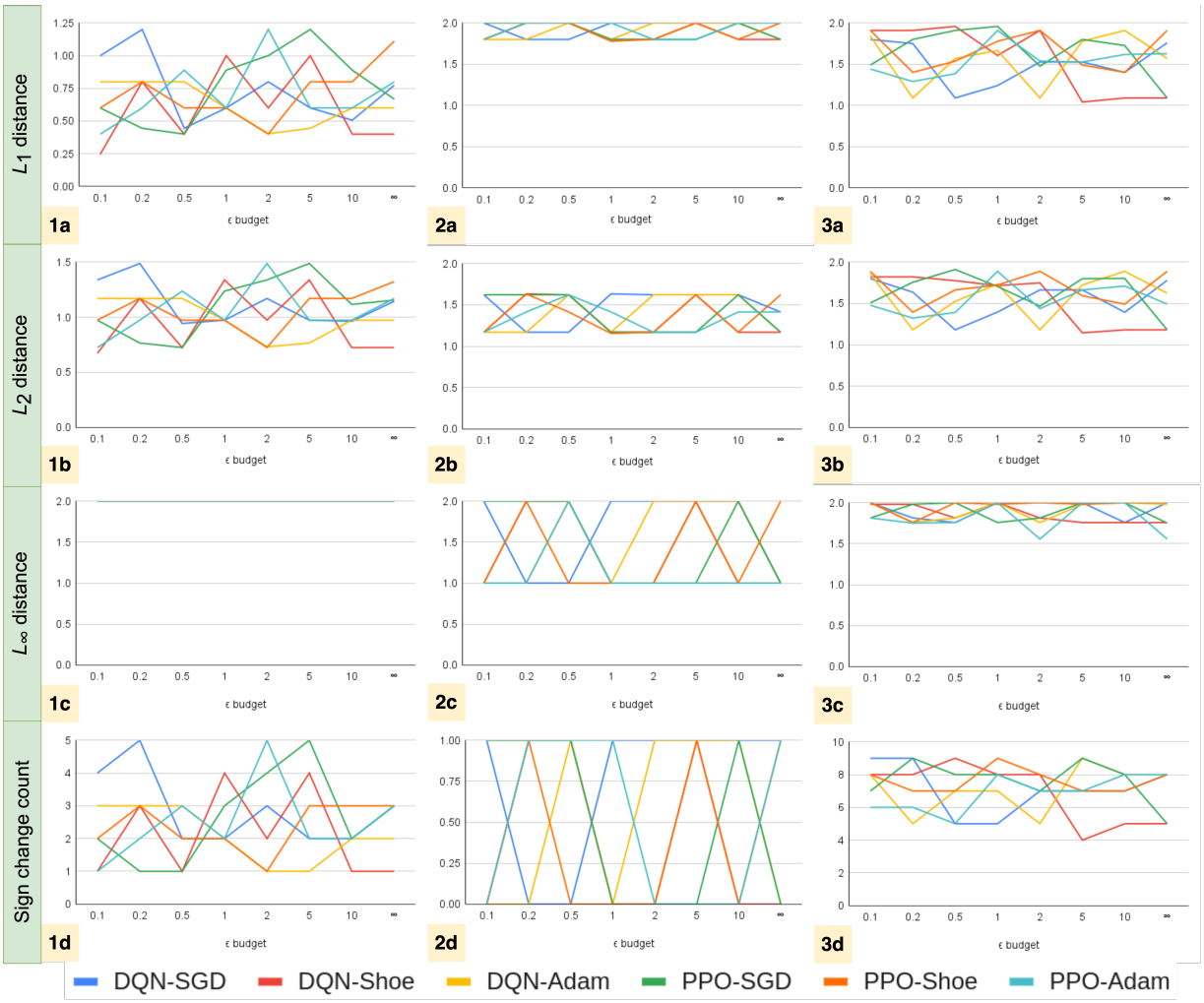

Figure 5.7 Mountain Car: Alpha distances vs privacy budget graphs for all policies: 1a, 2a, 3a: L_1 distance, 1b, 2b, 3b: L_2 distance, 1c, 2c, 3c: L_∞ distance, 1d, 2d, 3d: sign change counts, corresponding to each unique environment (column 1 - R1, column 2 - R2, column 3 - R3).

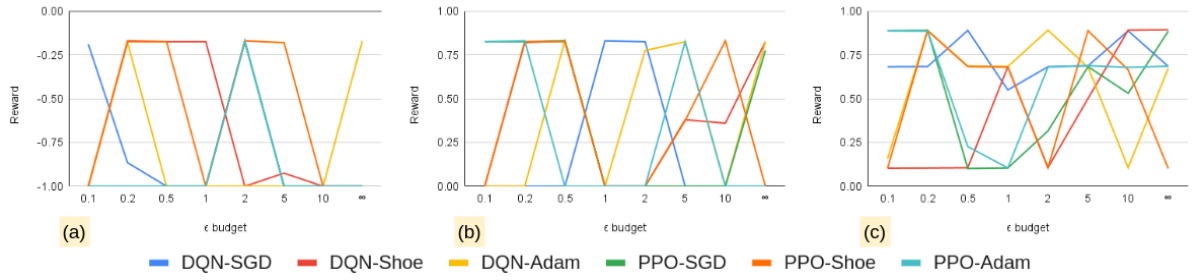

Figure 5.8 Mountain Car: Utility (test-time return) vs privacy trade-off for all policies corresponding to each unique environment (a - R1, b - R2, c - R3).

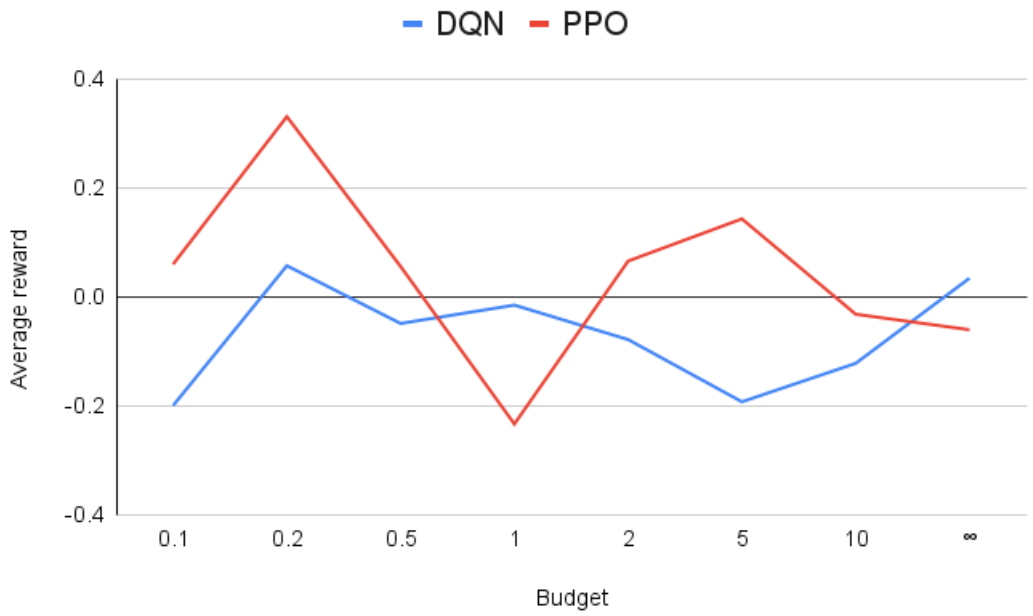

Figure 5.9 Mountain Car: Aggregated rewards across all environments and policy variants of the main classes of RL algorithms: DQN and PPO.

Chapter 6

Conclusion

This thesis investigates the nature of privacy achievable in Reinforcement Learning algorithms, and builds a class of adversarial attacks on them. The adversary's goal is to breach the privacy of the RL environment by exploiting RL policies trained on that environment, to reverse engineer the underlying sensitive reward function that withholds key information about any private environment.

We introduce the new Privacy-Aware Inverse RL analysis framework (PRIL) for enhancing reward privacy in reinforcement learning (RL) that performs a novel reward reconstruction attack and demonstrated its ability to fairly assess the level of privacy achieved in protecting the reward structure from adversarial attacks. We believe that the PRIL framework is extendable to real-world domains. We studied the set of existing privacy techniques for RL, performed a detailed evaluation of their effectiveness and identified that there is a significant gap between the current standard of privacy offered and the standard of privacy needed to protect reward functions in RL. We quantify this gap by measuring distances between the original and reconstructed rewards. We also, formally discover the private Bellman Update algorithm along with its sensitivity proof.

Meaningful user privacy - while hard to achieve perfectly in every context, is very important. This body of work helps uncover some of the more complex and nuanced privacy challenges in a data rich area of computing - reinforcement learning. This is one of the first few works in this sub-field. We urge the community to dig deeper in this direction, so that the real-world Reinforcement Learning based systems that are built in the near future are more private, safe and robust.

Chapter 7

Future Work

7.1 Scope and Future Work

While we demonstrate our work on domains from the OpenAI Gym toolkit, we believe it is extendable to real-world domains with sensitive data. Our work is the first in this direction and serves as evidence that there is a need to inspect further. Deep RL is increasingly being used for recommendation systems (RecSys) in dynamic environments. Consider the case when the recommendation engine for every user is a unique private RL policy whose job is to recommend items to users and learn their preferences in an online fashion (given the user's historical data). The reward is the user's feedback (ratings) to the recommended action. While the policy provides privacy guarantees for its training process, it can leak the user's feedback when subject to the re-identification attack via reward reconstruction. PRIL can help assess this threat better.

We surveyed a range of Inverse RL algorithms - finite state space LP, sample trajectories ([38]), deep Inverse RL ([64]), and maximum entropy Inverse RL ([68], [65]). Despite starting with the simplest cases - LP for finite state spaces and sample trajectories, we observe a significant privacy gap. These method acts as a baseline for other IRL methods. With increased complexity, the reward function would be represented parametrically which would allow the system to evaluate performance on much larger and richer (and potentially continuous) environments. As the performance of Inverse RL as an attacker improves, we expect the issue of privacy gap to become even more important to address.

Our work introduces a novel direction of evaluating the privacy guarantees of RL systems. In the future, we hope to build on our work in multiple ways: extending to the multi-agent scenario, extending to a diverse set of domains, assessing the effect of generalization and exploration on privacy [32], testing the performance of other RL algorithms such as PPO-Clip and PPO-KL, and evaluating the performance of other complex inverse RL algorithms in improving the framework.

While there are multiple privacy techniques for RL algorithms, many of them are not directly relevant for the work done in this body of work:

- We decided not to adopt the private upper confidence bound (UCB) and private deep Q learning methods from [59], as their mode of operation is completely different. They assume that every

episode of experience comes from a different private entity, whereas we focus on protecting any single reward function.

- We decided not to apply the DP-FN technique to PPO as it is not a natural extension. It would require significant deeper analysis to be able to apply DQN's private weight update to the updates to the actor network in PPO, considering that the interleaved updates to the critic network would be a source of privacy leakage.
- In the space of Inverse RL, we used the best solution for our domains. For future work, we will look at how other inverse RL techniques fare in more complex domains.

Related Publications

1. **Kritika Prakash**, Fiza Husain, Praveen Paruchuri, and Sujit Gujar, *How Private Is Your RL Policy? An Inverse RL Based Analysis Framework*[44]. (Association for the Advancement of Artificial Intelligence - AAAI 2022, **accepted** as a full paper for *oral presentation.*) Link to paper: `https://www.aaai.org/AAAI22Papers/AAAI-7524.PrakashK.pdf`. Link to public implementation: `https://github.com/magnetar-iiith/PRIL`

7.2 Other Publications

1. Aditya Gear, **Kritika Prakash**, Nonidh Singh, and Praveen Paruchuri, *PredictRV: A Prediction Based Strategy for Negotiations with Dynamically Changing Reservation Value*[18]. (International Conference on Group Decision and Negotiation - GDN 2020, **accepted** as a full paper for *oral presentation*, and was awarded the *best student paper runner-up award.*)

2. Susobhan Ghosh, **Kritika Prakash**, Sanjay Chandlekar, Easwar Subramanian, Sanjay Bhat, Sujit Gujar, Praveen Paruchuri, *Vidyut Vanika: An Autonomous Broker Agent for Smart Grid Environment*[17]. (Policy, Awareness, Sustainability and Systems Workshop - PASS 2019, **accepted** as a short paper for *oral presentation.*)

Bibliography

[1] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

[3] M. Alger. Inverse reinforcement learning, 2016.

[4] M. S. Andersen, J. Dahl, L. Vandenberghe, et al. Cvxopt: A python package for convex optimization. *Available at cvxopt. org*, 54, 2013.

[5] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath. A brief survey of deep reinforcement learning. *arXiv preprint arXiv:1708.05866*, 2017.

[6] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[7] B. Balle, M. Gomrokchi, and D. Precup. Differentially private policy evaluation. In *International Conference on Machine Learning*, pages 2130–2138. PMLR, 2016.

[8] R. Bassily, A. Smith, and A. Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 464–473. IEEE, 2014.

[9] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[10] K. Chaudhuri, C. Monteleoni, and A. D. Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(3), 2011.

[11] T. Chen, J. Liu, Y. Xiang, W. Niu, E. Tong, and Z. Han. Adversarial attack and defense in reinforcement learning-from ai security view. *Cybersecurity*, 2(1):1–22, 2019.

[12] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman. Leveraging procedural generation to benchmark reinforcement learning. *arXiv preprint arXiv:1912.01588*, 2019.

[13] K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 1282–1289. PMLR, 2019.

[14] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In S. Halevi and T. Rabin, editors, *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[15] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[16] J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[17] A. S. Gear, K. Prakash, N. Singh, and P. Paruchuri. Predictrv: A prediction based strategy for negotiations with dynamically changing reservation value. In *International Conference on Group Decision and Negotiation*, pages 135–148. Springer, 2020.

[18] S. Ghosh, K. Prakash, S. Chandlekar, E. Subramanian, S. Bhat, S. Gujar, and P. Paruchuri. Vidyutvanika: An autonomous broker agent for smart grid environment. In *Policy, Awareness, Sustainability and Systems (PASS) Workshop*, volume 7, 2019.

[19] A. Gleave, M. Dennis, C. Wild, N. Kant, S. Levine, and S. Russell. Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615*, 2019.

[20] A. Hannun, B. Knott, S. Sengupta, and L. van der Maaten. Privacy-preserving multi-party contextual bandits. *arXiv preprint arXiv:1910.05299*, 2019.

[21] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017.

[22] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 241–250, 2000.

[23] J. Ho and S. Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

[24] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

[25] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

[26] J. P. Ignizio and T. M. Cavalier. *Linear programming*. Prentice-Hall, Inc., 1994.

[27] Z. Ji, Z. C. Lipton, and C. Elkan. Differential privacy and machine learning: a survey and review. *arXiv preprint arXiv:1412.7584*, 2014.

[28] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[29] Y. Koren, S. Rendle, and R. Bell. Advances in collaborative filtering. *Recommender systems handbook*, pages 91–142, 2022.

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

[31] J. Lebensold, W. Hamilton, B. Balle, and D. Precup. Actor critic with differentially private critic. *arXiv preprint arXiv:1910.05876*, 2019.

[32] K. Lee, K. Lee, J. Shin, and H. Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. *arXiv preprint arXiv:1910.05396*, 2019.

[33] H. B. McMahan, G. Andrew, U. Erlingsson, S. Chien, I. Mironov, N. Papernot, and P. Kairouz. A general approach to adding differential privacy to iterative training procedures. *arXiv preprint arXiv:1812.06210*, 2018.

[34] I. Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

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

[36] A. W. Moore. Efficient memory-based learning for robot control. Technical report, University of Cambridge, 1990.

[37] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125. IEEE, 2008.

[38] A. Y. Ng, S. J. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[39] P. Ohm. Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA l. Rev.*, 57:1701, 2009.

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

[41] X. Pan, W. Wang, X. Zhang, B. Li, J. Yi, and D. Song. How you act tells a lot: Privacy-leakage attack on deep reinforcement learning. *arXiv preprint arXiv:1904.11082*, 2019.

[42] N. Papernot, S. Chien, S. Song, A. Thakurta, and U. Erlingsson. Making the shoe fit: Architectures, initializations, and tuning for learning with privacy. 2019.

[43] E. Pashenkova, I. Rish, and R. Dechter. Value iteration and policy iteration algorithms for markov decision problem. In *AAAI'96: Workshop on Structural Issues in Planning and Temporal Reasoning*. Citeseer, 1996.

[44] K. Prakash, F. Husain, P. Paruchuri, and S. P. Gujar. How private is your rl policy? an inverse rl based analysis framework. *arXiv preprint arXiv:2112.05495*, 2021.

[45] J. Sakuma, S. Kobayashi, and R. N. Wright. Privacy-preserving reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 864–871, 2008.

[46] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

[48] K. Seo and J. Yang. Differentially private actor and its eligibility trace. *Electronics*, 9(9):1486, 2020.

[49] N. Shlomo, C. Tudor, and P. Groom. Data swapping for protecting census tables. In *International Conference on Privacy in Statistical Databases*, pages 41–51. Springer, 2010.

[50] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[51] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[52] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[53] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 2009.

[54] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[55] R. S. Sutton, D. A. McAllester, S. P. Singh, Y. Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPs*, volume 99, pages 1057–1063. Citeseer, 1999.

[56] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[57] D. Vatsalan, P. Christen, and V. S. Verykios. A taxonomy of privacy-preserving record linkage techniques. *Information Systems*, 38(6):946–969, 2013.

[58] V. S. Verykios and A. Gkoulalas-Divanis. A survey of association rule hiding methods for privacy. In *Privacy-preserving data mining*, pages 267–289. Springer, 2008.

[59] G. Vietri, B. Balle, A. Krishnamurthy, and S. Wu. Private reinforcement learning with pac and regret guarantees. In *International Conference on Machine Learning*, pages 9754–9764. PMLR, 2020.

[60] B. Wang and N. Hegde. Privacy-preserving q-learning with functional noise in continuous state spaces. *arXiv preprint arXiv:1901.10634*, 2019.

[61] Y. Wang, H. Sibai, S. Mitra, and G. E. Dullerud. Differential privacy for sequential algorithms. *arXiv preprint arXiv:2004.00275*, 2020.

[62] S. L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[63] X. Wu, F. Li, A. Kumar, K. Chaudhuri, S. Jha, and J. Naughton. Bolt-on differential privacy for scalable stochastic gradient descent-based analytics. In *Proceedings of the 2017 ACM International Conference on Management of Data*, pages 1307–1322, 2017.

[64] M. Wulfmeier, P. Ondruska, and I. Posner. Deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

[65] M. Wulfmeier, P. Ondruska, and I. Posner. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

[66] T. Xie, P. S. Thomas, and G. Miklau. Privacy preserving off-policy evaluation. *arXiv preprint arXiv:1902.00174*, 2019.

[67] S. Zhifei and E. M. Joo. A survey of inverse reinforcement learning techniques. *International Journal of Intelligent Computing and Cybernetics*, 2012.

[68] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.