How Private Is Your RL Policy? An Inverse RL Based Analysis Framework

Kritika Prakash
Machine Learning Lab
IIIT Hyderabad
kritika.prakash@research.iiit.ac.in

Fiza Husain
Machine Learning Lab
IIIT Hyderabad
fiza.husain@students.iiit.ac.in

Praveen Paruchuri
Machine Learning Lab
IIIT Hyderabad
praveen.p@iiit.ac.in

Sujit P. Gujar
Machine Learning Lab
IIIT Hyderabad
sujit.gujar@iiit.ac.in

Abstract

Reinforcement Learning (RL) enables agents to learn how to perform various tasks from scratch. In domains like autonomous driving, recommendation systems and more, optimal RL policies learned could cause a privacy breach if the policies memorize any part of the private reward. We study the set of existing differentially-private RL policies derived from various RL algorithms such as Value Iteration, Deep Q Networks, and Vanilla Proximal Policy Optimization. We propose a new Privacy-Aware Inverse RL (PRIL) analysis framework, that performs reward reconstruction as an adversarial attack on private policies that the agents may deploy. For this, we introduce the 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 this framework, we empirically test the effectiveness of the privacy guarantee offered by the private algorithms on multiple instances of the FrozenLake domain of varying complexities. Based on the analysis performed, we infer a gap between the current standard of privacy offered and the standard of privacy needed to protect reward functions in RL. We do so by quantifying the extent to which each private policy protects the reward function by measuring distances between the original and reconstructed rewards.

1 Introduction

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) [7] to make the agent's policy quantifiably private and to get a rigorous privacy guarantee. Like many other areas in 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 in
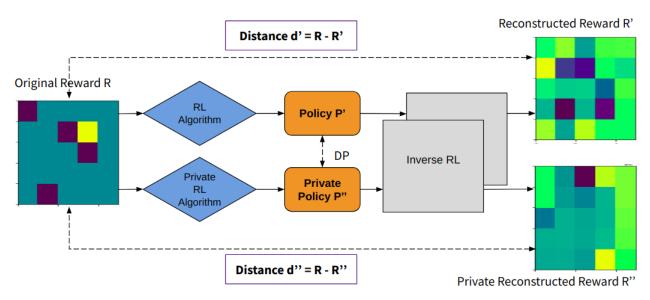
Figure 1: PRIL: Privacy-Aware Inverse RL analysis framework

achieving meaningful privacy? What role does the type of reward function, algorithms (both RL and DP algorithms), and environment play? In other words, does privacy in policy translate well to privacy in reward? To investigate this, we first build an autonomous agent whose aim is to learn a private policy using existing privacy techniques. The intent is to reach a goal state that helps maximize the agent's expected reward in discrete finite-state environments. We then investigate the true level of reward privacy offered by the existing state-of-the-art privacy techniques for RL algorithms.

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 [17] 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 applied to a set of RL algorithms:

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

We build privacy into the agent 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.

In summary, our key contributions are:

1. We study and analyze the existing set of privacy techniques for RL.
2. We introduce a novel reward reconstruction attack and 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.

The next section reviews related work. Section 3 provides background on RL, DP, and inverse RL. Section 4 explains the PRIL framework. Section 5 describes the experimental pipeline and setup. Section 6 provides an empirical analysis and discussion on our findings. Section 7 concludes the work. Deferred proofs and scope appears in the appendix. All source code and experiments are made publicly available[1].

2 Related Work

Previous works on privacy-preserving RL make the use of DP. [24] shows how to achieve joint-DP in episodic-RL via the upper confidence bound and Q-learning algorithms, where each training episode comes from a different environment. [25] makes the use of functional noise to make Q-learning private. [24] gives us probably-approximately correct (PAC) and regret guarantees for private RL. [4] propose differentially private policy evaluation using the Monte-Carlo algorithm. [10] introduces a private way of using multi-party contextual bandits. For the actor-critic class of algorithms, [14] presents a differentially private critic, and [23] presents a differentially private actor.

In RL, a significant amount of work has been done to perform adversarial attacks that target the quality of the learned policy ([9], [11], [13], [6]). Building on this, there has been work to make RL policies robust to such attacks [18]. However, very few works have looked into privacy attacks in RL. [19] shows that agents can memorize the environment and its private transition dynamics, by performing privacy attacks using genetic algorithms and candidate inference. [8] introduce an adversarial inverse RL algorithm based on an adversarial reward learning formulation to improve robustness.

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, linkage attack, 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.

3 Background and Preliminaries

In this section, we introduce the basics of RL, inverse RL, and DP.

3.1 Reinforcement Learning

We focus on non-deterministic environments with discrete and finite state and action spaces. 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). 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 policy and value function for our experiments:

1. VI [21]: VI computes an optimal state value function for an MDP. The method uses Bellman updates to converge to the optimal values.

2. DQN [16]: DQN is a model-free off-policy deep RL approach that uses a deep neural network for the Q-function which uses a batch of past experiences (replay memory) to train the agent to learn the optimal policy.

3. PPO [22]: PPO is a first-order optimization based policy-gradient algorithm that uses the actor-critic approach to find the best policy. The actor model learns to take an 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.

---

[1]Link to code: `https://github.com/magnetar-iiith/PRIL`

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	Private SGD optimizer + ReLU
DP-Shoe	Private SGD optimizer + tan-h
DP-Adam	Private 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 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 2: List of policy classes used for experiments

3.2 Inverse RL

Inverse RL [17] is a method of extracting a reward function, given the observed, optimal behaviour in an environment. We use the method of inverse RL in finite-state spaces to reconstruct the private reward function by solving a linear programming (LP) [12] 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.

4

3.3 Differential Privacy

DP [7] 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. Since we care about the privacy of the rewards, we say that two reward functions are adjacent if the maximum $L2$ norm of their point-wise difference is upper bounded by 1.

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 [15] 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*

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

3.4 Private RL Methods

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 [7].

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], [20].

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 [25].

We assume that we share the entire training process including every loss gradient update publicly, (worst-case privacy guarantee).

4 PRIL: Privacy-Aware Inverse RL Analysis Framework

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 paper, 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. It does so by computing (a variety of) distances between the reconstructed reward and the original reward. The 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, it predicts the reconstructed rewards, R' and R'', from P' and P'' respectively. 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, such as - L_1 *norm,* L_2 *norm,* L_∞ *norm, and number of sign changes.*

5 Experimental Setup

We will now discuss our overall experimental pipeline and setup. We perform our experiments on 24 custom environments (as shown in Figure 2) in the *FrozenLake domain* - a discrete-state OpenAI Gym [5] toolkit. In all these
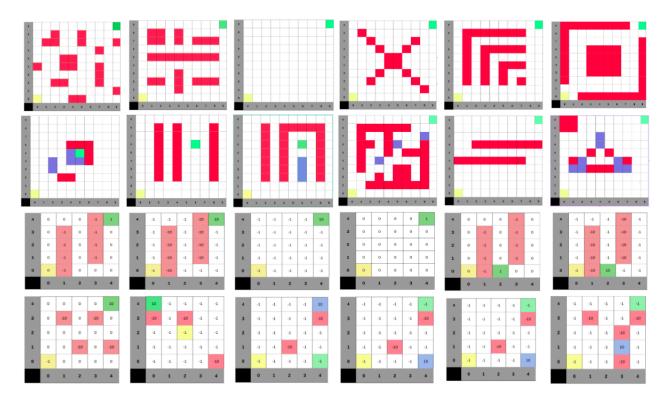
Figure 2: All 24 FrozenLake environments used. Here, green: goal (G), red: frozen (F), yellow: start state, white: safe (S), and blue: high-reward (A).

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

For VI, we evaluate the performance of VI-DP-Bellman (private Bellman update via local DP) as well as non-private VI.

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

We evaluate the *privacy-utility trade-off* by simultaneously measuring the average returns of the the private policy over multiple sample trajectories during test time (as shown in Figure 4). 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).
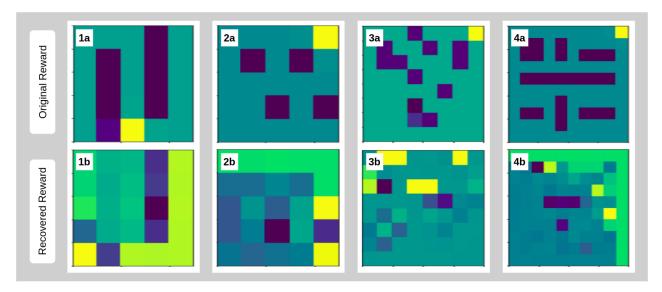
Figure 3: Original and reconstructed reward heatmaps for two 5x5 FrozenLake environments (1a, 2a) and for two 10x10 FrozenLake environments (3a, 4a)

Reward distance as a measure of privacy guarantee: We used four reward distance metrics for our experiments - 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. L_1 distance: Normalize the rewards R, R' using L_1-norm, and then take the L_1 distance across the 2 vectors.
2. L_2 distance: Normalize the rewards R, R' using L_2-norm, and then take the L_2 distance across the 2 vectors.
3. L_∞ distance: Normalize the rewards R, R' using L_∞-norm, and then take the L_∞ distance across the 2 vectors.
4. 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.

Policy return as a measure of agent utility: We measure how much utility 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.

Implementation details: We build 24 custom FrozenLake environments using the Open AI gym toolkit. We use an LP solver to solve the objective functions of Inverse RL. 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 use the publicly available code provided for the DQN-DP-FN strategy [25]. We use a reference implementation [2] for finite-state space inverse RL that makes the use of cvxopt [3] to solve the LP formulation. We use Linux OS based servers for training all the RL 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 runtime for the entire set of experiments was 3 weeks.

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. We evaluate each algorithm against the baseline case of no privacy ($\epsilon = \infty$).

Hyper-parameters and assumptions:

- We choose 9 different privacy budgets: - $\epsilon \in \{0.1, 0.105, 0.2, 0.5, 1.0, 2.0, 5.0, 10.0, \infty\}$ (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 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 5.
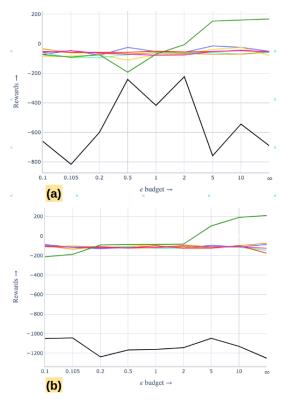
Figure 4: Utility (test-time return) vs privacy trade-off for all policies averaged across grid-sizes 5x5 (a) and 10x10 (b). Legend is the same as that in figure 5.
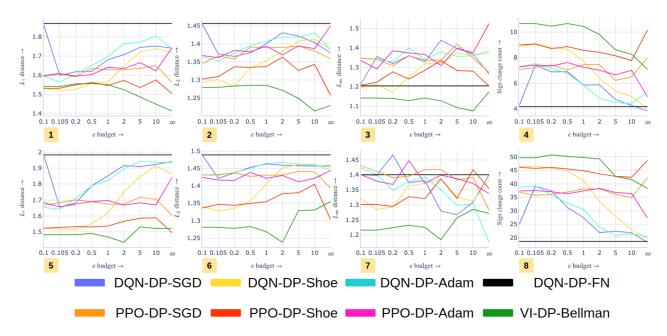

Figure 5: Reward distance vs privacy budget graphs for all strategies: 1,5: L_1 distance, 2,6: L_2 distance, 3,7: L_∞ distance, 4,8: Sign change counts. 1,2,3,4: averaged over 5x5 grid sized environments, 5,6,7,8: averaged over 10x10 grid sized environments

- The intuition behind the DP-Shoe set of strategies comes from [20] titled "Making the shoe fit: : Architectures, initializations, and tuning for learning with privacy". Their key result states that SGD optimizer is strictly
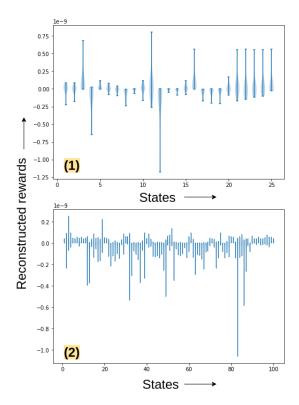
Figure 6: Variance Violin plots for (1): PPO-DP-Shoe on an MDP of grid-size 5x5 with a privacy budget $\epsilon = 0.5$ showing variance over 10 runs; (2): VI-DP-Bellman on an MDP of grid-size 10x10 with a privacy budget $\epsilon = 0.5$ showing variance over 20 runs

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

Table 3: l_2 - Sensitivities of policy classes

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 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 3.

- 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 [23], which shows that DP-Actor outperforms DP-Critic and DP-Both (where both the networks are made private using DP).

- Learning rate $= 0.15$

- Mini-batch size = 50

- Number of micro-batches = 5 (each of size 10)

- Discount factor $\gamma = 0.99$

- Wind factor = 0.0001

- For reproducibility of the experiments, we fixed a random seed and in our case we used seed = 0.

9

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

Table 4: DL hyper-parameters for testing

- Convergence threshold for value iteration $= 1.0 \times 10^{-10}$. Alternatively, we enforce the number of iterations to be less than 10000.

- For each graph in figure 5, 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.

- 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.

- 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).

6 Analysis and Discussion

Figure 5 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 FrozenLake 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 (wrt 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.

Figure 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.
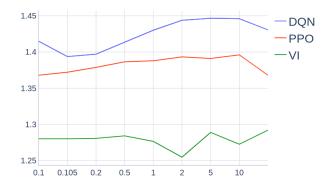

Figure 7: Aggregated L_2 distances across all environments and policy variants of the three main classes of RL algorithms: DQN, PPO, and VI.

Figure 3 shows the heatmaps 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 7 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 4).

Figure 6 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.

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 papers that experiment on FrozenLake 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.

While we demonstrate our work on a grid-world domain, we believe it is extendible 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. PRILcan help assess this threat better.

We surveyed a range of Inverse RL (IRL) algorithms - finite state space LP, sample trajectories ([17]), deep IRL ([26]), and maximum entropy IRL ([28], [27]). Despite starting with the simplest case - LP for finite state spaces, we observe a significant privacy gap. The LP 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 IRL 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, 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.

7   Conclusion

This paper introduces a 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 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

Policy	$\epsilon = 0.1$	$\epsilon = 0.105$	$\epsilon = 0.2$	$\epsilon = 0.5$	$\epsilon = 1.0$	$\epsilon = 2.0$	$\epsilon = 5.0$	$\epsilon = 10.0$	$\epsilon = \infty$
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 5: Standard deviations

functions in RL. We quantify this gap by measuring distances between the original and reconstructed rewards via the reward reconstruction attack.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] Matthew Alger. Inverse reinforcement learning, 2016.

[3] Martin S Andersen, Joachim Dahl, Lieven Vandenberghe, et al. Cvxopt: A python package for convex optimization. *Available at cvxopt. org*, 54, 2013.

[4] Borja Balle, Maziar Gomrokchi, and Doina Precup. Differentially private policy evaluation. In *International Conference on Machine Learning*, pages 2130–2138. PMLR, 2016.

[5] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[6] Tong Chen, Jiqiang Liu, Yingxiao Xiang, Wenjia Niu, Endong Tong, and Zhen Han. Adversarial attack and defense in reinforcement learning-from ai security view. *Cybersecurity*, 2(1):1–22, 2019.

[7] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, pages 265–284, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[8] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[9] Adam Gleave, Michael Dennis, Cody Wild, Neel Kant, Sergey Levine, and Stuart Russell. Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615*, 2019.

[10] Awni Hannun, Brian Knott, Shubho Sengupta, and Laurens van der Maaten. Privacy-preserving multi-party contextual bandits. *arXiv preprint arXiv:1910.05299*, 2019.

[11] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

[12] James P Ignizio and Tom M Cavalier. *Linear programming*. Prentice-Hall, Inc., 1994.

[13] Jernej Kos and Dawn Song. Delving into adversarial attacks on deep policies. *arXiv preprint arXiv:1705.06452*, 2017.

[14] Jonathan Lebensold, William Hamilton, Borja Balle, and Doina Precup. Actor critic with differentially private critic. *arXiv preprint arXiv:1910.05876*, 2019.

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

[16] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[17] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[18] Tuomas Oikarinen, Tsui-Wei Weng, and Luca Daniel. Robust deep reinforcement learning through adversarial loss. *arXiv preprint arXiv:2008.01976*, 2020.

[19] Xinlei Pan, Weiyao Wang, Xiaoshuai Zhang, Bo Li, Jinfeng Yi, and Dawn Song. How you act tells a lot: Privacy-leakage attack on deep reinforcement learning. *arXiv preprint arXiv:1904.11082*, 2019.

[20] Nicolas Papernot, Steve Chien, Shuang Song, Abhradeep Thakurta, and Ulfar Erlingsson. Making the shoe fit: Architectures, initializations, and tuning for learning with privacy. 2019.

[21] Elena Pashenkova, Irina Rish, and Rina 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.

[22] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[23] Kanghyeon Seo and Jihoon Yang. Differentially private actor and its eligibility trace. *Electronics*, 9(9):1486, 2020.

[24] Giuseppe Vietri, Borja Balle, Akshay Krishnamurthy, and Steven Wu. Private reinforcement learning with pac and regret guarantees. In *International Conference on Machine Learning*, pages 9754–9764. PMLR, 2020.

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

[26] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

[27] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

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

A   Appendix

Here we will present the scope of our work along with the sensitivity proof of our VI-DP-Bellman strategy.

A.1   Scope

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

- We decided not to adopt the private upper confidence bound (UCB) and private deep Q learning methods from [24], 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 domain - where we have access to the transition dynamics for a finite state space. For future work, we will look at how other inverse RL techniques fare in large state space domains, or in domains where we do not have access to the transition dynamics.

A.2   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:

$$V(s) = \max_a \Sigma_{s'} T(s, a, s')[r(s, a) + \gamma V(s')] + \mathcal{N}(0, \sigma) \tag{1}$$

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.

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:

$$G_t = \Sigma_{j=0}^{T} \gamma^j r_{t+j+1} \tag{2}$$

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

$$V_\pi(s) = E_\pi[G_t|s = s_t] = E_\pi[\Sigma_{j=0}^{T} \gamma^j r_{t+j+1}|s = s_t] \tag{3}$$

Bellman equation for optimal value function:

$$V(s) = \max_a \Sigma_{s'} P(s, a, s')[r(s, a) + \gamma V(s')] \tag{4}$$

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 $||r_1 - r_2||_2 \leq 1$.

$$V_1(s) = \max_a \Sigma_{s'} P(s, a, s')[r_1(s, a) + \gamma V(s')] \tag{5}$$

$$V_2(s) = \max_a \Sigma_{s'} P(s, a, s')[r_2(s, a) + \gamma V(s')] \tag{6}$$

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

$$\nabla V(s) = \max_{a_1} \Sigma_{s'} P(s, a_1, s')[r_1(s, a_1) + \gamma V_1(s')]$$
$$- \max_{a_2} \Sigma_{s''} P(s, a_2, s'')[r_2(s, a_2) + \gamma V_2(s'')] \tag{7}$$

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,

$$\nabla V(s) = \Sigma_{a_1} \pi_1(a_1|s) \Sigma_{s'} P(s, a_1, s')[r_1(s, a_1) + \gamma V_1(s')]$$
$$- \Sigma_{a_2} \pi_2(a_2|s) \Sigma_{s''} P(s, a_2, s'')[r_2(s, a_2) + \gamma V_2(s'')] \tag{8}$$

Upon re-arranging the terms,

$$\nabla V(s) = \Sigma_a \Sigma_{s'} P(s, a, s')[\pi_1(a|s)(r_1(s, a) + \gamma V_1(s'))$$
$$- \pi_2(a|s)(r_2(s, a) + \gamma V_2(s'))] \tag{9}$$

$$\nabla V(s) = \Sigma_a \Sigma_{s'} P(s, a, s')[(\pi_1(a|s)r_1(s, a) - \pi_2(a|s)r_2(s, a))$$
$$+ (\gamma \pi_1(a|s)V_1(s') - \gamma \pi_2(a|s)V_2(s'))] \tag{10}$$

To simplify,

$$\nabla V(s) = \Sigma_a \Sigma_{s'} P(s, a, s')[(\pi_1 r_1 - \pi_2 r_2)$$
$$+ \gamma(\pi_1 V_1 - \pi_2 V_2)] \tag{11}$$

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:

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'}(\max V_1 - \min V_2) \tag{12}$$

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'} \max(V_1 - V_2) \tag{13}$$

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'} \max \nabla V(s') \tag{14}$$

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'} ||\nabla V(s')||_\infty \tag{15}$$

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

$$\nabla V(s) \leq (r_1 - r_2) + \Sigma_{s'} ||\nabla V(s')||_\infty$$
$$\leq (r_1 - r_2) + \Sigma_{s'} ||\nabla V(s')||_2 \tag{16}$$

Taking l_2-norm of the entire equation,

$$||\nabla V(s)||_2 \leq ||((r_1 - r_2)$$
$$+ \Sigma_{s'} ||\nabla V(s')||_2)||_2 \tag{17}$$

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,

$$||((r_1 - r_2) + \Sigma_{s'} ||\nabla V(s')||_2)||_2$$
$$= ((\nabla r(s) + \nabla W)^2)^{0.5} \tag{18}$$
$$= \nabla r(s) + \nabla W$$

Thus,

$$\nabla V(s) \leq (r_1 - r_2)(s) + \Sigma_{s'} \nabla V(s') \tag{19}$$

If we sum the above term across all states,

$$\Sigma_s \nabla V(s) \leq \Sigma_s (r_1 - r_2)(s)$$
$$+ |S| \times (\Sigma_{s'} \nabla V(s')) \tag{20}$$

$$\Sigma_s \nabla V(s) \leq \Sigma_s (r_1 - r_2)(s)$$
$$+ |S| \times (\Sigma_{s'} \nabla V(s')) \tag{21}$$

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

$$\Sigma_s \nabla V(s') \leq \frac{\Sigma_{s \in S} (\nabla r)(s)}{|S - 1|} \tag{22}$$

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

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

Therefore,

$$\Sigma_s \nabla V(s') \leq \frac{|S|}{|S - 1|} \tag{23}$$

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.