Can In-Context Reinforcement Learning Recover From Reward Poisoning Attacks?

Paulius Sasnauskas[*]
MPI-SWS
psasnaus@mpi-sws.org

Yiğit Yalın
MPI-SWS
yyalin@mpi-sws.org

Goran Radanović
MPI-SWS
gradanovic@mpi-sws.org

Abstract

We study the corruption-robustness of in-context reinforcement learning (ICRL), focusing on the Decision-Pretrained Transformer (DPT, Lee et al., 2023). To address the challenge of reward poisoning attacks targeting the DPT, we propose a novel adversarial training framework, called Adversarially Trained Decision-Pretrained Transformer (AT-DPT). Our method simultaneously trains an attacker to minimize the true reward of the DPT by poisoning environment rewards, and a DPT model to infer optimal actions from the poisoned data. We evaluate the effectiveness of our approach against standard bandit algorithms, including robust baselines designed to handle reward contamination. Our results show that the proposed method significantly outperforms these baselines in bandit settings, under a learned attacker. We additionally evaluate AT-DPT on an adaptive attacker, and observe similar results. Furthermore, we extend our evaluation to the MDP setting, confirming that the robustness observed in bandit scenarios generalizes to more complex environments.

1 Introduction

Recent years have shown the impressive capabilities of transformer-based models on a range of tasks (Vaswani et al., 2017; Raffel et al., 2020). The community has been shifting from single-task learning, to multi-task learning, and even multi-domain learning (Reed et al., 2022). This has been made possible in part due to in-context learning, also called few-shot learning (Brown et al., 2020), which allows a model to adapt to new tasks simply by reading a handful of examples in the prompt, rather than requiring parameter updates or retraining. Recently, transformers and in-context learning have found growing use in decision-making tasks, particularly in reinforcement learning (RL), where interactions with the environment replace traditional text-based examples (Chen et al., 2021a; Xu et al., 2022; Laskin et al., 2022; Lee et al., 2023). In this paper, we focus on the robustness of in-context RL to reward poisoning attacks – one of the major security threats for safe deployment of RL agents.

Reward poisoning attacks have been extensively explored in recent RL literature (Lin et al., 2017; Ma et al., 2019; Zhang et al., 2020b; Wu et al., 2023; Nika et al., 2023). This line of work predominantly focuses on the canonical RL setting, modeling reward poisoning attacks as an attacker that corrupts the reward of a learning agent during training. In contrast to *test-time* adversarial attacks, poisoning attacks influence the policy that the agent adopts at test time; i.e., they are *training-time* attacks. This is perhaps not surprising, given that this line of work typically focuses on Markov stationary policies, implying that the agent's behavior is independent of the rewards at *test time*.

However, an in-context RL agent can implement a *learning algorithm* in-context, using approaches such as Algorithm Distillation (Laskin et al., 2022) or the Decision-Pretrained Transformer (Lee

[*]This work was done as a part of an internship project at the Max Planck Institute for Software Systems.

Preprint.

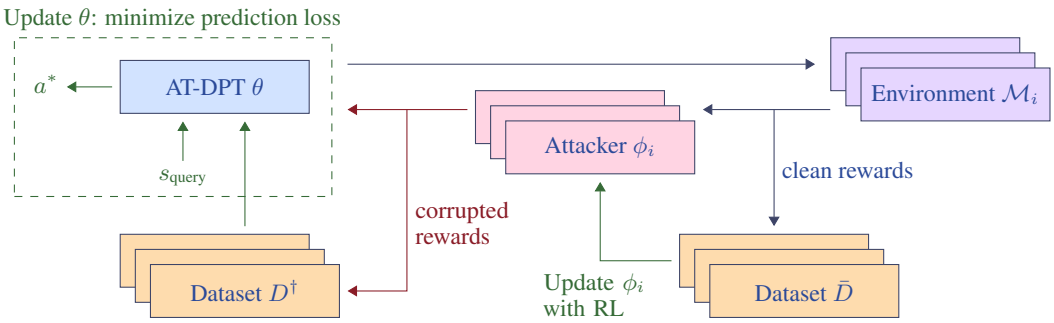

Figure 1: The training procedure of AT-DPT. We use adversarial training to optimize the parameters θ of a transformer model, which is our learning agent. In each round, we first collect data by deploying the agent in m environments. In environment $i \in 0, \ldots, m$, the agent observes rewards corrupted by an adversary defined by parameters ϕ_i. We collect clean ($\bar{D}$) and corrupted ($D^\dagger$) datasets containing trajectories with clean and corrupted rewards, respectively. The agent is trained to predict an optimal action for a query state given a context sampled from a corrupted dataset. The adversary is trained to minimize the agent's return under a soft budget constraint, expressed as a penalty term. During the test phase, the agent is deployed in a new corrupted environment.

et al., 2023). In this case, contextual information encodes past interactions between the environment and the agent, including past rewards. By corrupting the agent's rewards at test time, an adversary can still influence the agent's behavior. Simply put, such test-time reward poisoning schemes attack the learning algorithm implemented in-context.

In this work, we aim to develop a training protocol for in-context RL that enables models to be robust against test-time reward poisoning attacks targeting in-context learners. We focus on using the Decision-Pretrained Transformer (DPT) as a base approach. At a high level, the novel training protocol should implement a corruption-robust learning algorithm in-context. This differs from corruption-robust RL approaches typically studied in the literature (Lin et al., 2017; Ma et al., 2019; Zhang et al., 2020b; Sun et al., 2021; Wu et al., 2023; Nika et al., 2023): in our setting, rewards are corrupted only at test time, meaning the corruption does not affect the training process of the agent's policy. Our contributions are as follows.

Framework. We introduce and formalize a novel attack modality predicated on reward poisoning. In this attack modality, the attacker influences the context that the agent's policy is conditioned on by corrupting the agent's rewards. More specifically, the attacker aims to minimize the agent's return by perturbing the underlying reward function under a soft budget constraint encoded via a penalty term.

Method. We combine in-context learning with adversarial training to develop an agent that is robust against reward poisoning. Specifically, we introduce the Adversarially Trained Decision-Pretrained Transformer (AT-DPT), which is trained by simultaneous optimization: an adversary tries to minimize the environment's reward, while DPT learns to infer optimal actions from the corrupted data. An overview of the training procedure can be seen in Figure 1.

Experiments. We conduct a systematic evaluation of the proposed method, comparing its corruption-robustness capabilities to various baselines, including robust baselines designed to handle reward contamination, under various levels of poisoning. Our results show that AT-DPT can recover from a wide range of reward poisoning attacks and overall yields better performance under corruption than the baselines considered.

We believe that the results presented in the work show potential of transformer-based policies in implementing algorithms that are robust against data contamination.

2 Related work

Adversarial ML. Adversarial ML is a line of work with the goal of understanding the effects of adversaries on models, also studying methods to defend against such adversaries. These adversaries

have been extensively studied in computer vision, and more recently, several attacks have been proposed in RL. These include test-time attacks on observations (Biggio et al., 2013; Szegedy et al., 2013; Goodfellow et al., 2015; Papernot et al., 2017), (training-time) poisoning attacks (Mei and Zhu, 2015; Li et al., 2016), and backdoor attacks Chen et al. (2017); Gu et al. (2017); Salem et al. (2022). Closest to our work are poisoning attacks on RL (Huang et al., 2017; Sun et al., 2021; Zhang et al., 2021), which have considered different poisoning aims, including transitions (Ma et al., 2019; Rakhsha et al., 2020), rewards (Lin et al., 2017; Zhang et al., 2020b; Sun et al., 2020; Wu et al., 2023; Nika et al., 2023), states (Zhang et al., 2020a), and actions (Rangi et al., 2022). Prior work has also considered poisoning attacks in multi-agent RL (Mohammadi et al., 2023; Wu et al., 2024). We contribute to this line of work by studying the robustness of in-context RL under *test-time* poisoning attacks.

Corruption robustness in RL. Our work is closest to the literature on corruption-robust bandits, RL, and multi-agent RL (Lykouris et al., 2018; Rakhsha et al., 2020; Niss and Tewari, 2020; Chen et al., 2021b; Lee et al., 2021; Wei et al., 2022; Ding et al., 2022; Nika et al., 2023; Xu et al., 2024). These works often establish guarantees for the suboptimality gap in terms of the level of corruption. Rather than focusing on theory, we contribute a practical method for training corruption robust in-context RL. As explained in the introduction, this approach is conceptually different: corruption robust learning is implemented in-context. We experimentally compare the efficacy of our approach to bandit and RL algorithms robust to reward contamination, such as corruption robust UCB (Niss and Tewari, 2020; Ding et al., 2022) and Natural Policy Gradient (NPG, Kakade, 2001; Zhang et al., 2021).

Our work is also tied to the literature on robust offline RL (Yang et al., 2022; Panaganti et al., 2022; Ye et al., 2023; Yang et al., 2023; Yang and Xu, 2024). Prior work DeFog (Hu et al., 2023), or concurrent work LHF (Chen et al., 2025) rely on filtering the learning histories during training. We note that both of these works are developed for robustness against random or noisy perturbations. For *adversarial* corruption robustness, another concurrent work (Xu et al., 2025) studies several improvements for the Decision Transformer. However, this method focuses on the single-task setting, compared to ours.

In-context reinforcement learning. In terms of RL paradigms, the closest to our work is in-context RL. We have already explained the connection to the Decision-Pretrained Transformer Lee et al. (2023), which we build upon. A similar work, Algorithm Distillation, trained with episodic trajectories from learning algorithm histories distills a policy which implicitly produces actions imitating policy improvement (Laskin et al., 2022). An extension of this involves injecting noise in the curriculum to allow generating learning histories without the need for optimal actions (Zisman et al., 2024). Both this and another (Dong et al., 2024) prior work show that ICRL is sensitive to perturbations the pretraining dataset. We also mention the work of Tang et al. (2024), who study the Adversarially Robust Decision Transformer (ARDT) – a method robust against an adaptive adversary within a Markov game framework, capable of choosing actions which minimize the victim's rewards. This framework, translated to ours, would correspond to an adversary modifying transition probabilities and the victim observing the action the adversary took. In contrast, instead of adversarial transition probabilities, we consider adversarial rewards generated by the attacker, and the victim only observes the realized reward, without knowledge of whether an attacker is interfering, nor knowledge of their algorithm. For an in-depth discussion on in-context RL, we refer to Moeini et al. (2025).

Meta-RL. Our work is broadly related to meta-RL, since we consider a multi-task setting. Within decision-making and RL, meta-RL has been used in a variety of ways – optimizing a policy conditioned on histories of past transitions via an RNN (Duan et al., 2016), similarly, utilizing a Structured State Space Sequence model replacing the RNN (Lu et al., 2023), learning good 'starting point' parameters that make learning in tasks faster (Finn et al., 2017), learning a dynamics model shared across tasks (Nagabandi et al., 2018). Transformers have also been utilized in prior work in learning multi-task policies (Reed et al., 2022; Lee et al., 2022). For a more in-depth discussion on meta-RL, we refer to the survey by Beck et al. (2023).

Other. Recently there have been many works studying various different attacks on large language models (LLMs) to provoke an unsafe response (Zhao et al., 2024; He et al., 2024; Cheng et al., 2024, and many others), also called red-teaming (Ganguli et al., 2022). The increasing use of LLMs within decision-making systems provoke the need to study robustness. Therefore, we advocate for the study of robust decision-making algorithms and hope our method contributes to this body of knowledge.

3 Setup

Notation. We will use $\Delta(\mathcal{A})$ to refer to the probability distribution over $\mathcal{A}$, and $\|\cdot\|_2$ denotes the Euclidean norm. We will use notation similar to Lee et al. (2023).

3.1 In-context sequential decision-making

Environment. We consider a multi-task sequential decision making setting, where we denote $\mathcal{T}$ as the distribution of tasks. Each task $\mathcal{M} \sim \mathcal{T}$ is formalized as an episodic finite-horizon Markov decision process (MDP) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, R, T, H, \rho \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $R : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$ is the reward function, $T : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition function, $H \in \mathbb{N}$ is the horizon, and $\rho \in \Delta(\mathcal{S})$ is the starting state distribution. We denote realized states, actions, and rewards at timestep h by s_h, a_h, and r_h, respectively. We distinguish between the clean and corrupted settings and use $\bar{r}_h$ to denote the true rewards, and $r_h^\dagger$ to denote the rewards produced by an attacker. The attack model is introduced in the next subsection. Let $\mu_{\bar{R}}(s, a)$ denote the mean of the underlying environment reward for the state-action pair (s, a). For a stochastic policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, the value function is defined as $V^\pi(\rho) = \mathbb{E}_{s\sim\rho}\left[\sum_{h=1}^H \bar{r}_h \mid \pi, s_0 = s\right]$, where the expectation is w.r.t. the randomness of the underlying rewards when rolling out policy π in $\mathcal{M}$. The solution to task $\mathcal{M}$ is an optimal policy $\pi^\star_{\mathcal{M}}$ that maximizes the value function, i.e., $V^{\pi^\star_{\mathcal{M}}}(\rho) = \max_\pi V^\pi(\rho)$.

Agent. We model a learning agent as a context-dependent policy parameterized by a transformer with parameters θ which maps the history of interactions D and a query state s_{query} to a distribution over actions. We denote this policy by $\pi_\theta(a_h \mid D, s_h)$ and $D = \{(s_i, a_i, r_i, s_{i+1})\}_{i=0}^{H-1}$ is the *in-context dataset* consisting of a set of previous interactions. To implement an efficient learner, we can train $\pi_\theta(\cdot \mid D, s_h)$ to predict optimal actions $a_h^\star \sim \pi^\star_{\mathcal{M}}(\cdot \mid s_h)$ for a task $\mathcal{M}$ sampled from a given task distribution – this approach is the backbone of the Decision-Pretrained Transformer (DPT, Lee et al., 2023).

3.2 Attack model

We consider bounded reward poisoning attacks applied to a fraction of tasks at *test-time*. We employ Huber's ε-contamination model (Huber, 1964) and assume that the agent observes the corrupted reward in ε-fraction of timesteps. We model the attacker $\pi_\phi^\dagger : \mathcal{S}\times\mathcal{A}\times\mathbb{R}\times(\mathcal{S}\times\mathcal{A}\times\mathbb{R}\times\mathcal{S})^C \to \Delta(\mathbb{R})$ as a function of the state, action and reward of the last timestep along with an in-context dataset $\bar{D} \in (\mathcal{S} \times \mathcal{A} \times \mathbb{R} \times \mathcal{S})^C$ consisting of C tuples of agent's interactions. Formally, at timestep h, the environment generates $\bar{r}_h \sim R(s_h, a_h)$, and the agent observes

$$\tilde{r}_h = \begin{cases} r_h^\dagger \sim \pi_\phi^\dagger(\cdot \mid s_h, a_h, \bar{r}_h, \bar{D}) & \text{with probability } \varepsilon, \\ \bar{r}_h & \text{otherwise.} \end{cases}$$

The attacker observes the underlying environment reward $\bar{r}_h$ to generate $r_h^\dagger$, but the victim π_θ only observes the realized reward $\tilde{r}_h$. We call an attacker *adaptive* if $C > 0$, meaning it leverages the agent's past interactions, and *non-adaptive* if $C = 0$. In the non-adaptive case ($C = 0$) we simplify $\pi_\phi^\dagger(\cdot \mid s_h, a_h, \bar{r}_h, \bar{D}) = \pi_\phi^\dagger(\cdot \mid s_h, a_h, \bar{r}_h)$. In both cases the attacker aims to minimize the agent's expected return in $\mathcal{M}$ under a soft budget constraint, without forcing a specific policy for the agent. We denote the mean and variance of corrupted rewards by $\mu_\phi(s, a) = \mathbb{E}_{r^\dagger\sim\pi_\phi^\dagger(\cdot|s,a)}[r^\dagger]$ and $\sigma_\phi(s, a) = \text{Var}_{r^\dagger\sim\pi_\phi^\dagger(\cdot|s,a)}[r^\dagger]$. Formally, the attacker's objective is

$$L(\mathcal{M}, \phi, \theta) = \mathbb{E}\left[\sum_{h=1}^H -\bar{r}_h \mid \pi_\theta, \pi_\phi^\dagger\right] - \lambda \cdot c_\mu\left(\|\mu_\phi - \mu_{\bar{R}}\|_2\right) - \lambda \cdot c_\sigma\left(\|\sigma_\phi\|_2\right), \tag{1}$$

where we take the expectation over the stochasticity of the environment, the agent's policy, and the contamination model. c_μ, c_σ are penalty functions for exceeding budget B and B_σ respectively, and $\lambda > 0$ controls the strength.

3.3 In-context RL with corrupted rewards

To account for the change induced by the attacker, we set the agent's objective to $U(\mathcal{M}, \theta, \phi) = \mathbb{E}\left[\sum_h^H \bar{r}_h \mid \pi_\theta, \pi_\phi^\dagger\right]$, where the expectation is taken over the randomness of the realized rewards when running the policy π_θ in $\mathcal{M}$, while corrupting its context $D^\dagger$ using the ε-contamination model with the attack policy $\pi_\phi^\dagger$.

We search for a Nash equilibrium $(\theta^\star, \{\phi^\star_\mathcal{M}\}_{\mathcal{M}\in\mathcal{T}})$ such that $\theta^\star \in \arg\max_\theta \mathbb{E}_{\mathcal{M}\in\mathcal{T}}[U(\mathcal{M}, \theta, \phi^\star_\mathcal{M})]$ and $\phi^\star_\mathcal{M} \in \arg\max_\phi L(\mathcal{M}, \theta^\star, \phi)$ for all $\mathcal{M} \in \mathcal{T}$. Our goal is to devise a training procedure for approximating this equilibrium. We do this by sampling M tasks, and for every task i training a separate attacker $\pi^\dagger_{\phi_{\mathcal{M}_i}}$. Along with the attackers we simultaneously train the agent π_θ. This provokes our adversarial training approach.

4 Method

We extend DPT (Lee et al., 2023) with adversarial training. We follow a similar approach as in the original work. The setup consists of three phases.

Pretraining. Lee et al. (2023) use GPT-2 as the underlying transformer model, and we adopt the same architecture. The model π_θ is initialized from scratch and trained via supervised learning by predicting an optimal action from context D_{pre} and query state s_q. During the pretraining phase, the model observes a dataset $D_{\text{pre}} \sim \mathcal{D}_{\text{pre}}$, which consists of tuples (s, a, r, s') sampled from a set of M tasks $\{\mathcal{M}_i \sim \mathcal{T}\}_{i=1}^M$. This dataset can be collected in various ways, such as through random interactions with the environments. Alongside these interactions, we also sample a query state $s_q \sim \mathcal{D}_{\text{query}}$ and its corresponding optimal action $a^\star \sim \pi^\star_\mathcal{M}(\cdot \mid s_q)$. The model is then trained to minimize $\min_\theta \mathbb{E}_{D_{\text{pre}}\sim\mathcal{D}_{\text{pre}}, s_q\sim\mathcal{D}_{\text{query}}} \ell(\pi_\theta(\cdot|D_{\text{pre}}, s_q), a^\star)$, where ℓ is the NLL loss.

In-context learning. During the test phase π_θ is deployed in $\mathcal{M} \sim \mathcal{T}$ with an empty context $D = \{\}$. The original work updated the context D with the entire trajectory $\{(s_h, a_h, r_h, s_{h+1})\}_{h=1}^H$ only after the entire episode (Lee et al., 2023). Whereas, in our method we update context D from interacting with the environment, with transitions (s_h, a_h, r_h, s_{h+1}) after every timestep h, to support robustness against adaptive attacks.

Adversarial training. Before testing, we include an additional phase for adversarial training. An illustration of this training process is shown in Figure 1. In the adversarial setting π_θ is deployed in $\mathcal{M}$ under an attacker $\pi_\phi^\dagger$, contaminating the victim's dataset $D^\dagger$ as specified in the previous section. We account for this by introducing an additional adversarial training stage between the original pretraining and in-context learning. To train the agent and the attacker, recall that we use two different contexts – a context with poisoned rewards $D^\dagger$ for the agent, and a context with underlying rewards $\bar{D}$ for the attacker. We repeat this process for N rounds, updating θ and ϕ after each round. Parameters θ are updated as in the original DPT setting, with s_q sampled from the environment, and $a^\star$ provided by an oracle.[2] The pseudocode of this method can be found in Algorithm 1.

We consider attackers parameterized by ϕ (e.g., a neural network). To train the attacker we use the REINFORCE algorithm (Williams, 1992) – after each episode we update ϕ with the objective specified in Equation (1). Recall that while the victim π_θ only observes the realized reward $\tilde{r}_h$, the attacker has to have access to the underlying environment reward $\bar{r}_h$. The attacker's goal is to poison a single algorithm, which we denote the **attacker target**. That is, a different policy might emerge from an attacker targeting, for example, DPT versus an attacker targeting TS.

Bandit setting. In the bandit settings we consider a direct parameterization of a deterministic attack, i.e., for an action $a_h^{(i)}$ (choosing arm i) at timestep h the attack becomes $\pi_\phi^\dagger(\cdot \mid a_h^{(i)}, \bar{r}_h) = \pi_\phi^\dagger(a_h^{(i)}, \bar{r}_h) = \bar{r}_h + \phi(i)$, where $\phi \in \mathbb{R}^{|\mathcal{A}|}$.

Adaptive attacker. We also consider a context-dependent algorithm, e.g., a transformer, to enable the attacker to adapt to the defenses of the victim. For this we utilize the same architecture (GPT-2) as

[2] In the algorithm and our experiments we require access to clean environments sampled from $\mathcal{T}$ at training time, although offline trajectories could be used with simulated attacks and (near-)optimal actions.

Algorithm 1 Adversarially Trained Decision Pretrained Transformer (AT-DPT)

input: victim π_θ – DPT with pretrained params θ_0
input: attacker $\pi^\dagger_\phi$ with initial params ϕ_0, budget B, fraction of steps poisoned ε
Sample M tasks $\{\mathcal{M}_i \sim \mathcal{T}\}_{i=1}^m$
for round n in $0 \dots N-1$, simultaneously in all $\mathcal{M}$ **do**
  roll out π_{θ_n} for H steps in $\mathcal{M}_i$ poisoned by $\pi^\dagger_\phi$ with ε-contamination model and budget B,
    where DPT collects corrupted dataset $D^\dagger$, and attacker collects dataset $\bar{D}$
  $\phi_{n+1} \leftarrow$ train on $\bar{D}$ with REINFORCE:
    see Equation (1)
  $\theta_{n+1} \leftarrow$ train on $D^\dagger$ via supervised learning:
    $\min_\theta \ell(\pi_\theta(\cdot \mid D^\dagger, s_q), a^\star)$, $a^\star$ provided by oracle
end for

the victim. The interaction in the environment is then modified as follows. At the start of an episode empty context $\bar{D} = \{\}$ is initialized for the attacker. At every step h the attacker samples a reward $r^\dagger_h \sim \pi^\dagger_\phi(\cdot \mid \bar{D}, s_h, a_h, \bar{r}_h)$ for the victim and appends $(s_h, a_h, \bar{r}_h)$ to the dataset $\bar{D}$.

MDP Setting. In the MDP setting we also consider a direct parameterization of a deterministic non-adaptive attack, similar to the bandit attacker, i.e., for a state-action pair $(s^{(i)}, a^{(j)})$ the attack becomes $\pi^\dagger_\phi(\cdot \mid s^{(i)}, a^{(j)}, \bar{r}) = \pi^\dagger_\phi(s^{(i)}, a^{(j)}, \bar{r}) = \bar{r} + \phi(i, j)$, where $\phi \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{A}|}$.

5 Experiments

We sample $M = 200$ tasks to run in parallel. For each round we train both the attacker and DPT for multiple (e.g., 20) iterations on the same dataset. We set penalties for exceeding the budget $c_\mu(x; B) = \max(0, x - B)$ and $c_\sigma(x; B_\sigma) = \max(0, x - B_\sigma)$ with $\lambda = 10$.

5.1 Baseline algorithms

To evaluate our method's performance in the bandit setting we compare it with widely used baseline bandit algorithms, and choose several corruption robust algorithms: Thompson sampling (TS, Thompson, 1933), upper confidence bound (UCB, Auer et al., 2002), robust Thompson sampling (RTS, Xu et al., 2024) – a Thompson sampling based algorithm robust to adversarial reward poisoning, which features an added term to the bonus term in TS, and corruption-robust upper confidence bound (crUCB, Niss and Tewari, 2020) – a UCB style algorithm robust to ε-contamination, where we chose the trimmed mean variant, while the mean is estimated with a fraction of smallest and largest observed values removed for every arm, otherwise being very similar to UCB. For linear bandits we compare our method to LinUCB (Li et al., 2010), and a corruption robust variant – CRLinUCB (Ding et al., 2022, Section 4).

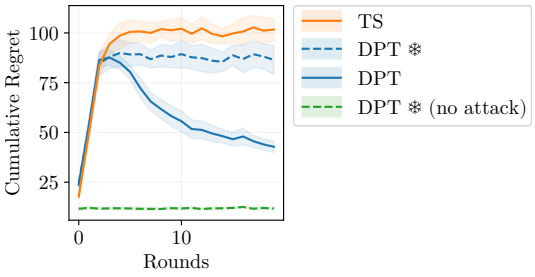

Figure 2: Comparison of the cumulative regret per round (lower is better) of different methods throughout 20 rounds of adversarial training (simultaneously learning AT-DPT and an attacker) in the bandit setting. Within one round we perform $H = 500$ steps. The y axis indicates cumulative regret for that round. Mean and 95% confidence interval (2×SEM) over 10 experiment replications. Attack budget $B = 3$, $\varepsilon = 0.4$. DPT ❄ indicates DPT with frozen parameters.

For the MDP baselines we choose a policy-gradient based method – Natural policy gradient (NPG, Kakade, 2001); and a value-based method – Q-learning (Watkins and Dayan, 1992). Additionally, we include DPT with frozen parameters (indicated as DPT ❄) as a baseline to observe the effect of adversarial training. More details about the baselines can be found in the Appendix.

In addition to algorithm baselines, we also consider two baselines for evaluation – we show performance of the algorithms in the clean environment, and we also consider a uniform random attack

Table 1: Comparison of cumulative regret (lower is better) of different algorithms under different attackers trained for 20 rounds, with $\varepsilon = 0.4$ steps poisoned. For $\varepsilon = 0.1$ and 0.2 see the Appendix. Mean and 95% confidence interval (2×SEM) over 10 experiment replications. Attack budget $B = 3$. * We use tuned versions of RTS and crUCB which outperform the base versions; full comparisons including base versions are given in the Appendix.

Algorithm	Attacker Target						Unif. Rand. Attack	Clean Env.
	AT-DPT	DPT ❄	TS	RTS*	UCB1.0	crUCB*		
AT-DPT	24.2 ± 1.2	24.8 ± 1.4	29.8 ± 3.0	28.3 ± 1.9	24.5 ± 0.8	23.8 ± 1.4	45.4 ± 5.5	13.0 ± 0.9
DPT ❄	63.6 ± 8.6	59.4 ± 5.2	62.0 ± 8.6	59.1 ± 7.3	55.4 ± 8.1	58.8 ± 7.8	27.2 ± 2.9	11.5 ± 0.5
TS	106.3 ± 3.8	97.7 ± 4.9	94.3 ± 3.8	93.1 ± 6.0	89.6 ± 1.8	92.6 ± 4.8	19.1 ± 0.8	8.7 ± 0.6
RTS*	102.9 ± 4.5	97.0 ± 4.2	90.4 ± 4.4	92.5 ± 5.0	89.2 ± 3.4	89.0 ± 3.0	16.2 ± 1.4	10.2 ± 0.4
UCB1.0	104.1 ± 3.6	95.8 ± 4.9	90.6 ± 4.1	90.0 ± 5.2	88.1 ± 3.4	91.2 ± 3.4	17.6 ± 0.8	16.0 ± 0.5
crUCB*	86.0 ± 4.4	85.0 ± 2.3	82.0 ± 4.4	82.4 ± 3.3	79.4 ± 3.0	82.5 ± 5.1	17.9 ± 0.6	15.8 ± 0.5

– the poisoned reward for timestep h is $r_h^\dagger = \bar{r}_h + \phi(i)$, where $\phi \in \mathbb{R}^{|\mathcal{A}|}$ is generated once at the start of evaluation by sampling from a uniform random distribution, and later clipped by the budget constraint $\|\phi\|_2 < B$.

5.2 Bandit setting

Environment. We begin with empirical results in a simple scenario – the multi-armed bandit problem. We follow a similar bandit setup to that presented in the original DPT paper (Lee et al., 2023). We sample 5-armed bandits ($|\mathcal{A}| = 5$), each arm's reward function being a normal distribution $R(\cdot \mid s, a) = \mathcal{N}(\mu^{(a)}, \sigma^2)$, where $\mu^{(a)} \sim \text{Unif}[0, 1]$ independently and $\sigma = 0.3$. The optimal policy in this environment is to always choose the arm with the largest mean: $a^\star = \arg\max_a \mu^{(a)}$. We follow the same pretraining scheme as the original work. For evaluation, we present the empirical cumulative regret: $\sum_h \bar{r}(a^\star) - \bar{r}(a_h)$. Low regret indicates the policy is close to optimal.

Hyperparameters To pretrain DPT in the bandit setting we use the following architecture and hyperparameters. The Transformer has 4 layers, 4 attention heads per layer, embedding dimension – 32, no dropout. We set the context length equal to episode length $H = 500$, learning rate $\eta = 0.001$ and train for 400 epochs. We pretrain DPT in the same way as the original, see the work by Lee et al. (2023) for more details. For adversarial training we use the learning rate $\eta = 0.0001$ for the victim and $\eta_{\text{attacker}} = 0.03$. We consider attackers with a diagonal covariance matrix, and set $B_\sigma = 1$.

Adversarial training makes DPT robust to poisoning attacks. In Figure 2, we present the training-time performance of DPT under adversarial training. The training curve shows the per-round cumulative regret, averaged across M tasks seen during training. We observe the regret significantly increase in the first rounds, but in further rounds DPT learns to recover from the attacks, and results improve. In the figure we compare performance of TS and frozen DPT under the same attack, and also show the performance of frozen DPT on the clean environment (no attack). We refer to the adversarially trained DPT models as AT-DPT.

Evaluation. To evaluate AT-DPT on attacks trained for it, we cross-validate to prevent evaluation on the same attack AT-DPT has seen during training – we evaluate one AT-DPT with an attacker which is targeting AT-DPT for a different seed. We report the mean and 95% confidence interval (2×SEM) across 10 different experiment replications. We run adversarial training for $N = 20$ rounds. Table 1 presents an extensive evaluation of AT-DPT and other method performance against attackers targeting various different methods. We can clearly see AT-DPT outperforming all baselines in an adversarially trained attacker setting. Given that AT-DPT displays robustness against attackers from different algorithms illustrates that AT-DPT can successfully recover from attacks that are out-of-distribution. Although, adversarial training seems to trade-off the performance in the clean and random attack environment, where the frozen model (DPT ❄), or even a baseline algorithm like TS perform better.

Adaptive Attacks. For the adaptive attacker we utilize the same architecture as the victim, except without pretraining. In this setting we use $\eta_{\text{attacker}} = 0.00003$. Table 2 shows a comparison of

performance under both adaptive and non-adaptive attackers. The result shows low regret for AT-DPT, displaying robustness against this type of attack as well.

5.3 Linear bandit setting

Environment. We follow a similar setup as in the original DPT work (Lee et al., 2023). We sample d-armed linear bandits, where the reward is given by $\mathbb{E}[r \mid a, \mathcal{M}_i] = \langle \omega_i, \psi(a) \rangle$, and $\omega_i \in \mathbb{R}^d$ is a task-specific parameter vector, and $\psi : \mathcal{A} \to \mathbb{R}^d$ is a feature vector shared across all tasks. Both ω_i for every i and ψ are sampled from $\mathcal{N}(\mathbf{0}, I_d/d)$. In our experiments we chose $d = 2$ and $|\mathcal{A}| = 10$, same as in the original paper.

Results. From the results in Table 3 we see CRLinUCB performing only marginally better than all other algorithms in the clean case and uniform random attack. Although, under a more complex attack AT-DPT outperforms all other algorithms, and matches CRLinUCB in the clean case and uniform random attack.

5.4 MDP setting

Environment. In the MDP setting we consider an extension of a sparse reward MDP considered in prior work – the Dark Room environment (Lee et al., 2023; Laskin et al., 2022; Zintgraf et al., 2020) – a 2D gridworld environment where the agent only observes its own state and gains a reward of 1 when at the goal state. The agent has 5 actions – $\mathcal{A} = \{\text{up}, \text{down}, \text{left}, \text{right}, \text{stay}\}$. We consider a modification of this environment – instead of having one goal, we consider two goals – one giving a reward of 1, the other giving 2. To pretrain the DPT we supply optimal actions that lead to the goal giving reward of 2. We refer to this environment as Darkroom2.

To conform to the sparse reward nature of this environment we constrain the attacker to only output attacks in $\{-1, 0, 1\}$, having a softmax parameterization. This results in the observed reward being one of $\{-1, 0, 1, 2, 3\}$. We do not perform any reward normalization or scaling. In the evaluations we present the underlying episode reward $\sum_h^H \bar{r}_h$ as the performance metric.

Hyperparameters To pretrain DPT in the Darkroom2 setting we use the same model architecture as for the bandit setting, the context length equal to episode length $H = 200$, learning rate is $\eta = 0.0001$ and train for 150 epochs. For adversarial training we use the learning rate $\eta = 0.00003$ for the victim and $\eta_{\text{attacker}} = 0.03$.

Evaluation. To evaluate AT-DPT we perform cross-validation with different attackers same as in the bandit setting. For evaluation, we present the total underlying episode reward $\sum_h \bar{r}_h$ in the tables. We report the mean and 95% confidence interval (2×SEM) across 10 different experiment replications. We run adversarial training for $N = 400$ rounds. The results, seen in Table 4, show that AT-DPT is

Table 2: Comparison of the cumulative regret (lower is better) of adaptive and non-adaptive attackers. Attackers trained for 400 rounds, with $\varepsilon = 0.4$ steps poisoned. For $\varepsilon = 0.1$ and 0.2 see the Appendix. AT-DPT (A) means AT-DPT trained against the adaptive attacker, AT-DPT (n-A) means AT-DPT trained against the non-adaptive attacker. Mean and 95% confidence interval (2×SEM) over 10 experiment replications. Attack budget $B = 3$.
* We use tuned versions of RTS and crUCB which outperform base versions; details in the Appendix.

Algorithm	Attacker Target: Adaptive AT-DPT	Adaptive TS	Non-adaptive AT-DPT	Non-adaptive TS	Unif. Rand. Attack	Clean Env.
AT-DPT (A)	37.1 ± 6.6	36.4 ± 9.4	38.0 ± 6.4	42.6 ± 6.7	35.1 ± 8.3	21.3 ± 9.0
AT-DPT (n-A)	88.1 ± 20.0	81.0 ± 11.2	22.8 ± 1.6	29.8 ± 2.2	47.0 ± 6.0	13.8 ± 1.2
DPT ❄	97.9 ± 18.6	82.1 ± 20.7	61.6 ± 8.0	61.6 ± 6.6	29.2 ± 2.5	12.1 ± 0.8
TS	90.2 ± 21.9	104.2 ± 26.7	106.3 ± 5.5	94.3 ± 4.8	19.6 ± 1.0	9.1 ± 0.7
RTS*	90.5 ± 21.3	103.6 ± 26.8	104.5 ± 5.5	90.9 ± 4.2	16.7 ± 0.9	10.5 ± 0.6
UCB	94.3 ± 22.4	103.9 ± 28.4	101.3 ± 5.0	87.8 ± 4.4	18.3 ± 0.7	16.0 ± 0.4
crUCB*	85.1 ± 23.5	79.6 ± 29.4	88.4 ± 4.4	79.9 ± 4.7	18.0 ± 0.6	15.8 ± 0.3

Table 3: Comparison of the cumulative regret (lower is better) of the different algorithms under different attackers in the **linear bandit setting**, with $\varepsilon = 0.4$ steps poisoned. Attack budget $B = 3$. For $\varepsilon = 0.1$ and 0.2 see the Appendix. Mean and 95% confidence interval ($2\times$SEM) over 10 experiment replications.
* We use a tuned version of CRLinUCB which outperforms the base version; details in the Appendix.

Algorithm	Attacker Target				Unif. Rand. Attack	Clean Env.
	AT-DPT	DPT ❄	LinUCB	CRLinUCB*		
AT-DPT	2.49 ± 1.06	2.50 ± 1.08	2.83 ± 1.10	1.79 ± 1.02	5.33 ± 1.16	3.89 ± 0.86
DPT ❄	70.29 ± 7.32	71.42 ± 7.46	70.83 ± 7.76	63.84 ± 7.18	6.62 ± 1.30	3.35 ± 0.84
LinUCB	37.69 ± 4.46	35.93 ± 3.86	35.22 ± 4.14	34.82 ± 4.36	5.21 ± 1.16	3.51 ± 0.88
CRLinUCB*	37.45 ± 4.76	33.03 ± 4.00	35.56 ± 4.26	35.36 ± 4.80	5.12 ± 1.48	2.94 ± 0.78

robust against different attackers, but only slightly better than NPG. Additional results can be found in the Appendix.

The robustness displayed by NPG has been also observed by Zhang et al. (2021) – they find that NPG can be robust against ε-contamination, if the rewards generated by the adversary are bounded. We also observe that attacks with $\varepsilon = 0.1$ and $\varepsilon = 0.2$ are not very effective for NPG and Q-learning.

The main advantage of using AT-DPT over NPG or other RL methods in these types of scenarios is that of generalization – DPT is a meta-learner, which infers the task from a few interactions with the environment and follows an optimal policy almost immediately. Conversely, NPG and Q-learning are task-specific 'online' learners, meaning they require interactions from the current environment to improve their policies; although not to be confused with the standard definition of online learning (Levine et al., 2020).

These algorithms require a few (tens/hundreds) of episodes before converging to a stable policy. In our experiments we trained a different NPG and Q-learning policy for each environment, although one could argue that it may be possible to use a universal task conditioned policy. In these settings the agent is not aware what is the current task, therefore it is unclear what it needs to conditioned on.

6 Discussion

In our work we have presented AT-DPT – a method to adversarially train the DPT to robustify it against reward poisoning attacks. This is done via simultaneously training the attacker, minimizing the underlying environment rewards, and the victim, optimizing for the optimal actions from the poisoned data. By showing extensive evaluations on the bandit and MDP setting we demonstrated AT-DPT has the ability to recover optimal actions from the poisoned data.

Table 4: Comparison of the average episode reward (higher is better) of the different algorithms under different attackers trained for 300 rounds (5 rounds for Q-learning and NPG) in the **Darkroom2 environment** (5×5 grid). Mean and 95% confidence interval ($2\times$SEM) over 10 experiment replications, with $\varepsilon = 0.4$ steps poisoned. For $\varepsilon = 0.1$ and 0.2 see the Appendix. Attack budget $B = 10$. § NPG and Q-learning require multiple episodes of online learning to converge to a stable policy; we run them for 100 episodes before evaluating their performance.

Algorithm	Attacker Target				Unif. Rand. Attack	Clean Env.
	AT-DPT	DPT ❄	NPG	Q-learning		
AT-DPT	241.2 ± 9.1	268.7 ± 8.5	241.0 ± 12.7	239.1 ± 8.8	219.6 ± 14.9	266.4 ± 15.3
DPT ❄	214.8 ± 7.9	140.2 ± 10.4	206.1 ± 7.5	205.9 ± 7.8	123.9 ± 6.9	302.3 ± 6.3
NPG§	237.2 ± 6.7	243.7 ± 7.9	228.9 ± 4.0	228.1 ± 8.1	230.3 ± 6.4	248.5 ± 6.0
Q-learning§	198.3 ± 4.4	235.9 ± 4.8	112.5 ± 5.0	140.3 ± 9.2	249.7 ± 6.2	247.7 ± 6.3

We see that by training the DPT with poisoned rewards in the context leads to behavior that is robust against these perturbations. Similarly, within the text domain Cheng et al. (2024) find that pretraining a transformer with noisy labels works well against that type of perturbation.

Limitations and future work. The main limitation of our method, also a limitation of DPT is the need of actions provided by the oracle for training (Lee et al., 2023). The authors of DPT propose relaxing this requirement by supplying actions generated by another RL agent which performs well for the current task, although this might not be possible in an adversarial scenario. A different approach, where training on offline trajectories with a simulated attacker could be viable.

We also observe in our results the capability of AT-DPT to generalize beyond the attack it has been trained on (i.e., adversarially trained against its own specific attacker, generalizes to an attacker trained for TS, for example). This suggests it may be possible to exploit this further by adversarially training AT-DPT with multiple different contamination levels ε. Additionally in our results we only consider a single attack specification per experiment. To make AT-DPT even more robust, and potentially alleviate the trade-off observed in the clean and random attack environment it would be possible to train AT-DPT with multiple different attack specifications (e.g., mixing in non-adaptive and adaptive attacks), or diversify them, which we leave as a direction for future work.

Acknowledgments and Disclosure of Funding

This research was, in part, funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 467367360.

References

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2–3):235–256, May 2002. ISSN 0885-6125. doi: 10.1023/A:1013689704352. URL https://doi.org/10.1023/A:1013689704352.

Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part III 13*, pages 387–402. Springer, 2013.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in Neural Information Processing Systems*, 34:15084–15097, 2021a.

Weiqin Chen, Xinjie Zhang, Dharmashankar Subramanian, and Santiago Paternain. Filtering learning histories enhances in-context reinforcement learning. *arXiv preprint arXiv:2505.15143*, 2025.

Xinyun Chen, Chang Liu, Bo Li, Kimberly Lu, and Dawn Song. Targeted backdoor attacks on deep learning systems using data poisoning. *arXiv preprint arXiv:1712.05526*, 2017.

Yifang Chen, Simon Du, and Kevin Jamieson. Improved corruption robust algorithms for episodic reinforcement learning. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 1561–1570. PMLR, 18–24 Jul 2021b. URL https://proceedings.mlr.press/v139/chen21d.html.

Chen Cheng, Xinzhi Yu, Haodong Wen, Jingsong Sun, Guanzhang Yue, Yihao Zhang, and Zeming Wei. Exploring the robustness of in-context learning with noisy labels. In *ICLR 2024 Workshop on Reliable and Responsible Foundation Models*, 2024.

Qin Ding, Cho-Jui Hsieh, and James Sharpnack. Robust stochastic linear contextual bandits under adversarial attacks. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 7111–7123. PMLR, 28–30 Mar 2022. URL https://proceedings.mlr.press/v151/ding22c.html.

Juncheng Dong, Moyang Guo, Ethan X Fang, Zhuoran Yang, and Vahid Tarokh. In-context reinforcement learning without optimal action labels. In *ICML 2024 Workshop on In-Context Learning*, 2024.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. RL^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*, 2022.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Pengfei He, Han Xu, Yue Xing, Hui Liu, Makoto Yamada, and Jiliang Tang. Data poisoning for in-context learning. *arXiv preprint arXiv:2402.02160*, 2024.

Kaizhe Hu, Ray Chen Zheng, Yang Gao, and Huazhe Xu. Decision transformer under random frame dropping. *arXiv preprint arXiv:2303.03391*, 2023.

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

Peter J. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35 (1):73 – 101, 1964. doi: 10.1214/aoms/1177703732. URL https://doi.org/10.1214/aoms/1177703732.

Sham Kakade. A natural policy gradient. In *Proceedings of the 15th International Conference on Neural Information Processing Systems: Natural and Synthetic*, NIPS'01, page 1531–1538, Cambridge, MA, USA, 2001. MIT Press.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Hansen, Angelos Filos, Ethan Brooks, et al. In-context reinforcement learning with algorithm distillation. *arXiv preprint arXiv:2210.14215*, 2022.

Chung-Wei Lee, Haipeng Luo, Chen-Yu Wei, Mengxiao Zhang, and Xiaojin Zhang. Achieving near instance-optimality and minimax-optimality in stochastic and adversarial linear bandits simultaneously. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6142–6151. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/lee21h.html.

Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised pretraining can learn in-context reinforcement learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=dCYBAGQXLo.

Kuang-Huei Lee, Ofir Nachum, Mengjiao Sherry Yang, Lisa Lee, Daniel Freeman, Sergio Guadarrama, Ian Fischer, Winnie Xu, Eric Jang, Henryk Michalewski, et al. Multi-game decision transformers. *Advances in Neural Information Processing Systems*, 35:27921–27936, 2022.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review. *and Perspectives on Open Problems*, 5, 2020.

Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. *Advances in neural information processing systems*, 29, 2016.

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, page 661–670, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781605587998. doi: 10.1145/1772690.1772758. URL https://doi.org/10.1145/1772690.1772758.

Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748*, 2017.

Chris Lu, Yannick Schroecker, Albert Gu, Emilio Parisotto, Jakob Foerster, Satinder Singh, and Feryal Behbahani. Structured state space models for in-context reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 47016–47031. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/92d3d2a9801211ca3693ccb2faa1316f-Paper-Conference.pdf.

Thodoris Lykouris, Vahab Mirrokni, and Renato Paes Leme. Stochastic bandits robust to adversarial corruptions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 114–122, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355599. doi: 10.1145/3188745.3188918. URL https://doi.org/10.1145/3188745.3188918.

Yuzhe Ma, Xuezhou Zhang, Wen Sun, and Jerry Zhu. Policy poisoning in batch reinforcement learning and control. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/315f006f691ef2e689125614ea22cc61-Paper.pdf.

Shike Mei and Xiaojin Zhu. Using machine teaching to identify optimal training-set attacks on machine learners. In *Proceedings of the aaai conference on artificial intelligence*, volume 29, 2015.

Amir Moeini, Jiuqi Wang, Jacob Beck, Ethan Blaser, Shimon Whiteson, Rohan Chandra, and Shangtong Zhang. A survey of in-context reinforcement learning. *arXiv preprint arXiv:2502.07978*, 2025.

Mohammad Mohammadi, Jonathan Nöther, Debmalya Mandal, Adish Singla, and Goran Radanovic. Implicit poisoning attacks in two-agent reinforcement learning: Adversarial policies for training-time attacks. *arXiv preprint arXiv:2302.13851*, 2023.

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

Andi Nika, Adish Singla, and Goran Radanovic. Online defense strategies for reinforcement learning against adaptive reward poisoning. In *26th International Conference on Artificial Intelligence and Statistics*, pages 335–358. PMRL, 2023.

Laura Niss and Ambuj Tewari. What you see may not be what you get: Ucb bandit algorithms robust to ε-contamination. In Jonas Peters and David Sontag, editors, *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*, volume 124 of *Proceedings of Machine Learning Research*, pages 450–459. PMLR, 03–06 Aug 2020. URL https://proceedings.mlr.press/v124/niss20a.html.

Kishan Panaganti, Zaiyan Xu, Dileep Kalathil, and Mohammad Ghavamzadeh. Robust reinforcement learning using offline data. *Advances in neural information processing systems*, 35:32211–32224, 2022.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Amin Rakhsha, Goran Radanovic, Rati Devidze, Xiaojin Zhu, and Adish Singla. Policy teaching via environment poisoning: Training-time adversarial attacks against reinforcement learning. In *International Conference on Machine Learning*, pages 7974–7984. PMLR, 2020.

Anshuka Rangi, Haifeng Xu, Long Tran-Thanh, and Massimo Franceschetti. Understanding the limits of poisoning attacks in episodic reinforcement learning. *arXiv preprint arXiv:2208.13663*, 2022.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-maron, Mai Giménez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *Transactions on Machine Learning Research*, 2022.

Ahmed Salem, Rui Wen, Michael Backes, Shiqing Ma, and Yang Zhang. Dynamic backdoor attacks against machine learning models. In *2022 IEEE 7th European Symposium on Security and Privacy (EuroS&P)*, pages 703–718. IEEE, 2022.

Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma, Yan Zheng, Kangjie Chen, and Yang Liu. Stealthy and efficient adversarial attacks against deep reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5883–5891, Apr. 2020. doi: 10.1609/aaai.v34i04.6047. URL https://ojs.aaai.org/index.php/AAAI/article/view/6047.

Yanchao Sun, Da Huo, and Furong Huang. Vulnerability-aware poisoning mechanism for online rl with unknown dynamics. In *International Conference on Learning Representations*, 2021.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Xiaohang Tang, Afonso Marques, Parameswaran Kamalaruban, and Ilija Bogunovic. Adversarially robust decision transformer. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=WEf2LT8NtY.

William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933. ISSN 00063444. URL http://www.jstor.org/stable/2332286.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

Chen-Yu Wei, Christoph Dann, and Julian Zimmert. A model selection approach for corruption robust reinforcement learning. In Sanjoy Dasgupta and Nika Haghtalab, editors, *Proceedings of The 33rd International Conference on Algorithmic Learning Theory*, volume 167 of *Proceedings of Machine Learning Research*, pages 1043–1096. PMLR, 29 Mar–01 Apr 2022. URL https://proceedings.mlr.press/v167/wei22a.html.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Young Wu, Jeremy McMahan, Xiaojin Zhu, and Qiaomin Xie. Reward poisoning attacks on offline multi-agent reinforcement learning. In *Proceedings of the aaai conference on artificial intelligence*, volume 37, pages 10426–10434, 2023.

Young Wu, Jeremy McMahan, Xiaojin Zhu, and Qiaomin Xie. Data poisoning to fake a nash equilibria for markov games. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38 (14):15979–15987, Mar. 2024. doi: 10.1609/aaai.v38i14.29529. URL https://ojs.aaai.org/index.php/AAAI/article/view/29529.

Jiawei Xu, Rui Yang, Feng Luo, Meng Fang, Baoxiang Wang, and Lei Han. Robust decision transformer: Tackling data corruption in offline rl via sequence modeling. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=phAlw3JPms.

Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *international conference on machine learning*, pages 24631–24645. PMLR, 2022.

Yinglun Xu, Zhiwei Wang, and Gagandeep Singh. Robust thompson sampling algorithms against reward poisoning attacks. *arXiv preprint arXiv:2410.19705*, 2024.

Rui Yang, Chenjia Bai, Xiaoteng Ma, Zhaoran Wang, Chongjie Zhang, and Lei Han. Rorl: Robust offline reinforcement learning via conservative smoothing. *Advances in neural information processing systems*, 35:23851–23866, 2022.

Rui Yang, Han Zhong, Jiawei Xu, Amy Zhang, Chongjie Zhang, Lei Han, and Tong Zhang. Towards robust offline reinforcement learning under diverse data corruption. *arXiv preprint arXiv:2310.12955*, 2023.

Zhihe Yang and Yunjian Xu. Dmbp: Diffusion model-based predictor for robust offline reinforcement learning against state observation perturbations. In *The Twelfth International Conference on Learning Representations*, 2024.

Chenlu Ye, Rui Yang, Quanquan Gu, and Tong Zhang. Corruption-robust offline reinforcement learning with general function approximation. *Advances in Neural Information Processing Systems*, 36:36208–36221, 2023.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. *Advances in Neural Information Processing Systems*, 34:21024–21037, 2020a.

Xuezhou Zhang, Yuzhe Ma, Adish Singla, and Xiaojin Zhu. Adaptive reward-poisoning attacks against reinforcement learning. In *International Conference on Machine Learning*, pages 11225–11234. PMLR, 2020b.

Xuezhou Zhang, Yiding Chen, Xiaojin Zhu, and Wen Sun. Robust policy gradient against strong data corruption. In *International Conference on Machine Learning*, pages 12391–12401. PMLR, 2021.

Shuai Zhao, Meihuizi Jia, Luu Anh Tuan, and Jinming Wen. Universal vulnerabilities in large language models: In-context learning backdoor attacks. *arXiv preprint arXiv:2401.05949*, 2024.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Hkl9JlBYvr.

Ilya Zisman, Vladislav Kurenkov, Alexander Nikulin, Viacheslav Sinii, and Sergey Kolesnikov. Emergence of in-context reinforcement learning from noise distillation. In *Forty-first International Conference on Machine Learning*, 2024.

Supplementary Material: Can In-Context Reinforcement Learning Recover From Reward Poisoning Attacks?

Appendices

A Baseline algorithms

A.1 Robust TS

Xu et al. (2024) provide the Robust TS algorithm. This algorithm relies on a corruption level hyperparameter $\bar{C}$. The recommendation given by the authors is to set this to $\sum_h^H c_h \leq \bar{C}$, where c_h is the corruption level (i.e., in our case, $c_h = r_h - \bar{r}_h$) for step h, if the corruption level is known. If the corruption level is unknown, the authors suggest setting $\bar{C} = \sqrt{H \frac{\ln |\mathcal{A}|}{|\mathcal{A}|}}$.

Following these recommendations, in an environment with $H = 500$, $|\mathcal{A}| = 5$, $\varepsilon = 0.4$, our preliminary findings are:

- assume corruption level is known: $\bar{C} \approx 120$ – RTS performance is worse than TS; indicated as RTS ($\bar{C}$ known);
- assume corruption level is unknown: $\bar{C} \approx 12.7$ – RTS performance is worse than TS; indicated as RTS ($\bar{C}$ unk.);
- tuned $\bar{C}$ for our setup: $\bar{C} = 0.5$ – RTS performance is better than TS; indicated as RTS ($\bar{C}$ tuned).

We report the best scores (obtained with the tuned variant) in the main text, giving the full three variant comparison in Table 5.

A.2 crUCB

Niss and Tewari (2020) provide a few variants of the crUCB algorithm. We chose the α-trimmed variant, which performs best empirically. We introduce a modification to the algorithm due to poor original variant empirical performance. The modified variant is shown in Algorithm 2, where f – α-trimmed mean function – if n is the number of rewards observed for that arm, removes $\lceil \alpha\, n \rceil$ lowest and $\lceil \alpha\, n \rceil$ highest rewards observed for that specific arm; removing $2\lceil \alpha\, n \rceil$ elements in total, and $\mathbf{x}_a^{(h)}$ – list of observed rewards for arm a at step h.

Algorithm 2 crUCB (α-trimmed variant), modified

input: α – fraction of steps poisoned
input: σ_0 – upper bound on sub-Gaussian constant (hyperparameter)
input: f – mean estimate function (α-trimmed mean)
for step $h = 1, \ldots, H$ **do**
  for each $a \in \mathcal{A}$ **do**
    $\hat{\mu}_a^{(h)} \leftarrow f(\mathbf{x}_a^{(h)})$ (α-trimmed mean estimate of rewards)
    $N_a^{(h)} \leftarrow$ number of times action a has been played
  end for
  Choose action $a = \arg\max_{a \in \mathcal{A}} \hat{\mu}_a^{(h)} + \sigma_0 \left(\sqrt{\frac{4 \log(h)}{\lfloor (1-2\alpha)\, N_a^{(h)} \rfloor}} \right)$
end for

The original bonus term in the algorithm is $\frac{\sigma_0}{1-2\alpha} \left(\sqrt{\frac{4 \log(h)}{N_a^{(h)}}} \right)$.

Assume $f(\mathbf{z})$ with n elements returns zero if $\mathbf{z}$ contains fewer than $n - 2\lceil \alpha\, n \rceil$ elements. The failure is observed when the assumption above is true – the estimated mean returns zero, whereas the bonus is not infinity, leading to arms which have only been played one time have a very low score.

We report the best scores (obtained with the modified variant) in the main text, giving full results in Table 5 comparing:

- the original variant, indicated as crUCB (orig.) or (o.);
- the original variant with σ_0 scaled by $\sqrt{1-2\alpha}$, indicated as crUCB (low σ_0) or (l. σ_0);
- the modified variant, indicated as crUCB (mod.) or (m.).

A.3 CRLinUCB

We source the CRLinUCB algorithm from Ding et al. (2022). The authors suggest setting the upper bound of the budget C' to equal εBH. We found that the algorithm did not perform well when set to this value. We then tuned this variant, and present a number of results in the tables:

- the original variant, denoted as CRLinUCBv1, where the hyperparameters are set to the values suggested by Theorem 1 by Ding et al. (2022);
- the variant where the bound is divided by the time horizon H, denoted as CRLinUCBv2, which approximately matches the values of the experiments of Ding et al. (2022);
- a third variant, CRLinUCBv3, where the hyperparameters are interpolated between v1 and v2, they are within the same order of magnitude with the geometric mean of the values used in v1 and v2.

In the main text we report the results from CRLinUCBv2, which seemed to work best in our case.

B Additional results

B.1 Bandit setting

We present Table 5, which shows the full set of results from the bandit setting. Note, that RTS ($\bar{C}$ unk.), RTS ($\bar{C}$ known) did not perform well, as also noted in Appendix A.1. Similarly, crUCB (orig.) did not perform well, as noted in Appendix A.2. The high values obtained in the case, where the attacker is crUCB (o.) mean that the attacker trained against this algorithm was not performing well, and therefore led to a weak attack. Recall that the setup has a dual objective, and simply judging by the regret or reward of a single row or column is not enough.

Table 5: Comparison of the cumulative regret (lower is better) of the different algorithms under different attackers trained for 20 rounds, in the bandit setting. Mean and 95% confidence interval (2×SEM) over 10 experiment replications. Attack budget $B = 3$.

Algorithm	Attacker Target								Unif. Rand. Attack	Clean Env.
	AT-DPT	DPT ❄	TS	RTS ($\bar{C}$ t.)	UCB1.0	crUCB (o.)	crUCB (l. σ_0)	crUCB (m.)		
$\varepsilon = 0.1$										
AT-DPT	14.5 ± 0.9	13.9 ± 0.7	14.4 ± 0.8	14.6 ± 1.0	14.8 ± 0.8	14.4 ± 1.4	14.1 ± 0.6	14.2 ± 1.3	14.2 ± 1.2	12.4 ± 1.1
DPT ❄	24.2 ± 1.8	22.7 ± 2.0	22.1 ± 1.3	23.0 ± 1.6	22.2 ± 1.6	21.2 ± 1.8	22.6 ± 1.5	22.2 ± 1.8	15.2 ± 0.8	12.1 ± 0.5
TS	27.9 ± 1.1	26.6 ± 2.1	28.4 ± 1.8	27.2 ± 1.6	25.9 ± 1.6	22.4 ± 2.1	28.0 ± 1.8	28.6 ± 1.8	12.0 ± 1.0	8.9 ± 0.4
RTS ($\bar{C}$ tuned)	27.1 ± 0.6	27.0 ± 1.4	26.9 ± 1.0	26.5 ± 2.0	24.2 ± 1.1	21.3 ± 1.4	26.2 ± 1.4	26.8 ± 1.2	13.0 ± 0.8	10.5 ± 0.3
RTS ($\bar{C}$ unk.)	59.8 ± 0.5	59.4 ± 0.8	59.2 ± 0.6	59.7 ± 0.7	58.7 ± 0.6	55.7 ± 1.0	59.2 ± 1.0	59.1 ± 0.7	49.9 ± 0.8	49.3 ± 0.8
RTS ($\bar{C}$ known)	94.9 ± 0.8	94.4 ± 0.9	94.5 ± 0.8	94.6 ± 1.2	94.0 ± 0.8	91.8 ± 1.1	93.8 ± 1.1	94.4 ± 1.0	84.7 ± 1.6	84.1 ± 1.7
UCB1.0	30.8 ± 1.5	28.5 ± 1.5	29.5 ± 0.8	28.5 ± 1.8	27.3 ± 1.5	24.5 ± 1.0	28.7 ± 1.5	29.4 ± 1.3	17.9 ± 0.4	16.1 ± 0.3
crUCB (orig.)	82.4 ± 0.8	81.9 ± 0.7	82.5 ± 0.9	81.9 ± 1.3	82.1 ± 0.7	81.3 ± 1.0	82.1 ± 0.8	81.9 ± 0.8	79.2 ± 1.4	79.3 ± 1.5
crUCB (low σ_0)	19.5 ± 1.8	19.1 ± 1.2	20.0 ± 1.0	18.8 ± 2.1	20.1 ± 1.5	18.6 ± 1.7	19.9 ± 1.6	19.6 ± 2.0	11.1 ± 0.7	9.3 ± 0.5
crUCB (mod.)	19.4 ± 1.7	18.4 ± 1.2	20.5 ± 1.2	17.8 ± 1.6	19.7 ± 1.1	18.7 ± 1.6	19.5 ± 1.0	18.4 ± 1.6	11.0 ± 0.7	9.2 ± 0.3
$\varepsilon = 0.2$										
AT-DPT	17.9 ± 1.4	17.1 ± 1.4	19.0 ± 1.5	18.4 ± 1.3	17.8 ± 1.4	17.4 ± 1.9	17.0 ± 0.9	16.9 ± 1.2	20.4 ± 2.0	14.5 ± 1.8
DPT ❄	35.2 ± 4.1	33.2 ± 3.5	37.1 ± 5.2	35.1 ± 3.1	33.0 ± 3.5	28.9 ± 3.1	35.1 ± 3.7	33.1 ± 3.9	22.3 ± 1.1	12.1 ± 0.5
TS	51.1 ± 3.2	48.6 ± 3.5	51.1 ± 3.1	50.1 ± 2.7	47.2 ± 2.4	33.7 ± 3.3	51.8 ± 1.7	49.9 ± 3.8	18.7 ± 1.1	8.9 ± 0.4
RTS ($\bar{C}$ tuned)	49.8 ± 3.4	44.8 ± 2.7	48.1 ± 2.0	48.3 ± 3.1	44.3 ± 3.8	32.4 ± 2.5	47.6 ± 2.0	46.3 ± 1.7	19.1 ± 1.1	10.5 ± 0.3
RTS ($\bar{C}$ unk.)	76.0 ± 2.1	73.2 ± 1.2	74.5 ± 1.7	74.2 ± 1.7	72.1 ± 1.3	63.5 ± 1.2	73.8 ± 1.2	73.7 ± 1.2	53.0 ± 0.9	49.3 ± 0.8
RTS ($\bar{C}$ known)	131.7 ± 1.8	130.7 ± 1.5	131.3 ± 1.5	130.9 ± 1.5	130.6 ± 1.5	127.0 ± 1.6	130.4 ± 1.5	131.0 ± 1.6	116.4 ± 2.5	115.7 ± 2.6
UCB1.0	51.9 ± 2.5	46.7 ± 1.8	50.8 ± 1.9	47.3 ± 1.9	45.2 ± 2.7	34.2 ± 2.3	49.1 ± 1.8	48.1 ± 2.7	23.9 ± 0.8	16.1 ± 0.3
crUCB (orig.)	101.6 ± 1.2	100.8 ± 1.2	101.7 ± 1.3	100.5 ± 1.6	101.0 ± 1.0	98.8 ± 1.3	101.1 ± 1.1	101.3 ± 1.2	96.0 ± 2.0	95.6 ± 2.0
crUCB (low σ_0)	34.7 ± 2.1	33.6 ± 2.1	34.4 ± 2.1	31.6 ± 2.8	32.9 ± 1.6	30.5 ± 3.0	32.9 ± 2.2	34.1 ± 1.9	15.2 ± 0.6	9.4 ± 0.6
crUCB (mod.)	33.7 ± 1.8	33.6 ± 1.7	33.9 ± 2.2	31.1 ± 2.3	31.8 ± 2.2	29.9 ± 3.0	33.4 ± 2.7	33.5 ± 1.5	15.1 ± 1.0	9.5 ± 0.3
$\varepsilon = 0.4$										
AT-DPT	24.2 ± 1.2	24.8 ± 1.4	29.8 ± 3.0	28.3 ± 1.9	24.5 ± 0.8	23.4 ± 1.6	24.4 ± 1.3	23.8 ± 1.4	38.7 ± 1.7	17.8 ± 1.4
DPT ❄	63.6 ± 8.6	59.4 ± 5.2	62.0 ± 8.6	59.1 ± 7.3	55.4 ± 8.1	41.8 ± 6.2	58.5 ± 7.4	58.8 ± 7.8	37.2 ± 1.2	12.1 ± 0.5
TS	106.3 ± 3.8	97.7 ± 4.9	94.3 ± 3.8	93.1 ± 6.0	89.6 ± 1.8	48.0 ± 2.4	92.2 ± 6.0	92.6 ± 4.8	34.2 ± 1.6	8.9 ± 0.4
RTS ($\bar{C}$ tuned)	102.9 ± 4.5	97.0 ± 4.2	90.4 ± 4.4	92.5 ± 5.0	89.2 ± 3.4	46.8 ± 2.9	90.6 ± 4.8	89.0 ± 3.0	33.9 ± 1.6	10.5 ± 0.3
RTS ($\bar{C}$ unk.)	113.0 ± 2.6	108.3 ± 2.8	104.2 ± 2.9	104.5 ± 3.2	103.0 ± 2.4	73.9 ± 2.3	104.2 ± 2.7	105.0 ± 2.0	62.8 ± 1.5	49.3 ± 0.8
RTS ($\bar{C}$ known)	156.4 ± 1.9	155.2 ± 1.9	154.6 ± 1.7	154.6 ± 2.0	154.6 ± 1.9	149.7 ± 2.0	154.9 ± 1.9	155.3 ± 2.0	139.5 ± 3.4	138.9 ± 3.4
UCB1.0	104.1 ± 3.6	95.8 ± 4.9	90.6 ± 4.1	90.0 ± 5.2	88.1 ± 3.4	46.8 ± 2.4	91.9 ± 3.6	91.2 ± 3.4	38.1 ± 2.2	16.1 ± 0.3
crUCB (orig.)	148.3 ± 1.9	147.8 ± 2.0	148.0 ± 2.0	147.1 ± 2.1	147.9 ± 1.7	145.0 ± 1.9	147.7 ± 1.7	148.1 ± 1.8	139.8 ± 3.4	139.8 ± 3.5
crUCB (low σ_0)	85.9 ± 3.0	83.0 ± 3.7	82.8 ± 3.1	84.5 ± 3.5	80.9 ± 4.3	64.2 ± 3.6	82.4 ± 4.4	84.6 ± 4.6	31.4 ± 1.5	14.8 ± 0.4
crUCB (mod.)	86.0 ± 4.4	85.0 ± 2.3	82.0 ± 4.4	82.4 ± 3.3	79.4 ± 3.0	64.2 ± 2.4	80.2 ± 3.4	82.5 ± 5.1	31.8 ± 1.6	15.8 ± 0.3

B.2 Bandit setting, adaptive attack

Table 6 presents the full set of results comparing adaptive and non-adaptive attacks. The adaptive attacker columns in the table highlight, that these attacks work much better from the attacker's perspective, i.e., the attacks increase regret by a larger margin than in the non-adaptive case. We note that in both cases the regret of AT-DPT is low, meaning it is working well.

Table 6: Comparison of the cumulative regret (lower is better) of adaptive and non-adaptive attackers in the bandit setting. Attackers trained for 400 rounds. Mean and 95% confidence interval (2×SEM) over 10 experiment replications. Attack budget $B = 3$. * We use tuned versions of RTS and crUCB which outperform the base versions – see Appendix A for details.

Algorithm	Attacker Target: Adaptive AT-DPT	Adaptive TS	Non-adaptive AT-DPT	Non-adaptive TS	Unif. Rand. Attack	Clean Env.
			$\varepsilon = 0.1$			
AT-DPT (against adaptive)	21.5 ± 4.2	22.6 ± 5.5	19.6 ± 3.8	19.6 ± 4.2	19.4 ± 4.6	16.7 ± 5.1
AT-DPT (against non-adaptive)	44.6 ± 17.7	39.7 ± 15.4	14.1 ± 0.7	14.2 ± 0.6	17.5 ± 1.4	11.8 ± 1.1
DPT ❄	60.2 ± 18.5	47.1 ± 10.1	22.2 ± 1.1	21.9 ± 1.5	16.9 ± 1.1	12.1 ± 0.8
TS	102.8 ± 27.1	91.0 ± 29.3	27.8 ± 1.7	27.4 ± 1.6	11.1 ± 0.7	9.1 ± 0.7
RTS ($\bar{C}$ tuned)	101.5 ± 26.2	91.7 ± 29.0	26.1 ± 2.1	26.4 ± 2.1	11.5 ± 0.5	10.5 ± 0.6
UCB	103.9 ± 26.0	94.0 ± 27.2	28.8 ± 1.1	29.2 ± 1.1	15.9 ± 0.6	16.0 ± 0.4
crUCB (mod.)	67.2 ± 17.1	53.3 ± 15.8	18.9 ± 1.3	19.6 ± 1.5	9.8 ± 0.6	9.2 ± 0.3
			$\varepsilon = 0.2$			
AT-DPT (against adaptive)	26.6 ± 5.3	29.1 ± 8.5	25.9 ± 5.1	27.0 ± 4.5	25.3 ± 6.3	18.8 ± 7.6
AT-DPT (against non-adaptive)	54.7 ± 15.5	51.8 ± 11.8	17.5 ± 0.9	19.4 ± 1.3	24.9 ± 3.6	12.4 ± 1.3
DPT ❄	71.3 ± 20.4	61.6 ± 17.0	34.6 ± 3.6	35.1 ± 3.1	20.6 ± 1.2	12.1 ± 0.8
TS	74.9 ± 24.6	91.6 ± 35.0	51.6 ± 2.6	51.8 ± 3.6	13.3 ± 0.5	9.1 ± 0.7
RTS ($\bar{C}$ tuned)	75.5 ± 24.6	92.2 ± 34.2	49.5 ± 2.9	49.3 ± 2.7	13.3 ± 0.6	10.5 ± 0.6
UCB	76.8 ± 22.4	92.4 ± 30.9	51.6 ± 2.7	49.4 ± 1.9	16.4 ± 0.7	16.0 ± 0.4
crUCB (mod.)	55.1 ± 16.5	61.8 ± 25.0	34.6 ± 1.8	34.4 ± 1.2	11.0 ± 0.5	9.5 ± 0.5
			$\varepsilon = 0.4$			
AT-DPT (against adaptive)	37.1 ± 6.6	36.4 ± 9.4	38.0 ± 6.4	42.6 ± 6.7	35.1 ± 8.3	21.3 ± 9.0
AT-DPT (against non-adaptive)	88.1 ± 20.0	81.0 ± 11.2	22.8 ± 1.6	29.8 ± 2.2	47.0 ± 6.0	13.8 ± 1.2
DPT ❄	97.9 ± 18.6	82.1 ± 20.7	61.6 ± 8.0	61.6 ± 6.6	29.2 ± 2.5	12.1 ± 0.8
TS	90.2 ± 21.9	104.2 ± 26.7	106.3 ± 5.5	94.3 ± 4.8	19.6 ± 1.0	9.1 ± 0.7
RTS ($\bar{C}$ tuned)	90.5 ± 21.3	103.6 ± 26.8	104.5 ± 5.5	90.9 ± 4.2	16.7 ± 0.9	10.5 ± 0.6
UCB	94.3 ± 22.4	103.9 ± 28.4	101.3 ± 5.0	87.8 ± 4.4	18.3 ± 0.7	16.0 ± 0.4
crUCB (mod.)	85.1 ± 23.5	79.6 ± 29.4	88.4 ± 4.4	79.9 ± 4.7	18.0 ± 0.6	15.8 ± 0.3

B.3 Linear bandit setting

Table 7 presents the full results from the linear bandit setting. As described in Appendix A.3, CRLinUCBv1 and CRLinUCBv3 performed worse than the tuned version CRLinUCBv2. This is indicated by their poor performance on the clean and uniform random attack cases.

Table 7: Comparison of the cumulative regret (lower is better) of the different algorithms under different attackers in the **linear bandit setting**. Mean and 95% confidence interval (2×SEM) over 10 experiment replications. Attack budget $B = 3$.

Algorithm	Attacker Target						Unif. Rand. Attack	Clean Env.
	AT-DPT	DPT ❄	LinUCB	CRLinUCBv1	CRLinUCBv2	CRLinUCBv3		
$\varepsilon = 0.1$								
AT-DPT	2.55 ± 0.88	2.02 ± 0.92	2.28 ± 0.90	2.44 ± 0.96	1.55 ± 0.98	1.85 ± 0.94	4.60 ± 0.98	3.89 ± 0.86
DPT ❄	14.50 ± 2.30	14.42 ± 2.48	14.62 ± 2.34	14.06 ± 2.72	14.02 ± 2.74	13.19 ± 2.36	5.23 ± 1.02	3.35 ± 0.84
LinUCB	10.57 ± 2.00	7.92 ± 1.24	7.56 ± 1.20	9.65 ± 1.78	8.04 ± 1.52	9.23 ± 1.72	4.18 ± 0.90	3.51 ± 0.88
CRLinUCBv1	104.00 ± 7.06	103.24 ± 7.00	103.11 ± 7.04	103.03 ± 6.98	102.56 ± 7.02	102.97 ± 7.00	102.95 ± 7.00	110.85 ± 7.78
CRLinUCBv2	7.85 ± 1.58	7.99 ± 1.60	10.16 ± 2.00	10.94 ± 2.34	7.82 ± 1.92	9.07 ± 2.48	3.16 ± 0.96	2.94 ± 0.78
CRLinUCBv3	17.66 ± 1.18	17.34 ± 1.18	17.40 ± 1.24	17.86 ± 1.20	16.99 ± 1.14	16.58 ± 1.12	13.55 ± 0.96	34.08 ± 1.66
$\varepsilon = 0.2$								
AT-DPT	1.37 ± 0.94	1.20 ± 0.92	1.67 ± 0.96	1.30 ± 0.96	2.42 ± 0.94	2.00 ± 0.92	4.80 ± 0.98	3.89 ± 0.86
DPT ❄	33.65 ± 4.14	35.49 ± 4.66	32.23 ± 3.96	35.29 ± 4.24	33.67 ± 4.02	33.15 ± 3.84	5.91 ± 1.12	3.35 ± 0.84
LinUCB	19.11 ± 2.56	15.79 ± 2.32	18.80 ± 2.62	21.31 ± 3.16	16.95 ± 2.68	19.52 ± 2.62	4.37 ± 0.98	3.51 ± 0.88
CRLinUCBv1	100.19 ± 6.88	99.73 ± 6.74	99.35 ± 6.82	99.41 ± 6.88	100.48 ± 6.90	100.34 ± 6.88	107.34 ± 7.44	110.85 ± 7.78
CRLinUCBv2	16.42 ± 2.76	13.97 ± 2.46	16.56 ± 2.46	22.27 ± 3.66	16.02 ± 2.64	18.45 ± 2.78	3.41 ± 1.02	2.94 ± 0.78
CRLinUCBv3	31.53 ± 1.76	28.93 ± 1.62	28.66 ± 1.58	30.44 ± 1.80	30.02 ± 1.64	30.20 ± 1.74	19.42 ± 1.08	34.08 ± 1.66
$\varepsilon = 0.4$								
AT-DPT	2.49 ± 1.06	2.50 ± 1.08	2.83 ± 1.10	2.93 ± 1.06	1.79 ± 1.02	2.16 ± 1.10	5.33 ± 1.16	3.89 ± 0.86
DPT ❄	70.29 ± 7.32	71.42 ± 7.46	70.83 ± 7.76	69.49 ± 6.88	63.84 ± 7.18	73.45 ± 6.82	6.62 ± 1.30	3.35 ± 0.84
LinUCB	37.69 ± 4.46	35.93 ± 3.86	35.22 ± 4.14	49.39 ± 5.12	34.82 ± 4.36	39.97 ± 4.50	5.21 ± 1.16	3.51 ± 0.88
CRLinUCBv1	108.12 ± 6.96	107.54 ± 7.00	108.04 ± 6.98	107.84 ± 6.98	106.79 ± 6.98	107.13 ± 6.90	109.46 ± 7.64	110.85 ± 7.78
CRLinUCBv2	37.45 ± 4.76	33.03 ± 4.00	35.56 ± 4.26	46.23 ± 5.46	35.36 ± 4.80	37.75 ± 4.80	5.12 ± 1.48	2.94 ± 0.78
CRLinUCBv3	53.23 ± 3.02	51.76 ± 2.84	53.31 ± 2.98	54.45 ± 3.08	49.13 ± 2.66	51.43 ± 2.78	28.34 ± 1.56	34.08 ± 1.66

B.4 MDP setting

In Table 8 we present the full set of results for the Darkroom2 environment.

Table 8: Comparison of the average episode reward (higher is better) of the different algorithms under different attackers trained for 300 rounds (5 rounds for Q-learning and NPG) in the **Darkroom2 environment** (5×5 grid). Mean and 95% confidence interval (2×SEM) over 10 experiment replications. Attack budget $B = 10$. [§] NPG and Q-learning require multiple episodes of online learning to converge to a stable policy; we run them for 100 episodes before evaluating their performance.

Algorithm	Attacker Target				Unif. Rand. Attack	Clean Env.
	AT-DPT	DPT ❄	NPG	Q-learning		
			$\varepsilon = 0.1$			
AT-DPT	266.4 ± 18.2	267.8 ± 20.7	262.5 ± 17.5	258.9 ± 20.2	236.0 ± 22.5	273.4 ± 17.8
DPT ❄	233.6 ± 8.6	198.1 ± 13.0	220.9 ± 10.1	222.6 ± 6.8	141.7 ± 9.9	305.4 ± 6.1
NPG[§]	241.9 ± 6.6	248.1 ± 6.3	247.7 ± 7.1	243.3 ± 5.5	150.8 ± 2.1	151.5 ± 2.5
Q-learning[§]	283.5 ± 3.9	280.2 ± 5.1	246.6 ± 38.4	264.3 ± 17.2	266.0 ± 15.7	263.4 ± 15.4
			$\varepsilon = 0.2$			
AT-DPT	263.1 ± 17.6	269.7 ± 15.1	258.0 ± 17.7	258.0 ± 19.3	245.6 ± 17.0	275.8 ± 16.4
DPT ❄	227.9 ± 9.5	169.9 ± 13.3	213.6 ± 6.8	217.4 ± 9.5	131.5 ± 8.4	305.4 ± 6.1
NPG[§]	244.1 ± 7.5	244.4 ± 6.7	239.5 ± 9.4	241.2 ± 9.4	151.9 ± 2.1	151.5 ± 2.5
Q-learning[§]	240.6 ± 3.6	254.1 ± 6.2	236.5 ± 6.1	246.1 ± 5.3	246.8 ± 5.1	243.3 ± 6.4
			$\varepsilon = 0.4$			
AT-DPT	241.2 ± 9.1	268.7 ± 8.5	241.0 ± 12.7	239.1 ± 8.8	219.6 ± 14.9	266.4 ± 15.3
DPT ❄	214.8 ± 7.9	140.2 ± 10.4	206.1 ± 7.5	205.9 ± 7.8	123.9 ± 6.9	302.3 ± 6.3
NPG[§]	237.2 ± 6.7	243.7 ± 7.9	228.9 ± 4.0	228.1 ± 8.1	230.3 ± 6.4	248.5 ± 6.0
Q-learning[§]	198.3 ± 4.4	235.9 ± 4.8	112.5 ± 5.0	140.3 ± 9.2	249.7 ± 6.2	247.7 ± 6.3

B.5 Training curves

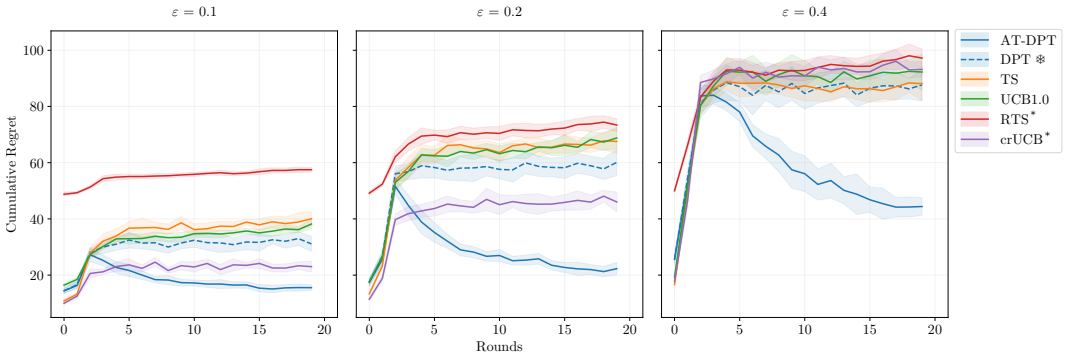

Figure 3: Adversarial training curves for training the attacker in the bandit setting, for different values of ε. Note, that in the case of AT-DPT it is trained along with the attackers. * We use tuned versions of RTS and crUCB, see Appendices A.1 and A.2 for more details.

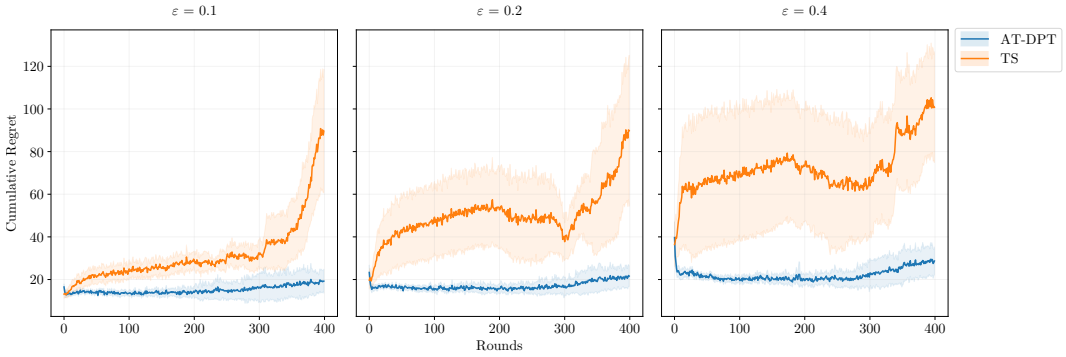

Figure 4: Adversarial training curves for training the adaptive attacker in the bandit setting, for different values of ε. Note, that in the case of AT-DPT it is trained along with the attackers.

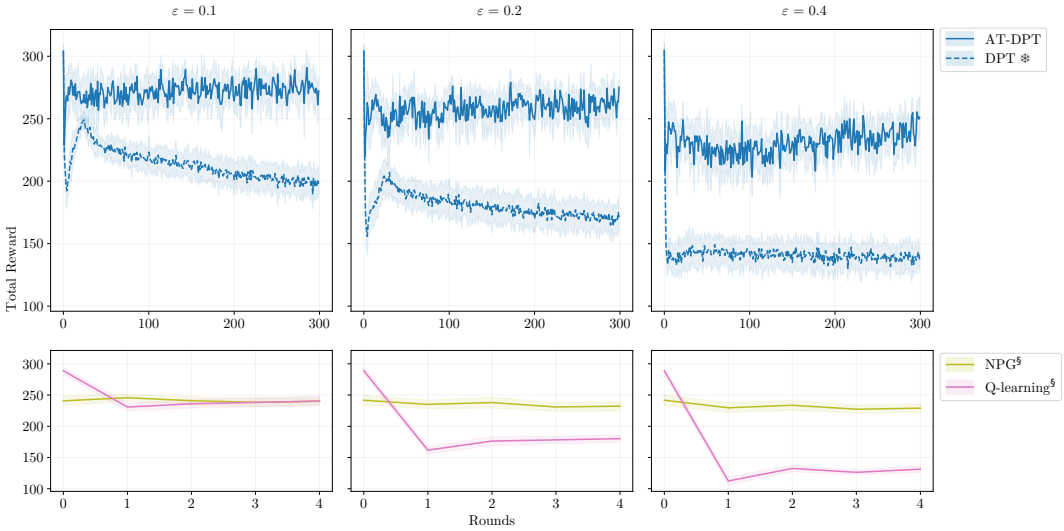

Figure 5: Adversarial training curves for training the attacker in the Darkroom2 environment, for different values of ε. Note, that in the case of AT-DPT it is trained along with the attackers. § NPG and Q-learning require multiple episodes of online learning to converge to a stable policy; we run them for 100 episodes before evaluating their performance.

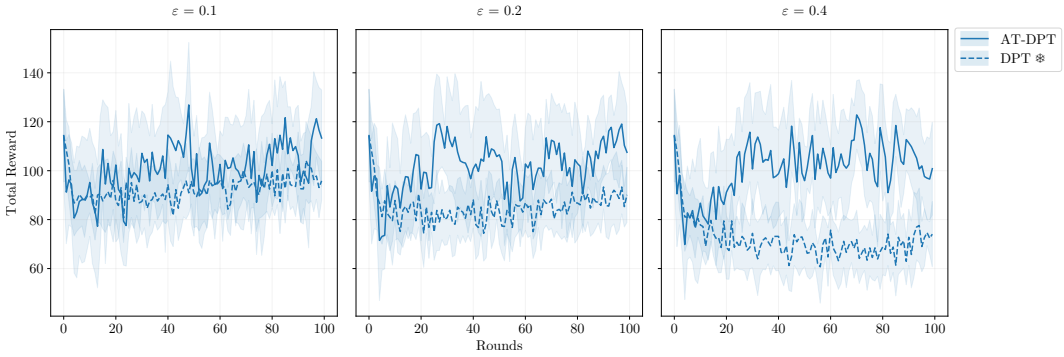

Figure 6: Adversarial training curves for training the attacker in the Miniworld environment, for different values of ε. Note, that in the case of AT-DPT it is trained along with the attackers.

C Further details

C.1 Compute resources

The experiments were run on a compute cluster with machines containing Nvidia A100 80GB PCIe and Nvidia H100 94GB NVL GPUs.

Approximate GPU machine runtime of experiments, per run:

- Bandit Environment:
  - Pretraining – 3.4 h
  - Adversarial training – 0.4 h
  - Evaluation – 0.6 h
- Bandit Environment, Adaptive Attacker:
  - Adversarial training – 0.6 h
  - Evaluation – 0.2 h
- Darkroom2 Environment:
  - Pretraining – 1.4 h
  - Adversarial training – 0.6 h
  - Evaluation – 3.4 h[§]
- Miniworld Environment:
  - Pretraining – 13.1 h
  - Adversarial training – 2.7 h
  - Evaluation – 0.9 h

[§] NPG and Q-learning required multiple episodes of online learning before converging to a stable policy, therefore leading to an increased evaluation run time.

C.2 Different budgets

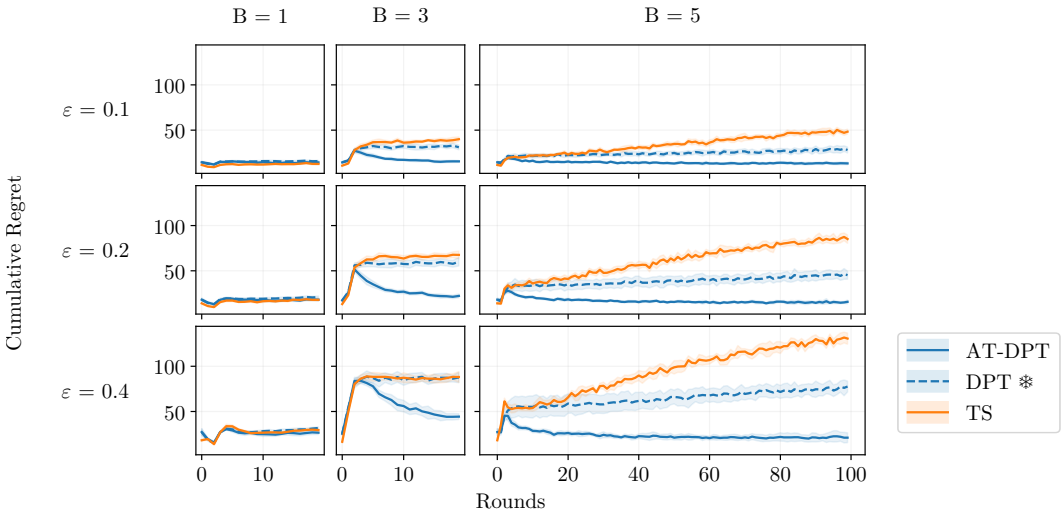

Figure 7: A study of the effect of the budget B on the regret in the bandit setting. We run the experiments for $B = 5$ for more rounds to observe convergence. We observe that a larger budget for the attacker leads to a higher regret for TS and DPT ❄, although adversarial training for AT-DPT helps it learn to recover from the attack.

C.3 Interpretation of attack in Darkroom2

We present an illustration of an example environment and attacker's strategy in Figure 8, taken from the middle of a sample AT-DPT adversarial training run. The attacker's strategy observed in the illustration shows the attack is not arbitrary – it is focusing on states nearby the goal. We can see an attack of $+1$ on a goal which gives a reward of 2 – this would change the observed reward into 3. A reward value of 3 was not seen during pretraining DPT, and in this round, upon encountering this it provokes undesirable behavior (*stay* at a low-reward state), causing a low episode reward. During the next round of training we find that the DPT has learned to recover from this mistake, and given the same attacker's strategy for that state successfully ignores this attack.

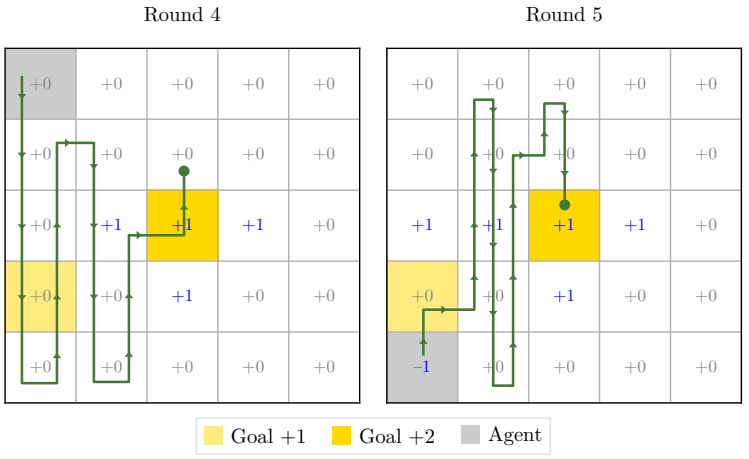

Figure 8: An illustration of the Darkroom2 environment with an attacker's poisoning strategy during a sample training run. Gray and blue numbers -1, $+0$, and $+1$ indicate the attacker's current poisoning strategy. Green path denotes trajectory taken by the agent for that round; green circle indicates the state where the agent chose to stop and exploit the current reward by choosing the stay action.

C.4 AT-DPT test phase

During the test phase, AT-DPT uses the trained parameters θ, and ϕ for the attacker. The procedure for this can be seen in Algorithm 3. We note, that the tables and plots show the performance based on clean rewards, and not $\tilde{r}$.

Algorithm 3 AT-DPT test phase

input: victim π_θ – AT-DPT with params θ
input: attacker $\pi^\dagger_\phi$ with params ϕ, budget B, fraction of steps poisoned ε
Sample M tasks $\{\mathcal{M}_i \sim \mathcal{T}\}_{i=1}^m$ ▷ Differing from the tasks in adversarial training
for all $\mathcal{M}_i$ simultaneously **do**
  $s_0 \sim \rho_{\mathcal{M}_i}$
  $D^\dagger \leftarrow \{\}$
  for $h = 0, \ldots, H-1$ **do**
    select action $a_h \sim \pi_{\theta_n}(\cdot \mid D^\dagger, s_h)$
    $\tilde{r}_h = \begin{cases} r^\dagger_h \sim \pi^\dagger_\phi(\cdot \mid s_h, a_h, \bar{r}_h) & \text{with probability } \varepsilon \\ \bar{r}_h \sim R(\cdot \mid s_h, a_h) & \text{otherwise} \end{cases}$
    $s_{h+1} \sim T(\cdot \mid s_h, a_h)$
    append $(s_h, a_h, \tilde{r}_h, s_{h+1})$ to $D^\dagger$
  end for
end for