A Review of Recent Works on Machine Teaching for Sequential Learners

Deep Patel
Department of Computer Science
University of Wisconsin-Madison
Madison, WI 53706
dbpatel5@wisc.edu

Abstract

We look at some of the recent work on machine teaching for sequential learners. Specifically, we look at a few approaches for teaching multi-armed bandits, Reinforcement Learning (RL) agents that are Q-learners and RL agents that learn from demonstrations – Inverse RL (IRL). Having demonstrated the utility of machine teaching for these sequential learners, we also look at a recent work that finds optimal adversarial attacks to mislead an RL agent into learning desired, harmful policies.

1 Introduction

The classical supervised learning paradigm assumes that a learner is given independent and identically distributed (i.i.d.) data from some unknown, underlying distribution on the data. The task, then, is to pick the hypothesis from a deemed-to-be-suitable hypothesis class that does well on the desired prediction task w.r.t. the underlying data distribution. However, there are many instances in human-computer interaction or computer-computer interaction wherein either entity would wish to adapt their responses based on the other entities response. For instance, a human user would like to nudge the movie recommendation system towards recommending movies of their favourite genre. In response, the recommendation system would optimize the recommendations as per the user's preferences. In the case of a robot navigating through the rubble for disaster relief, the next step and direction to take would depend not only on the past history of its trajectory, but also on the current location, its internal hardware status (e.g. power levels,) and presence of the (expected) nearest distress signal. It's also known that humans provide different demonstrations depending on whether they are teaching or simply performing the task [1]. These sequential decision-making situations can't be modelled under the i.i.d. learning paradigm precisely because of the dependence of next set of actions and observations on the previous ones.

This report will focus on learners for such sequential settings. Specifically, we will look at a few recent works that aim to 'teach' sequential learners such as multi-armed bandits and RL agents that are Q-learners or *learn from demonstrations* (specifically, IRL) by showing appropriate demonstrations or influencing the environment from where the learner gets feedback about the actions they perform. We now formally describe what this general notion of 'teaching' is (referred to as *Machine Teaching)* and follow that up with the notation that will be used throughout the rest of the report.

1.1 Machine Teaching

While a lot of machine learning research has focused on the complexity of learning and designing learning algorithms, there also have been several works that have studied the complexity of *teaching*

and designing teaching algorithms. Intuitively, the idea in machine teaching is to find an *optimal*[1] training set D^* to help learner learn the intended concept or hypothesis (θ^*). Formally, we can state this as a constrained optimization problem [2,3]:

$$\begin{aligned} \min_{D} \quad & \text{TeachingCost}(D) \\ s.t. \quad & \text{TeachingRisk}(\hat{\theta}) \leq \epsilon \\ & \hat{\theta} = \text{MachineLearning}(D) \end{aligned} \tag{1}$$

Here, we searching over all possible training sets and $\hat{\theta}$ is the model/hypothesis learnt by the student/machine learner under dataset D. Teaching risk represents assessment of teacher about how far from their intended model θ^* the learner's model $\hat{\theta}$ is. TeachingCost(D) denotes the teaching effort on the teacher's part (e.g. TeachingCost(D) = $|D|$).

Remark 1.1 *In practice, one often casts a machine learning problem as a risk minimization problem. Thus, the constrained optimization problem above will become a bilevel optimization problem which is hard to solve for, in general, without any further simplifying assumptions.*

Remark 1.2 *Another thing to note is that the training set D obtained by solving this constrained optimization problem need not be i.i.d. Thus, obtaining training sets in this manner could be be helpful especially for tasks beyond supervised learning paradigm such as sequential decision-making tasks.*

Remark 1.3 *One natural question that might arise is how much does the teacher know about the learner and vice versa. One could also ask how reliable the teacher and learner are in what they do. A detailed discussion on this would be beyond the scope of this report but we will certainly specify relevant assumptions in the context of each work that we will be reviewing below.*

Similar to the notion of VC-dimension [4], which provides a measure of *complexity of learning* for different hypothesis classes, there have been some works that have focused on understanding the *complexity of teaching* for various hypothesis classes via various notions of what's referred to as *Teaching Dimension* (TD) [5–8]. For this report, we will restrict our focus on the notion of classical TD [4] even though there are several such notions in the literature as cited above. One can define this classical TD by simply going back to the constrained optimization problem above and choosing the appropriate costs and machine learners. Specifically, if we choose TeachingCost(D) = $|D|$, MachineLearning(D) = Version Space of Learner[2] after seeing dataset D, and TeachingRisk($\hat{\theta}$) $= 0$ if $\hat{\theta} = \{\theta^*\}$ and ∞ otherwise, then the problem reduces to the following max-min optimization routine (this is also referred to as the classical TD for a given hypothesis class $\mathcal{H}$ [5]):

$$TD(\mathcal{H}) = \max_{h \in \mathcal{H}} \left(\min_{\tau \in T(h)} |\tau| \right) \tag{2}$$

where $T(h)$ is the set of examples such that only $h \in \mathcal{H}$ is consistent with $T(h)$. *In plain English, classical TD tells us the minimum number of examples a teacher must provide for the learner to uniquely identify any target hypothesis in the class $\mathcal{H}$.*

Remark 1.4 *But how does one find such optimal teaching sets? For finite hypothesis classes and feature space, Theorem 8 in Goldman et al. [5] showed that solving the max-min problem above (Equation 2) is equivalent to a minimum set covering problem in which there are $|\mathcal{H}| - 1$ objects to be covered and $|\mathcal{X}|$ sets from which to form the covering.*

Remark 1.5 *Just like how the VC-dimension controls the generalization error in PAC-learning framework and helps us assess complexity of learning, in the framework of machine teaching, the classical TD will play an analogous role of helping us assess complexity of teaching a target concept. We will discuss below one of the papers that establishes hardness of teaching an RL agent that's a Q-learner.*

[1] to be defined soon

[2] For a given hypothesis class $\mathcal{H}$, Version Space $V(D) = \{h \in \mathcal{H} \mid h \text{ is consistent with dataset } D\}$

1.2 Establishing common notation

Since we will be discussing several works in this report, we will set up a common, unifying set of notations here to avoid any confusion. We will model the environment as a Markov Decision Process (MDP) and denote it as a tuple $\mathcal{M} = \langle S, A, T, R, \gamma, S_0 \rangle$. Here, S is state space, A is the action space, $T : S \times A \to [0, 1]$ is the transition function, $R : S \to \mathbb{R}$ is the reward function, $\gamma \in [0, 1)$ is the discounting factor, and S_0 is the initial state distribution. We will define a policy $\pi : S \to \Delta^A$ as a mapping from state space to a probability simplex over the action space. The value function for a given policy π is defined as:

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) | \pi, s_0 = s\right] \quad \forall s \in S \tag{3}$$

Similarly, the Q-value of a state-action pair is defined as:

$$Q^\pi(s, a) = R(s) + \gamma \mathbb{E}_{s' \sim T(\cdot|s,a)} \left[V^\pi(s')\right] \quad \forall s \in S \;\; \forall a \in A \tag{4}$$

Finally, we will denote the optimal Q-value function as $Q^*(s, a) = \max_\pi Q^\pi(s, a) \;\; \forall s \in S \;\; \forall a \in A$

2 Teaching Active Sequential Learners

Let's go back to the example of movie recommendation system. Typically, the recommendation system will rely on user's response to the system's queries such as *Did you like this movie?* in order to build a personalized recommendation profile for the user. But the users and their preferences can keep changing over time and these users would expect the recommender system to adapt to changing user-base and their preferences. From the user's point-of-view, the challenge is to provide the right set of responses to the ever-changing recommender system so that it continues to provide relevant recommendations to them. Towards this, Peltola et al. [13] formulate the problem of teaching Multi-Armed Bandits (MABs) as a planning problem in a MDP.

One thing to note though is that we are considering active sequential learners here. Formally, an active sequential learner is defined by

(i) Machine learning model relating responses y to inputs x by parameterized function $y = f_\theta(x)$

(ii) deterministic learning algorithm, fitting parameters θ given a dataset $D = \{(x_i, y_i)\}_{i=1}^n$

(iii) query function that (possibly stochastically) chooses an input point x to query for response y in order to maximize learner's utility[3].

Thus, for $t = 1, \ldots, T$, the learner 1.) uses query function to obtain x_t, 2.) get response y_t for x_t from teacher or environment, and 3.) update training set $D_t = D_{t-1} \cup \{(x_t, y_t)\}$ and model accordingly. These dynamics define history

$$h_t \triangleq (x_1, y_1, x_2, y_2, \ldots, x_t) \;\; \forall t \tag{5}$$

Remark 2.1 *Since we are considering active sequential learners here, the teacher only provides the response/reward y for the learner query x. Thus, unlike in conventional machine teaching settings, the teacher is not in control of designing all of the learning data for the learner.*

Assumptions about the learner: The paper considers Bernoulli K-MAB learners. At each iteration t, learner chooses arm $i_t \in \{1, \ldots, K\}$ and receives stochastic reward $y_t \in \{0, 1\}$ from that chosen arm. Goal of the learner is to maximize the expected accumulated reward $R_T = \mathbb{E}[\sum_{t=1}^T y_t]$. Furthermore, we assume that the learner associates each arm k with a feature vector $x_k \in \mathbb{R}^M$ and models the rewards as

$$p(y_t|\mu_{i_t}) \triangleq \text{Bernoulli}(y_t|\mu_{i_t}) \tag{6}$$

with reward probabilities $\mu_k = \text{sigmoid}(x_k^T \theta)$ where $\theta \in \mathbb{R}^M$. The learner uses bandit arm selection strategy to select the next arm to query about. Any popular bandit algorithms can be employed for this.

[3] or minimize loss or minimize regret

Assumptions about the environment and the teacher: Recall that the role of the teacher is to provide reward y_t based on what arm learner picks at round t. We will assume that the teacher knows the true parameters θ_i^* $\forall i \in [K]$ and that the teacher wants to maximize the learner's expected accumulated reward.

Problem Formulation: We can formulate the teaching problem here as a planning problem for an appropriately defined MDP. Specifically, we define the MDP as $\mathcal{M} = \langle \mathcal{H}, \mathcal{Y}, \mathcal{T}, \mathcal{R}, \gamma \rangle$ where states $h_T \in \mathcal{H}$ correspond to the history sequences (Equation 5), actions are the responses $y_t \in \mathcal{Y} = \{0, 1\}$, transition probabilities $p(h_{t+1}|h_t, y_t) \in \mathcal{T}$ defined by the learner's sequential dynamics, rewards $R_t(h_t) \in \mathcal{R}$ define teacher's goal and $\gamma \in [0, 1)$ is a discounting factor.

For the problem at hand, the teacher's policy $\pi : \mathcal{H} \to \Delta^{\mathcal{Y}}$ will be $\pi(h_t) = p(\cdot|h_t, \pi) \in \Delta^{\mathcal{Y}}$ $\forall h_t$. As mentioned above, the goal of the teacher is to choose actions y_t to maximize the cumulative reward. Thus, the value function for this MDP can be defined as $V^{\pi}(h_t) = \mathbb{E}[\sum_{t=1}^{T} \gamma_{t-1} R_t(h_t)]$ where T is the teacher's planning horizon and expectation is over stochasticity in learner's queries and teacher's policy.

Thanks to this reduction to a planning problem, we can now use any planning algorithms to teach the MAB so as to maximize their cumulative reward.

3 Teaching an RL agent by demonstrations: Inverse RL

Now that we have some handle on teaching MABs, the next natural question would be: Can we teach sequential learners for whom the entire MDP may not be known. For instance, in robotics, one might know everything in the MDP except for the reward function. In such cases, what's often done is to teach the robot via Teaching-by-Demonstrations (TbD). That is, the robot estimates the reward function based on the set of demonstrations provided to it and eventually obtain a near-optimal policy. Brown et al. [9] show that one can teach in this framework; specifically, Inverse RL. A benefit of finding optimal teaching set of demonstrations for IRL is that this way we don't have to follow the i.i.d. assumption made my many IRL algorithms. Instead, we can focus on finding the most informative set of demonstrations for the RL agent. Since optimal teaching is typically more sample-efficient than *Active Learning*, another benefit of the teaching approach is that it provides a lower bound on number of queries needed to learn policies in the case of *Active Inverse RL*. Now let's look at the assumptions and proposed method in this work.

Assumptions about the environment: Assume that the reward is a linear function in some feature space. That is, $\exists \phi : S \to \mathbb{R}^k$ so that $R(s) = w^T \phi(s)$[4]. Thus, we can write the expected discounted return for any policy π as

$$\rho(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t w^T \phi(s_t) \mid \pi\right] = w^T \mu_\pi \tag{7}$$

where $\mu_\pi = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi\right]$.

Assumptions about the teacher: The teacher has the ability to demonstrate state-action pairs (s, a) by executing the optimal policy π^* that it wants the RL agent to learn.

Remark 3.1 *Simply finding set of demonstrations such that the IRL agent learns reward function R^*[5] within ϵ teaching risk is not a good idea. This is because there can be infinite reward functions that explain any optimal policy [12].*

Problem Formulation: Determine minimal set of observations so that learner finds reward function that yields an optimal policy with similar performance to that of teacher's policy under reward function R^*. This performance is defined as a policy loss of an estimated weight vector $\hat{w}$ compared with the true weight vector w^*:

$$\text{Loss}(\hat{w}, w^*) \triangleq w^{*T}(\mu_{\pi^*} - \mu_{\hat{\pi}}) \tag{8}$$

where π^* is the optimal policy under w^* and $\hat{\pi}$ is the optimal policy under $\hat{w}$. The policy loss defined above in Equation 8 is essentially the difference between the expected rewards under the teacher's

[4]This is a common assumption in the literature [10, 11]

[5]this is the true reward function of the underlying MDP

policy π^* and that under the learner's policy $\hat{w}$, when evaluated under the teacher's (i.e. true) reward function $R^*(\cdot) = {w^*}^T\phi(\cdot)$. Thus, we would like to find a set of demonstrations, D, that minimizes the following optimization problem:

$$\begin{aligned} \min_{D} \quad & |D| \\ s.t. \quad & \text{Loss}(\hat{w}, w^*) = 0 \\ & \hat{\pi} = \text{RL}(\hat{w}) \\ & \hat{w} = \text{IRL}(D) \end{aligned} \tag{9}$$

where $|D|$ is the number of state-action pairs in D, $\hat{\pi}$ is the optimal policy under $\hat{w}$ obtainable by RL algorithms and $\hat{w}$ is the reward 'function' estimated by the learner using IRL algorithms.

Remark 3.2 *A brute-force approach to solve the optimization problem defined above requires searching over the set of all possible demonstrations. And for each of those candidate demonstration sets, one needs to solve the RL and IRL problem to check feasibility of constraints.*

We can avoid searching over the space of all possible demonstrations by restricting our attention to *Behaviour Equivalence Classes* (BECs) for the target policy. BEC of a policy π is defined as the set of reward functions under which the policy π is optimal.

$$\text{BEC}(\pi) = \{w \in \mathbb{R}^k \mid \pi \text{ is optimal w.r.t. } R(\cdot) = w^T\phi(\cdot)\} \tag{10}$$

Given the linear nature of reward functions, BEC(π) can be characterized by an intersection of half-spaces as shown below in Theorem 3.1.

Theorem 3.1 (Ng and Russell 2000, [12) *Given an MDP, BEC(π) is given by the intersection of half-spaces:*

$$\begin{aligned} w^T(\mu_\pi^{(s,a)} - \mu_\pi^{(s,b)}) \geq 0 \\ \forall a \in \arg\max_{a' \in A} Q^*(s, a'), b \in A, s \in S \end{aligned} \tag{11}$$

where w defines reward function R and $\mu_\pi^{(s,a)}$ is the vector of expected feature counts by taking action a in state s and following π thereafter

$$\mu_\pi^{(s,a)} = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t) \mid \pi, s_0 = s, a_0 = a\right] \tag{12}$$

Proof Follows directly from the definition of $Q^*(\cdot, \cdot)$ and definition of expected discounted returns.

Corollary 1 *BEC for a set of demonstrations, D, from a policy π – BEC($D|\pi$) – is given by the following intersection of half-spaces:*

$$w^T(\mu_\pi^{(s,a)} - \mu_\pi^{(s,b)}) \geq 0 \;\; \forall (s, a) \in D, b \in A \tag{13}$$

Proof Proof is the same as that for Theorem 3.1 except that we now look only at those half-spaces corresponding to (s, a) pairs in the demonstration set D.

Coming back to the optimization problem we are trying to solve, restricting our attention to BEC(π) helps reduce our search space because we incur zero policy loss (Loss($\hat{w}, w^*$) = 0) by using a non-degenerate (i.e. non-constant) vector $w \in$ BEC(π^*) rather than using w^* for policy optimization. This is true because of the definition of BEC(π^*) (Equation 10 above). Thus, we focus on finding the demonstration set D such that the weight vector learned through IRL belongs to BEC(π^*). An obvious question is what if IRL doesn't return non-degenerate vector w. For standard IRL methods, we are guaranteed to obtain non-degenerate reward functions [14, 15]. We can further ensure feasibility by looking for demonstration sets D such that BEC($D|\pi^*$) = BEC(π^*).

Given this reduced search space, thanks to BEC($D|\pi^*$) (= BEC(π^*)), solving optimization problem as stated in Equation 9 above is akin to solving the set cover problem. In summary, given π^*, the optimal policy under the teacher's reward function w^*, proposed approach is as follows:

1. Solve for feature expectations $\mu_{\pi^*}^{(s,a)}$ by solving the following Bellman equation (efficiently doable):
$$\mu_{\pi^*}^{(s,a)} = \phi(s) + \gamma \mathbb{E}_{s'|a}[\mu_{\pi^*}^{(s')}] \tag{14}$$
where $\mu_{\pi^*}^{(s)} = \phi(s) + \gamma \mathbb{E}_{s'|\pi^*(s)}[\mu_{\pi^*}^{(s')}]$
2. Use these feature expectations to find the half-space constraints for BEC(π^*) using Theorem 3.1
3. Use Linear Programming (LP) to remove any redundant half-space constraints from BEC(π^*)
4. Generate candidate demonstrations under π^* from each starting state and calculate their corresponding half-space unit normal vectors using Corollary 1. This ensures BEC($D|\pi^*$) = BEC(π^*).
5. Greedily cover half-spaces in BEC(π^*) by sequentially picking candidate demonstrations that covers the most number of uncovered half-spaces.

It can be shown[6] that this proposed set-covering-based approach always terminates and that it's an $(1 - 1/e)$-approximation of the minimum number of demonstrations needed to fully define BEC(π^*).

4 Teaching an RL agent by reinforcement: Q-learning

Having looked at Inverse RL, we can now ask if we can teach an RL agent in the more general setting of Teaching-by-Reinforcement (TbR) wherein exploration by learner is still necessary. Zhang et al. [16] explore precisely this paradigm with a focus on Q-learning agents[7].

Assumptions about the environment: We will assume that we have an episodic MDP. Thus, MDP $\mathcal{M} = \langle S, A, R, P, S_0, H \rangle$ where H is the episode length. The discounting factor γ is not relevant in the episodic setting. Training and test phases are separated. In the test phase, output policy is fixed and evaluated.

Assumptions about the learner: During the training phase, learner interacts with the MDP for a finite number of episodes and outputs a policy at the end. The learner is assumed to belong to the family of Q-learning agent that includes ϵ-greedy Q-learning and variants of UCB like UCB-H and UCB-B.

Assumptions about the teacher: The teacher can decide when the training phase terminates. Thus, teaching can be declared complete as soon as the target policy is learnt. We do NOT require convergence of estimated Q-function to the true Q-function wr.t. the deployed policy. Due to limited space, we will only discuss the case where teacher is quite strong[8] and learner employing ϵ-greedy strategy (details below; Equation 15). That is, the teacher can generate arbitrary transitions $(s_t, r_t, s_{t+1}) \in S \times \mathbb{R} \times S$. Moreover, these transitions and action overrides need not obey the MDP at all.

Problem Formulation: Say the teacher wants to teach a target policy $\pi^\dagger$ which may or may not be equal to the optimal policy π^* under the true, underlying MDP $\mathcal{M}$. Since the teacher can't override learners actions a_t, teacher can't teach the desired action $\pi^\dagger(s)$ in single visit to state s. However, the teacher can still generate arbitrary transitions and rewards. Say, the learner employs ϵ-greedy learning strategy, then it will choose the next action as per the following rule:

$$\pi_t(s) \triangleq \begin{cases} a^* = \arg\max_a Q_t(s,a) & w.p. 1-\epsilon \\ unif(A \setminus \{a^*\}) & w.p. \epsilon \end{cases} \tag{15}$$

To add to this, assume that Q_0 is such that $Q_0(s, \pi^\dagger(s))$ is lowest among all actions. If the learner is greedy with $\epsilon = 0$ in Equation 15, then teacher will need to visit the state s for $|A| - 1$ times and generate punishing rewards r_t to convince the learner that the top non-target action (as per the Q-table) is worse than the desired action $\pi^\dagger(s)$. For a learner who operates with $\epsilon > 0$, it may perform

[6] Proposition 2 and 3, Appendix E of the paper

[7] but the results can be generalized to other RL agents that are not Q-learners, e.g. SARSA

[8] 'Level-2'-type as per the paper terminology

$\pi^\dagger(s)$ with non-zero probability and teacher can, in that case, generate a huge reward to promote this action and complete teaching for state s. It can be shown[9] that for any ϵ it still takes $|A|-1$ visits on average to a state s to teach a desired action in the worst case. We are done now because we can follow this greedy heuristic to teach desired action $\pi^\dagger(s)$ for every state s. This also tells us that the Teaching Dimension for such teachers is $|S|(|A|-1)$ as the learner might have to visit each state atmost $|A|-1$ times to learn the target action as per the teacher's desired policy $\pi^\dagger$.

5 Teacher can lie to hurt the learner

All this while we have considered teachers that want to help the learner maximize their cumulative rewards. However, there can be situations wherein an RL agent experiences security threats by an attacker that poisons its learning environment by manipulating rewards and transition dynamics at the time of training. Given the use of RL algorithms in sensitive applications such as cyber-physical systems [17] and personal assistive devices [18], it's important to understand the vulnerabilities of RL systems under poisoning attacks. Rakhsha et al. [19] propose an optimization framework for finding optimal attacks for RL agents. These poisoning attacks are mathematically equivalent to machine teaching with the teacher being an adversary/attacker. Due to space reasons, we will only consider offline attacks (details in assumptions about the teacher) wherein the RL agent learns optimal policy given an MDP via planning algorithms.

Assumptions about the environment: We will allow the discounting factor $\gamma \in [0,1]$ to allow for undiscounted rewards as well. We will only allow deterministic policies $\pi : S \to A$ and we will assume that the MDP is such that every policy π has a state distribution μ^π defined as

$$\mu^\pi(s) = \begin{cases} (1-\gamma)\sum_{t=0}^{\infty}\gamma^t T[s_t = s|s_0 \sim S_0, \pi] & if \gamma < 1 \\ \lim_{N\to\infty}\frac{1}{N}\sum_{t=0}^{N-1} T[s_t = s|s_0 \sim S_0, \pi] & if \gamma = 1 \end{cases} \tag{16}$$

such that $\mu^\pi(s) > 0$ for every state s. For $\gamma = 1$, $\mu^\pi(s)$ corresponds to the stationary state distribution induced by the policy π whereas for $\gamma < 1$, $\mu^\pi(s)$ corresponds to the discounted state distribution induced by the policy π. State distribution μ^π satisfies Bellman constraint:

$$\mu^\pi(s) = (1-\gamma)\cdot S_0(s) + \gamma \cdot \sum_{s'} T(s'|s, \pi(s)) \cdot \mu^\pi(s') \tag{17}$$

The expected average and discounted reward of policy π are respectively equal to

$$\lim_{N\to\infty}\frac{1}{N}\mathbb{E}\left[\sum_{t=0}^{N-1} R(s_t, a_t|s_0 \sim S_0, \pi\right] \text{ and } E\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, a_t)|s_0 \sim S_0, \pi\right] \tag{18}$$

where the expectation is over rewards received by agent starting from $s_0 \sim S_0$ and following policy π thereafter. We define the score of a policy π as

$$\rho(\pi, \mathcal{M}, S_0) \triangleq \sum_s \mu^\pi(s) \cdot R(s, \pi(s)) \tag{19}$$

Thus, a policy π^* is optimal if for every other deterministic policy π we have $\rho^{\pi^*} \geq \rho^\pi$ and ϵ-robust optimal if $\rho^{\pi^*} \geq \rho^\pi + \epsilon$. For a policy π, Q- and V-values are defined as follows with the corresponding Bellman equality:

$$Q^\pi(s,a) = \mathbb{E}\left[\gamma^t \cdot (R(s_t, a_t) - \rho^\pi)|s_0 = s, a_0 = a, \pi\right] \tag{20}$$

$$V^\pi(s) = Q^\pi(s, \pi(s)) \tag{21}$$

$$Q^\pi(s,a) = R(s,a) - \rho^\pi + \gamma \cdot \sum_{s' \in S} T(s'|s,a) \cdot V^\pi(s') \tag{22}$$

Assumptions about the learner: In offline setting, the RL agent is given an MDP $\mathcal{M}$ and it will learn a deterministic policy $\pi^* \in \arg\max_\pi \rho(\pi, \mathcal{M}, S_0)$ via a planning algorithm.

Assumptions about the teacher: Since we will only be discussing offline attacks in this report, we assume that the teacher/adversary poisons the environment by manipulating the reward function and

[9] Lemma 2 in the paper

the transition dynamics. Thus, if the environment MDP is $\bar{\mathcal{M}} = \langle S, A, \bar{R}, \bar{T} \rangle$, then the teacher/adversary changes it to $\widehat{\mathcal{M}} = \langle S, A, \hat{R}, \hat{T} \rangle$. The learner is unaware of this poisoning of environment and does planning for this new MDP $\widehat{\mathcal{M}}$ so as to mislead the learner into learning optimal policy $\pi^\dagger$ under this new, poisoned MDP $\widehat{\mathcal{M}}$.

Problem Formulation: Given a margin ϵ, the goal of the teacher/adversary is to poison the rewards and transition dynamics such that $\pi^\dagger$ is ϵ-robust optimal in the poisoned MDP $\widehat{\mathcal{M}}$. That is,

$$\rho(\pi^\dagger, \widehat{\mathcal{M}}, S_0) \geq \rho(\pi, \widehat{\mathcal{M}}, S_0) + \epsilon \;\; \forall \pi \neq \pi^\dagger \tag{23}$$

We also want to take into account the cost of the attack for the adversary. We do so by defining it for each state-action pair and by splitting the cost into cost for changing the rewards and cost for changining the transition dynamics, i.e. $|\hat{R}(s,a) - \bar{R}(s,a)|$ and $\sum_{s'} |\hat{T}(s'|s,a) - \bar{T}(s'|s,a)|$ resp. We combine the two costs as a weighted sum to define the total cost of an attack to the teacher-attacker as follows:

$$\text{cost}(\widehat{\mathcal{M}}, \bar{\mathcal{M}}, C_r, C_p, p) = \Big(\sum_{s,a} \big(C_r \cdot |\hat{R}(s,a) - \bar{R}(s,a)| + C_p \cdot \sum_{s'} |\hat{T}(s'|s,a) - \bar{T}(s'|s,a)| \cdot \big)^p \Big)^{1/p} \tag{24}$$

Ideally, to find MDP $\widehat{\mathcal{M}}$ for which target policy $\pi^\dagger$ is ϵ-robust optimal, we could use constraints as defined in Equation 23. The problem is that there are $|A|^{|S|} - 1$ possible policies satisfying that constraint of robust optimality (exponential in $|S|$). What can be done however is to realise that it's enough to satisfy these constraints for $|S| \cdot |A| - |S|$ policies that are 'neighbours' to the teacher's target policy.

Definition 1 *For a policy π, it's neighbour policy $\pi\{s:a\}$ is defined as*

$$\pi\{s:a\}(x) = \begin{cases} \pi(x), & x \neq s \\ a, & x = s \end{cases} \tag{25}$$

It suffices to check only for these neighbour policies that differ only one on state because, as stated in Lemma 1, sub-optimality of any policy can be checked by at its neighbour policies alone.

Lemma 1 *Policy π is ϵ-robust optimal iff we have $\rho^\pi \geq \rho^{\pi\{s:a\}} + \epsilon$ for every state s and action $a \neq \pi(s)$.*

Thus, the problem of poisoning attack can be formulated as finding the MDP $\widehat{\mathcal{M}}$ such that cost to the teacher-attacker is minimized whilst ensuring that the teacher-attacker's target policy $\pi^\dagger$ and its neighbouring policies still respect the MDP structure as defined by Equation 17. Formally, it can be stated as:

$$\begin{aligned}
\min_{\mathcal{M}, R, T, \mu^{\pi^\dagger}, \mu^{\pi^\dagger\{s;a\}}} \quad & \text{cost}(\mathcal{M}, \bar{\mathcal{M}}, C_r, C_p, p) \\
\text{s.t. } & \mu^{\pi^\dagger} \text{ and } T \text{ satisfy 17} \\
\forall s, a \neq \pi^\dagger(s) : & \mu^{\pi^\dagger\{s;a\}} \text{ and } T \text{ satisfy 17} \\
\forall s, a \neq \pi^\dagger(s) : & \sum_{s'} \mu^{\pi^\dagger\{s;a\}}(s') R(s', \pi^\dagger(s')) \geq \sum_{s'} \mu^{\pi^\dagger\{s;a\}}(s') R(s', \mu^{\pi^\dagger\{s;a\}}(s') + \epsilon \\
\forall s, a, s' : & T(s'|s,a) \geq \delta \bar{T}(s'|s,a) \\
& \mathcal{M} = \langle S, A, R, T, \gamma \rangle
\end{aligned} \tag{26}$$

Here, the third constraint corresponds to ensuring ϵ-robust optimality as necessitated in Lemma 1 above whereas the fourth constraint with $\delta \in (0, 1]$ specifies how much the teacher-attacker can redue the original values of transition probabilities. Although this problem can be non-convex in general, it can be reduced to a convex optimization problem under some assumptions. The paper talks about how to efficiently solve this optimization problem 26 for the offline attacks and develops a similar framework for the online attacks as well. We refer the interested reader to the paper for further details.

References

[1] Ho, M. K., Littman, M., MacGlashan, J., Cushman, F., & Austerweil, J. L. (2016). Showing versus doing: Teaching by demonstration. Advances in Neural Information Processing Systems, 29.

[2] Zhu, X., Singla, A., Zilles, S., & Rafferty, A. N. (2018). An overview of machine teaching. *arXiv preprint arXiv:1801.05927*.

[3] Zhu, X. (2015). Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 29, No. 1).

[4] Vapnik, V. (1968). On the uniform convergence of relative frequencies of events to their probabilities. In *Doklady Akademii Nauk USSR* (Vol. 181, No. 4, pp. 781-787).

[5] Goldman, S. A., & Kearns, M. J. (1995). On the complexity of teaching. *Journal of Computer and System Sciences*, 50(1), 20-31.

[6] Zilles, S., Lange, S., Holte, R., & Zinkevich, M. (2008). Teaching Dimensions based on Cooperative Learning. In *COLT* (pp. 135-146).

[7] Doliwa, T., Fan, G., Simon, H. U., & Zilles, S. (2014). Recursive Teaching Dimension, VC-dimension and Sample Compression. *The Journal of Machine Learning Research*, 15(1), 3107-3131.

[8] Gao, Z., Ries, C., & Simon, H. U. (2017). Preference-based teaching. *Journal of Machine Learning Research*, 18(31), 1-32.

[9] Brown, D. S., & Niekum, S. (2019). Machine Teaching for Inverse Reinforcement Learning: Algorithms and Applications. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 33, No. 01, pp. 7749-7758).

[10] Ziebart, B. D., Maas, A. L., Bagnell, J. A., & Dey, A. K. (2008). Maximum entropy Inverse Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 8, pp. 1433-1438).

[11] Pirotta, M., & Restelli, M. (2016). Inverse reinforcement learning through policy gradient minimization. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 30, No. 1).

[12] Ng, A. Y., & Russell, S. J. (2000). Algorithms for Inverse Reinforcement Learning. In *Proceedings of the Seventeenth International Conference on Machine Learning* (pp. 663-670).

[13] Peltola, T., Çelikok, M. M., Daee, P., & Kaski, S. (2019). Machine Teaching of Active Sequential Learners. *Advances in Neural Information Processing Systems*, 32.

[14] Arora, S., & Doshi, P. (2021). A Survey of Inverse Reinforcement Learning: Challenges, Methods and Progress. *Artificial Intelligence*, 297, 103500.

[15] Zhifei, S., & Joo, E. M. (2012). A survey of inverse reinforcement learning techniques. *International Journal of Intelligent Computing and Cybernetics*, 5(3), 293-311.

[16] Zhang, X., Bharti, S., Ma, Y., Singla, A., & Zhu, X. (2021). The Sample Complexity of Teaching-by-Reinforcement on Q-Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 35, No. 12, pp. 10939-10947).

[17] Li, C., & Qiu, M. (2019). *Reinforcement learning for cyber-physical systems: with cybersecurity case studies*. Chapman and Hall/CRC.

[18] Rybski, P. E., Yoon, K., Stolarz, J., & Veloso, M. M. (2007). Interactive robot task training through dialog and demonstration. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction* (pp. 49-56).

[19] Rakhsha, A., Radanovic, G., Devidze, R., Zhu, X., & Singla, A. (2021). Policy Teaching in Reinforcement Learning via Environment Poisoning Attacks. *Journal of Machine Learning Research*, 22(210), 1-45.