Curriculum Design for Teaching via Demonstrations: Theory and Applications

Anonymous Author(s)
Affiliation
Address
email

Abstract

We consider the problem of teaching via demonstrations in sequential decision-making settings. In particular, we study how to design a personalized curriculum over demonstrations to speed up the learner's convergence. We provide a unified curriculum strategy for two popular learner models: Maximum Causal Entropy Inverse Reinforcement Learning (MaxEnt-IRL) and Cross-Entropy Behavioral Cloning (CrossEnt-BC). Our unified strategy induces a ranking over demonstrations based on a notion of difficulty scores computed w.r.t. the teacher's optimal policy and the learner's current policy. Compared to the state of the art, our strategy doesn't require access to the learner's internal dynamics and still enjoys similar convergence guarantees under mild technical conditions. Furthermore, we adapt our curriculum strategy to teach a learner using domain knowledge in the form of task-specific difficulty scores when the teacher's optimal policy is unknown. Experiments on a car driving simulator environment and shortest path problems in a grid-world environment demonstrate the effectiveness of our proposed curriculum strategy.

1 Introduction

Learning from Demonstrations (LfD) is a paradigm in which a learner acquires a new set of skills by imitating a teacher's behavior. The importance of LfD is realized in real-world applications where the desired behavior cannot be explicitly defined but can be demonstrated easily. These applications include the settings involving both human-to-machine interaction [1–4], and human-to-human interaction [5, 6]. The two most popular approaches to LfD are Behavioral Cloning (BC) [7], and Inverse Reinforcement Learning (IRL) [8]. BC algorithms aim to directly match the behavior of the teacher using supervised learning methods. IRL algorithms operate in a two-step approach: first, a reward function explaining the teacher's behavior is inferred; then, the learner adopts a policy corresponding to the inferred reward.

In the literature, LfD has been extensively studied from the learner's point of view to design efficient learning algorithms [9–15]. However, much less work is done from the teacher's point of view to reduce the number of demonstrations required to achieve the learning objective. In this paper, we focus on the problem of Teaching via Demonstrations (TvD), where a helpful teacher assists the learner in converging quickly by designing a personalized curriculum [16–20]. Despite a substantial amount of work on curriculum design for reinforcement learning agents [21–27], curriculum design for LfD is much less investigated.

In the literature, curriculum design for IRL learners has been studied in two concrete settings: non-interactive and interactive. In the non-interactive setting [17, 18], the teacher provides a near-optimal set of demonstrations as a single batch. Their curriculum strategies do not incorporate any feedback from the learner, hence unable to adapt the teaching to the learner's progress. In the interactive

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

setting [28], the teacher can leverage the learner's progress to adaptively choose the next demonstrations to accelerate the learning process. We focus on designing a personalized curriculum of demonstrations in the interactive teaching setting. In our approach, we overcome the following two limitations of the state of the art: (i) they are not applicable to BC learner models, and (ii) they require access to the learner's internal dynamics which is often unknown in practical applications.

Our approach is motivated by works on curriculum design for supervised learning and reinforcement learning algorithms that use a ranking over the training examples using a difficulty score [29–35]. In particular, our work is inspired by theoretical results on curriculum learning for linear regression models [32]. We define difficulty scores for any demonstration based on the teacher's optimal policy and the learner's current policy. We then study the differential effect of the difficulty scores on the learning progress for two popular LfD learners: Maximum Causal Entropy Inverse Reinforcement Learning (MaxEnt-IRL) [10] and Cross-Entropy loss function based Behavioral Cloning (CrossEnt-BC) [36]. Our main contributions are as follows:

1. Our analysis for both MaxEnt-IRL and CrossEnt-BC learners leads to a unified curriculum strategy, i.e., a preference ranking over demonstrations. This ranking is obtained based on the ratio between the demonstration's likelihood under the teacher's optimal policy and the learner's current policy. Experiments on a synthetic car driving environment validate our curriculum strategy.
2. For the MaxEnt-IRL learner, we prove that our curriculum strategy achieves a linear convergence rate (under some mild technical conditions), notably without requiring access to the learner's internal dynamics.
3. We adapt our curriculum strategy to utilize domain knowledge, in the form of task-specific difficulty scores, when the teacher's optimal policy is unknown. As a proof of concept, we show that our strategy accelerates the learning process on shortest path problems in a grid-world environment.

2 Formal Problem Setup

Environment. We consider an environment defined as a Markov Decision Process (MDP) $\mathcal{M} := \left(\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, P_0, R^E\right)$, where the state and action spaces are denoted by $\mathcal{S}$ and $\mathcal{A}$ respectively, $\mathcal{T} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0,1]$ is the transition dynamics, γ is the discounting factor, and $P_0 : \mathcal{S} \to [0,1]$ is an initial distribution over states $\mathcal{S}$. A policy $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$ is a mapping from a state to a probability distribution over actions. The underlying reward function is given by $R^E : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$.

Teacher-learner interaction. We consider a setting with two agents: a teacher and a sequential learner. The teacher has access to the full MDP $\mathcal{M}$ and has a *target policy* π^E computed as a near-optimal policy w.r.t. R^E. The learner knows the MDP $\mathcal{M}$ but not the reward function R^E, i.e., has only access to $\mathcal{M} \setminus R^E$.

The teacher's goal is to provide an informative sequence of demonstrations to teach the policy π^E to the learner. Here, a teacher's demonstration $\xi = \left\{\left(s_\tau^\xi, a_\tau^\xi\right)\right\}_{\tau=0,1,\ldots}$ is obtained by first choosing an initial state $s_0^\xi \in \mathcal{S}$ (where $P_0(s_0^\xi) > 0$) and then choosing a trajectory, sequence of state-action pairs, obtained by executing the policy π^E in the MDP $\mathcal{M}$. The interaction between the teacher and the learner is formally described in Algorithm 1. For simplicity, we assume that the teacher directly observes the learner's policy π_t^L at any time t. In practice, the teacher could approximately infer the policy π_t^L by probing the learner and using Monte Carlo methods.

Generic learner model. Here, we describe a generic learner update rule for Algorithm 1. The learner searches for a policy in the following parameterized policy space: $\Pi_\Theta := \{\pi_\theta : \mathcal{S} \times \mathcal{A} \to [0,1]\,,\ \text{where } \theta \in \Theta\}$. For the policy search, the learner sequentially minimizes a loss function ℓ that depends on the policy parameter θ and the demonstration $\xi = \left\{\left(s_\tau^\xi, a_\tau^\xi\right)\right\}_\tau$ provided by the teacher. More concretely, we consider $\ell\left(\xi, \theta\right) := -\log \mathbb{P}\left(\xi|\theta\right)$, where $\mathbb{P}\left(\xi|\theta\right) = P_0(s_0^\xi) \cdot \prod_\tau \pi_\theta\left(a_\tau^\xi|s_\tau^\xi\right) \cdot \mathcal{T}\left(s_{\tau+1}^\xi|s_\tau^\xi, a_\tau^\xi\right)$ is the likelihood (probability) of the demonstration ξ under policy π_θ in the MDP $\mathcal{M}$. At time t, upon receiving a demonstration ξ_t provided by the teacher, the learner performs the following online projected gradient descent update: $\theta_{t+1} \leftarrow \mathrm{Proj}_\Theta\left[\theta_t - \eta_t g_t\right]$, where η_t is the learning rate, and $g_t = \left[\nabla_\theta \ell\left(\xi_t, \theta\right)\right]_{\theta=\theta_t}$. Note that the parameter θ_1 reflects the initial

Algorithm 1 Teacher-Learner Interaction

Initialization: Initial knowledge of learner π_1^L.
for $t = 1, 2, \ldots$ **do**
  Teacher observes the learner's current policy π_t^L.
  Teacher provides demonstration ξ_t to the learner.
  Learner updates it's policy to π_{t+1}^L using ξ_t.

knowledge of the learner. Given the learner's current parameter θ_t at time t, the learner's policy is defined as $\pi_t^L := \pi_{\theta_t}$.

Teaching objective. For any policy π, the value (total expected reward) of π in the MDP $\mathcal{M}$ is defined as $V^\pi := \sum_{s,a} \sum_{\tau=0}^{\infty} \gamma^\tau \cdot \mathbb{P}\left\{S_\tau = s \mid \pi, \mathcal{M}\right\} \cdot \pi\left(a \mid s\right) \cdot R^E\left(s, a\right)$, where $\mathbb{P}\left\{S_\tau = s \mid \pi, \mathcal{M}\right\}$ denotes the probability of visiting the state s after τ steps by following the policy π. Let π^L denote the learner's final policy at the end of teaching. The performance of the policy π^L (w.r.t. π^E) in $\mathcal{M}$ can be evaluated via $\left|V^{\pi^E} - V^{\pi^L}\right|$ [9, 37]. The teaching objective is to ensure that the learner's final policy *ϵ-approximates* the teacher's policy, i.e., $\left|V^{\pi^E} - V^{\pi^L}\right| \leq \epsilon$. The teacher aims to provide an optimized sequence of demonstrations $\{\xi_t\}_{t=1,2,\ldots}$ to the learner to achieve the teaching objective. The teacher's performance is then measured by the number of demonstrations required to achieve this objective. Based on existing work [28, 38], we assume that $\exists\ \theta^* \in \Theta$ such that $\pi^E = \pi_{\theta^*}$, we refer to θ^* as the *target teaching parameter*, and that a smoothness condition holds in the policy parameter space: $|V^{\pi_\theta} - V^{\pi_{\theta'}}| \leq \mathcal{O}\left(f\left(\|\theta - \theta'\|\right)\right), \forall \theta, \theta' \in \Theta$. Then, the teaching objective in terms of V^π convergence can be reduced to the convergence in the parameter space, i.e., we can focus on the quantity $\|\theta^* - \theta_t\|$.

3 Curriculum Design using Difficulty Scores

In this section, we introduce our proposed curriculum strategy. The teacher uses our strategy to select the demonstration ξ_t to provide to the learner at each time step t, as given in Algorithm 1. Our proposed approach is based on the concept of difficulty scores. The difficulty score for any demonstration ξ from the teacher's and learner's perspective (at any time t) is denoted by $\Psi^E(\xi)$ and $\Psi_t^L(\xi)$ respectively. Our curriculum strategy is then given by:

$$\xi_t \leftarrow \arg\max_{\xi} \frac{\Psi_t^L\left(\xi\right)}{\Psi^E\left(\xi\right)}. \tag{1}$$

We define difficulty scores and propose curriculum algorithms for two distinct teaching settings.

Teaching with optimal policy π^E: Here, we consider a teacher with complete knowledge of π^E, as considered in prior work on sequential teaching [28, 38]. This setting allows us to study lower bounds for teaching complexity and is also relevant for teaching human learners in pedagogical applications. We begin by assigning a difficulty score $\Psi_\theta\left(\xi\right)$ for any demonstration ξ w.r.t. a parameterized policy π_θ in the MDP $\mathcal{M}$. Inspired by difficulty scores for supervised learning algorithms [32], we consider a difficulty score which is directly proportional to the loss function ℓ, i.e., $\Psi_\theta\left(\xi\right) \propto g\left(\ell\left(\xi, \theta\right)\right)$, for a monotonically increasing function g.

Definition 1. *The difficulty score of a demonstration ξ w.r.t. policy π_θ in the MDP $\mathcal{M}$ is given by:*

$$\Psi_\theta\left(\xi\right) := \frac{1}{\prod_\tau \pi_\theta(a_\tau^\xi | s_\tau^\xi)}. \tag{2}$$

With the above definition[1], the teacher's and learner's difficulty scores are respectively given by: $\Psi^E\left(\xi\right) := \Psi_{\theta^*}\left(\xi\right)$ and $\Psi_t^L\left(\xi\right) := \Psi_{\theta_t}\left(\xi\right)$. Our strategy, Eq. 1, then leads to the CUR teaching algorithm given in Algorithm 2.

[1]For an MDP $\mathcal{M}$ with deterministic transition dynamics, one can easily note that $\Psi_\theta\left(\xi\right) \propto \exp\left(\ell\left(\xi, \theta\right)\right)$. (see Section 2)

Intuitively, the difficulty score of a demonstration ξ w.r.t. an agent (the teacher or the learner) is inversely proportional to the preference of the agent to follow the demonstration. Demonstrations with a higher likelihood under the agent's policy (higher preference) have a lower difficulty score and vice versa. Notice that the difficulty score can be evaluated with the knowledge of an agent's policy alone, which can be inferred with Monte Carlo methods and does not necessarily require access to the learner's internal parameters (*e.g.* θ), which is often unknown in practical applications. This is an advantage over the prior state-of-the-art teaching approaches for sequential learners [28, 38], which aim to directly steer the learner's internal parameter to the target parameter θ^*.

Teaching with domain knowledge f^E: In this setting, domain knowledge is utilized to accelerate the learning process when the teacher's optimal policy π^E is unknown. Here, the difficulty score Ψ_{θ^*} cannot be computed. We propose to use a task-specific difficulty score w.r.t the teacher, $\Psi^E(\xi) = f^E(\xi)$, designed using domain knowledge (see Section 6). Hence from our curriculum strategy, Eq. 1, we obtain the $\overline{\text{CUR}}$ teaching algorithm, Algorithm 3, which leads to a self-curriculum for the learner.

Algorithm 2 CUR algorithm

Input: Teacher's policy π^E
for $t = 1, 2, \ldots$ **do**
  Observes the learner's current policy π_t^L.
  Selects ξ_t as follows:
$\xi_t \leftarrow \arg\max_\xi \prod_\tau \frac{\pi^E(a_\tau^\xi|s_\tau^\xi)}{\pi_t^L(a_\tau^\xi|s_\tau^\xi)}$.

Algorithm 3 $\overline{\text{CUR}}$ algorithm

Input: Task-specific difficulty f^E
for $t = 1, 2, \ldots$ **do**
  Observes the learner's current policy π_t^L.
  Selects ξ_t as follows:
$\xi_t \leftarrow \arg\max_\xi \frac{f^E(\xi)}{\prod_\tau \pi_t^L(a_\tau^\xi|s_\tau^\xi)}$.

4 Theoretical Analysis of Curriculum Design

Here we present the curriculum analysis for two popular learner models, namely, MaxEnt-IRL and CrossEnt-BC. For our analysis, we consider a teacher with complete knowledge of π^E. We aim to obtain a ranking over the demonstrations to provide to the learner based on the difficulty scores defined in Eq. 2. To this end, we analyze the relationship between the difficulty scores (w.r.t. the learner and the teacher) of the provided demonstration and the teaching objective (convergence towards the target teaching parameter θ^*) during each sequential update step of the learner.

Given difficulty values ψ^E and ψ^L w.r.t. the teacher and the learner respectively, we define the feasible set of demonstrations at time t as $\mathcal{D}_t\left(\psi^E, \psi^L\right) := \left\{\xi : \Psi^E(\xi) = \psi^E \text{ and } \Psi_t^L(\xi) = \psi^L\right\}$. Let $\Delta_t\left(\psi^E, \psi^L\right)$ denote the expected convergence rate of the teaching objective at time t, given difficulty values ψ^E and ψ^L:

$$\Delta_t\left(\psi^E, \psi^L\right) := \mathbb{E}_{\xi_t|\psi^E,\psi^L}[\|\theta^* - \theta_t\|^2 - \|\theta^* - \theta_{t+1}(\xi_t)\|^2], \tag{3}$$

where the expectation is w.r.t. the uniform distribution over the set $\mathcal{D}_t\left(\psi^E, \psi^L\right)$. The following analysis motivates our proposed curriculum strategy.

4.1 Analysis for MaxEnt-IRL Learner

Here, we consider the popular MaxEnt-IRL learner model [10, 37, 39] in an environment MDP $\mathcal{M}$ with deterministic transition dynamics, i.e., $\mathcal{T} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to \{0, 1\}$. In our experiments we relax the simplification and consider stochastic transition dynamics. Under the MaxEnt-IRL learner model, the parametric policy takes the soft-Bellman form, $\pi_\theta(a|s) = \exp\left(Q_\theta(s,a) - V_\theta(s)\right)$ where $V_\theta(s) = \log \sum_a \exp Q_\theta(s,a)$ and $Q_\theta(s,a) = R_\theta(s,a) + \gamma \sum_{s'} \mathcal{T}(s'|s,a) \cdot V_\theta(s')$.

The parametric reward function $R_\theta : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ depends on the parameter $\theta \in \mathbb{R}^d$ and a feature mapping $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^{d'}$ which encodes each state-action pair (s, a) by a feature vector $\phi(s,a) \in \mathbb{R}^{d'}$. For any given θ, the corresponding policy π_θ can be efficiently computed via the Soft-Value-Iteration procedure with reward R_θ (see [37, Algorithm. 9.1]). For our theoretical analysis we consider R_θ with a linear form, i.e., $R_\theta(s,a) := \langle \theta, \phi(s,a) \rangle$, and $d = d'$.

Under the above setting, and a given parameter θ, the probability distribution $\mathbb{P}(\xi|\theta)$ over the demonstrations $\xi = \left\{\left(s_\tau^\xi, a_\tau^\xi\right)\right\}_\tau$ takes the following closed-form: $\mathbb{P}(\xi|\theta) = \frac{\exp\left(\langle \theta, \mu^\xi \rangle\right)}{Z(\theta)}$, where

$\mu^{\xi} := \sum_{\tau=0}^{\infty} \gamma^{\tau} \phi(s_{\tau}^{\xi}, a_{\tau}^{\xi})$, and $Z(\theta)$ is the normalization factor. Then, at time t, the gradient of the MaxEnt-IRL learner is given by $g_t = \mu^{\pi_{\theta_t}} - \mu^{\xi_t}$, where $\mu^{\pi} := \sum_{s,a} \sum_{\tau=0}^{\infty} \gamma^{\tau} \cdot \mathbb{P}\{S_{\tau} = s \mid \pi, \mathcal{M}\} \cdot \pi(a \mid s) \cdot \phi(s, a)$ is the feature expectation vector of policy π.

For the MaxEnt-IRL learner, we obtain the following theorem, which shows the differential effect of the difficulty scores (w.r.t. the teacher and the learner) on the expected rate of convergence of the teaching objective $\Delta_t\left(\psi^E, \psi^L\right)$. The result is analogous to that obtained for linear regression models [32].

Theorem 1. *Assume that η_t is sufficiently small for all t.*[2] *Then, for the MaxEnt-IRL learner, the expected convergence rate of the teaching objective $\Delta_t\left(\psi^E, \psi^L\right)$ is:*

- *monotonically decreasing with ψ^E, i.e., $\frac{\partial \Delta_t}{\partial \psi^E} < 0$, and*
- *monotonically increasing with ψ^L, i.e., $\frac{\partial \Delta_t}{\partial \psi^L} > 0$.*

Theorem 1 motivates our curriculum strategy in Eq. 1. Further, the following theorem states that our strategy can achieve linear convergence to the teaching objective, similar to state-of-the-art teaching approaches for sequential learners [28, 38]. Importantly, unlike the prior approaches, our strategy does not require access to the learner's internal parameters.

Theorem 2. *Consider Algorithm 1 with the MaxEnt-IRL learner and our curriculum strategy, Eq. 1. Then, the teaching objective $\|\theta^* - \theta_t\| \leq \epsilon$ is achieved in $t = \mathcal{O}(\log \frac{1}{\epsilon})$ iterations.*

4.2 Analysis for CrossEnt-BC Learner

Next, we consider the CrossEnt-BC learner model [15, 36]. In this case, the learner's parametric policy takes the following softmax form: $\pi_{\theta}(a|s) = \frac{\exp(H_{\theta}(s,a))}{\sum_{a'} \exp(H_{\theta}(s,a'))}$, where $H_{\theta} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a parametric scoring function that depends on the parameter $\theta \in \mathbb{R}^d$ and a feature mapping $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^{d'}$. For the curriculum analysis in this section, we consider a linear scoring function H_{θ} of the form $H_{\theta}(s, a) := \langle \theta, \phi(s, a) \rangle$ (with $d = d'$). Then, at time step t, the gradient g_t of the CrossEnt-BC learner is given by: $g_t = \sum_{\tau=0}^{\infty} \left(\mathbb{E}_{a \sim \pi_{\theta_t}(\cdot|s_{\tau}^{\xi_t})} \left[\phi(s_{\tau}^{\xi_t}, a)\right] - \phi(s_{\tau}^{\xi_t}, a_{\tau}^{\xi_t}) \right)$. In the experiments, we consider non-linear scoring functions as well (parameterized by neural networks).

Similar to Theorem 1, we obtain the following theorem for the CrossEnt-BC learner, which also justifies our curriculum strategy in Eq. 1.

Theorem 3. *Assume that η_t is sufficiently small for all t.*[2] *Then, for the CrossEnt-BC learner, the expected convergence rate of the teaching objective $\Delta_t\left(\psi^E, \psi^L\right)$ is roughly:*

- *monotonically decreasing with ψ^E, i.e., $\frac{\partial \Delta_t}{\partial \psi^E} < 0$, and*
- *monotonically increasing with ψ^L, i.e., $\frac{\partial \Delta_t}{\partial \psi^L} > 0$.*

5 Experimental Evaluation with Teacher's Policy

Inspired by the works of [28, 40, 41], we evaluate the performance of our curriculum strategy in a synthetic car driving environment on MaxEnt-IRL and CrossEnt-BC learners. In particular, we consider the environment of [28]. In this section, we consider the setting where the teacher knows the optimal policy π^E. Specifically, we evaluate our proposed CUR teaching algorithm, Algorithm 2.

Car driving setup. Figure 1 illustrates a car driving simulator environment consisting of 8 different types of tasks, denoted as `T0`, `T1`, ..., `T7`. Each type is associated with different driving skills. For instance, `T0` corresponds to a basic setup representing a traffic-free highway. `T1` represents a crowded highway. `T2` has stones on the right lane, whereas `T3` has a mix of both cars and stones. Similarly, `T4` has grass on the right lane, and `T5` has a mix of both grass and cars. `T6` and `T7` introduce more complex features such as pedestrians, HOV, and police.

The agent starts navigating from an initial state at the bottom of the left lane of each task, and the goal is to reach the top of a lane while avoiding cars, stones, and other obstacles. The agent can

[2] The technical conditions on η_t are deferred to the appendix.

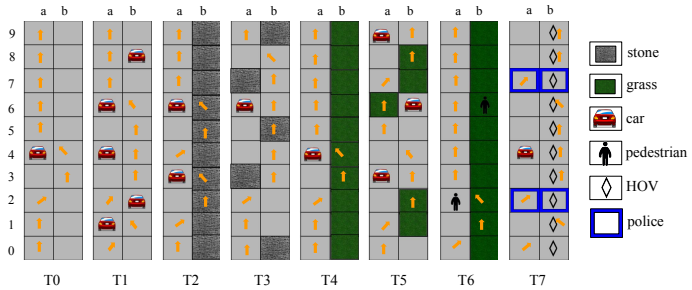
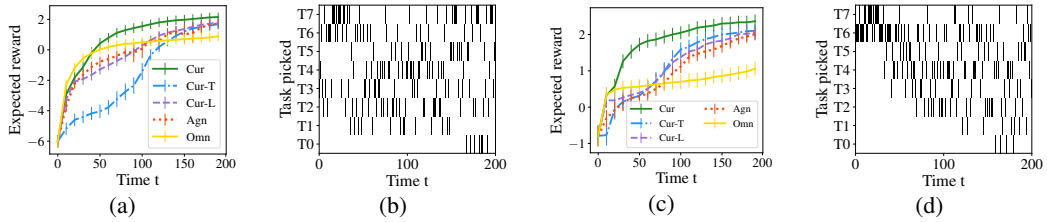


Figure 1: Car environment with 8 different types of tasks. In any given task, an agent starts from the bottom-left corner and the goal is to reach the top. Arrows represent the path taken by the teacher's optimal policy.

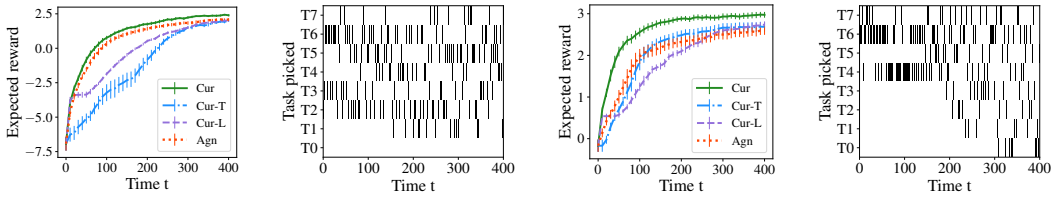
$\phi^E(s)$	$R^{E_{\text{car}}}$
`stone`	-1
`grass`	-0.5
`car`	-5
`ped`	-10
`car-front`	-2
`ped-front`	-5
`HOV`	+1
`police`	0
`HOV` $\wedge$ `police`	-5

Figure 2: $\phi^E(s)$ represents features of a state s. $R^{E_{\text{car}}}$ defines the teacher's reward function.



(a) (b) (c) (d)

Figure 3: Learning curves and curriculum visualization for MaxEnt-IRL learners (with varying initial knowledge) trained on the car driving environment: (a) reward convergence plot and (b) curriculum generated by the CUR teacher for the learner with initial knowledge of `T0`, (c) reward convergence plot and (d) curriculum generated by the CUR teacher for the learner with initial knowledge of `T0`–`T3`.



(a) (b) (c) (d)

Figure 4: Learning curves and curriculum visualization for CrossEnt-BC learners (with varying initial knowledge) trained on the car driving environment: (a) reward convergence plot and (b) curriculum generated by the CUR teacher for the learner with initial knowledge of `T0`, (c) reward convergence plot and (d) curriculum generated by the CUR teacher for the learner with initial knowledge of `T0`–`T3`.

take three different actions given by $\mathcal{A} = \{\texttt{left}, \ \texttt{straight}, \ \texttt{right}\}$. Action `left` steers the agent to the left of the current lane. If the agent is already in the leftmost lane when taking action `left`, then the lane is randomly chosen with uniform probability. We define similar stochastic dynamics for taking action `right`; action `straight` means no change in the lane. Irrespective of the action taken, the agent always moves forward.

Environment MDP Based on the above setup, we define the environment MDP, $\mathcal{M}_{\text{car}}$, consisting of 8 types of tasks, namely `T0`–`T7`, and 5 tasks of each type. Every location in the environment is associated with a state. Each task is of length 10 and width 2, leading to a state space of size $5 \times 8 \times 20$. We consider an action-independent reward function $R^{E_{\text{car}}}$, dependent on an underlying feature vector ϕ^E (see Figure 2).

The feature vector of a state s, denoted by $\phi^E(s)$, is a binary vector encoding the presence or absence of an object at the state. In this work we have two types of features: features indicating the type of the current cell as `stone`, `grass`, `car`, `ped`, `HOV`, and `police`, as well as features providing some look-ahead information such as whether there is a car or pedestrian in the immediate front cell (denoted as `car-front` and `ped-front`).

Given ϕ^E, the MDP's reward function is defined in Figure 2. Essentially there are different penalties (negative rewards) for colliding with specific obstacles such as `stone`, `car` e.t.c. In tasks of type `T7`, $R^{E_{\text{car}}}$ has $+1$ reward value for driving on `HOV`, however, if `police` is present while driving on `HOV`, a negative reward value of -5 is obtained. This results in a nonlinear reward function.

5.1 Teaching Algorithms and Learner Models

Teaching algorithms. The teacher's near-optimal policy, π^E, is obtained via policy iteration [42] with the true reward of the environment, $R^{E_{\text{car}}}$. The teacher interacts with the learner as shown in Algorithm 1, selecting demonstrations using its teaching algorithm. We compare the performance of our proposed CUR teacher (Algorithm 2), against the following baselines:

- AGN: an agnostic teacher that picks demonstrations based on random ordering [28, 32].
- OMN: The omniscient teacher is a state-of-the-art teacher [28, 38], which is applicable only to MaxEnt-IRL learners. With complete knowledge of θ^* and the learner's internal parameter θ_t, the teacher picks demonstrations to directly steer the learner towards θ^* (see Section 3).
- CUR-T: A variant of our CUR teacher (Algorithm 2) that samples demonstrations based on the difficulty score Ψ^E alone, and sets Ψ_t^L to constant.
- CUR-L: A similar variant of our CUR teacher (Algorithm 2) that samples demonstrations based on the difficulty score Ψ_t^L alone, and sets Ψ^E to constant.

As commonly done in prior work [32, 43], we incorporate randomization in the training process for our CUR teacher and its variants using a scheduling mechanism. Demonstrations of higher preference are prioritized at the beginning of training, while during later stages, all demonstrations are provided with uniform probability to the learner. This is further detailed in the appendix.

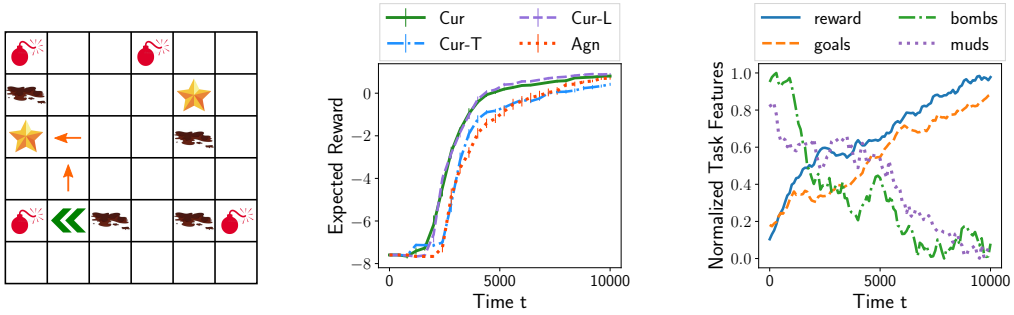
MaxEnt-IRL learner For alignment with the prior state-of-the-art work on teaching sequential MaxEnt-IRL learners [28], we perform teaching over states in our experiments. At time t, the teacher picks a state s_t (where $P_0(s_0^\xi) > 0$) and provides all demonstrations starting from s_t, $\Xi_{s_t} = \left\{\xi = \left\{\left(s_\tau^\xi, a_\tau^\xi\right)\right\}_\tau \text{ s.t. } s_0^\xi = s_t\right\}$, to the learner. The gradient g_t of the MaxEnt-IRL learner is then given by $g_t = \mu^{\pi_{\theta_t}, s_t} - \mu^{\Xi_{s_t}}$, where (i) $\mu^{\Xi_{s_t}} := \frac{1}{|\Xi_{s_t}|}\sum_{\xi\in\Xi_{s_t}} \mu^\xi$, and (ii) μ^{π, s_t} is the feature expectation vector of policy π with starting state set to s_t (see Section 4.1). The learner's feature mapping is given by $\phi(s,a) = \phi^E(s)$ and a non-linear parametric reward function, $R_\theta^L(s,a) = \langle\theta_{1:d'}, \phi(s,a)\rangle + \langle\theta_{d'+1:2d'}, \phi(s,a)\rangle^2$ is chosen, where d' is dimension of $\phi(s,a)$, since a linear representation cannot capture the optimal behaviour for $\mathcal{M}_{\text{car}}$. In our experiments, we consider learners with varying levels of initial knowledge. This means that the initial parameter θ_1 is obtained by training the learner on a subset of tasks before the teaching process starts. In this setting, for our curriculum strategy, Eq. 1, the difficulty score of a set of demonstrations is computed as the mean difficulty score, Eq. 2, of the individual demonstrations in the set.

CrossEnt-BC learner We consider the CrossEnt-BC learner model (see Section 4.2). The learner's feature mapping is given by $\phi(s,a) = \mathbb{E}_{s'\sim\mathcal{T}(\cdot|s,a)}[\phi^E(s')]$, and a quadratic parametric form is selected for the scoring function, i.e. $H_\theta(s,a) = \langle\theta_{1:d'}, \phi(s,a)\rangle + \langle\theta_{d'+1:2d'}, \phi(s,a)\rangle^2$, where d' is the dimension of $\phi(s,a)$. We consider learners with varying initial knowledge and perform teaching over states similar to the MaxEnt-IRL learner.

5.2 Experimental results

Figures 3a, 3c and 4a, 4c show the convergence of the total expected reward for the MaxEnt-IRL and CrossEnt-BC learners respectively, averaged over 10 runs. The CUR teacher is competitive with the OMN teacher, despite not requiring information about the learner's internal parameter. For non-linear parametric reward functions, the MaxEnt-IRL learner no longer solves a convex optimization problem. As a result, forcing the learner to converge to a fixed parameter doesn't necessarily perform well, as seen by the poor performance of the OMN teacher in Figure 3c. The CUR teacher consistently outperforms the AGN teacher, as well as both the CUR-T and CUR-L variants. This clearly shows the effectiveness of our proposed ranking strategy that combines the two difficulty scores.

Figures 3b,3d and 4b,4d visualize the curriculum generated by the CUR teacher for the MaxEnt-IRL and CrossEnt-BC learners respectively. Here, the curriculum refers to the type of task, `T0`–`T7`, associated with the demonstrations provided by the teacher to the learner at time step t. For both types of learners we see that the teacher focuses on tasks the learner is yet to master. For example, in Figures 4d and 3d, at the beginning of training the CUR teacher focuses on tasks of type `T4`–`T7`, which the learner has not been trained on initially.



(a) Grid-World Environment (b) Reward convergence graph (c) Curriculum Visualization

Figure 5: Grid-world environment for the shortest path problem and results of our curriculum teaching strategy: (a) illustration of the grid-world environment (see Section 6), (b) reward convergence curves (see Section 6.2), (c) visualization of the curriculum generated by the $\overline{\text{CUR}}$ teacher. The y-axis represents different features of the tasks provided to the learner, normalized in the range $[0, 1]$.

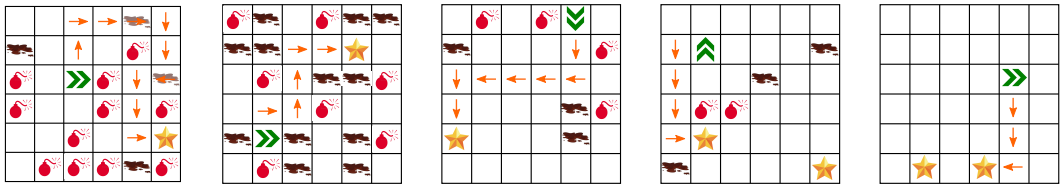


Figure 6: Illustrative examples of the tasks selected by the $\overline{\text{CUR}}$ teacher from the beginning to the later stages of the training process (left to right). See details in section 6.2.

6 Experimental Evaluation without Teacher's Policy

In this section, we evaluate our curriculum strategy in the teaching via domain knowledge setting. The optimal policy π^E is unknown and the teacher designs a task-specific difficulty score f^E (see Section 3). In particular, we study the effectiveness of our $\overline{\text{CUR}}$ algorithm (Algorithm 3) on shortest path problems in a grid-world environment.

Shortest path problem setup and environment MDP A grid-world, Figure 5a, is considered where the agent's objective is to navigate towards a goal cell (depicted by a star) from its initial position (depicted by a green arrow). In the environment, there are cells containing muds and bombs (shown in brown and red respectively) which the agent aims to avoid. There also may be two goal cells and the agent is required to navigate to any one of the goals. Essentially the agent aims to solve the shortest path problem (depicted by orange arrows). We refer to each individual navigation problem as a task.

A state in the environment MDP, $\mathcal{M}_{\text{path}}$, corresponds to a configuration of the grid world including the agent's location and direction. The size of the state space sees a combinatorial growth with the size of the grid. The agent's action space consists of 3 actions, $\mathcal{A} = \{\texttt{move}, \texttt{left}, \texttt{right}\}$. The actions `left` or `right` changes the agent's direction accordingly. The agent moves one step forward in its current direction with the action `move`. The environment reward function $R^{E_{\text{path}}}$ has a -1 reward for each action performed by the agent. Reaching a goal cell has a +10 reward, whereas encountering a cell with mud has a -1 reward, and encountering a cell with a bomb has a -5 reward. Reaching a goal or a bomb ends the agent's episode. Each state s is characterized by a feature vector $\phi^E(s)$ which encodes the agent's location and direction as well as the position of bombs, muds, and goals in the grid-world (see the appendix for details).

6.1 Teaching Algorithms and Learner Model

Teaching algorithms For the shortest path problem, the teacher's demonstration, ξ, for each task corresponds to an optimal path for that task. To pick demonstrations using the $\overline{\text{CUR}}$ teaching algorithm, Algorithm 3, we define a task-specific difficulty score induced by domain knowledge as

follows:

$$f^E(\xi) = \frac{\#goals \times \#optimal_paths}{optimal_reward}, \tag{4}$$

where the optimal rewards for the training tasks are linearly transformed to make all values ≥ 1. Intuitively, f^E is proportional to the difficulty of a task as the greater the number of goals present and optimal paths, the more challenging the task for the learner. Similarly, a higher optimal reward implies a shorter and easier path to a goal.

As baselines, we again consider the teaching algorithms of Section 5.1. As before, the $\overline{\text{CUR}}$ teacher and its variants use a scheduling mechanism to incorporate randomization during training. During the initial epochs, demonstrations of higher preference are provided to the learner. While towards the end of training all demonstrations are provided with uniform probability, similar to prior work [32, 43]. For the AGN teacher, the complete training dataset is provided at each epoch which is the common approach to training. The complete experimental setup is detailed in the appendix.

Learner model We consider a CrossEnt-BC learner (see Section 4.2) with a softmax policy and feature mapping $\phi(s, a) = \phi^E(s)$. The learner's scoring function H_θ is parameterized by a 6 layer Convolutional Neural Network (CNN) which includes 2 residual blocks. The CNN takes as input the feature mapping of a state, i.e., the configuration of the grid-world, and outputs a score for each action. The learner minimizes the cross-entropy loss between its predictions and the teacher's demonstrations during training.

6.2 Results

Figure 5b shows the reward convergence curves on test data, i.e., new tasks not seen during training, for the different teaching algorithms averaged over 5 independent runs. The $\overline{\text{CUR}}$ teacher leads to faster reward convergence than the AGN teacher. The CUR-L teacher is competitive with the $\overline{\text{CUR}}$ teacher in this setting. The curriculum generated by the $\overline{\text{CUR}}$ teacher is depicted in Figure 5c. Figure 6 visualizes the curriculum in terms of specific tasks selected during training. At the beginning of training, mostly tasks with a single goal and a higher number of muds/bombs in the grid world are taught. We hypothesize that this teaches the agent how to avoid muds and bombs while navigating to the goal. Later on during training, tasks with higher optimal reward and a greater number of goals are taught. Here we believe the agent is taught how to identify the path with maximum reward among all paths that lead to a goal. Essentially the learner is first taught the general navigation task followed by the most difficult concept of deciding the optimal path.

7 Discussion and Conclusions

We presented a theoretical analysis of curriculum learning for the sequential MaxEnt-IRL and CrossEnt-BC learner models based on the concept of difficulty scores. Under mild technical conditions, we proved that picking demonstrations with a higher likelihood under the teacher's optimal policy and a lower likelihood under the learner's current policy leads to faster convergence. Further, we proposed a unified curriculum strategy, which does not necessitate knowledge of the learner's internal dynamics, and proved that it achieves linear convergence for the MaxEnt-IRL learner. We validated our theoretical results on a synthetic car driving environment. Additionally, we adapted our proposed curriculum strategy to teach a learner using domain knowledge in the form of task-specific difficulty scores when the teacher's optimal policy is unavailable. The effectiveness of our approach was highlighted on shortest path problems in a grid-world environment.

Our work provides theoretical underpinnings of curriculum design for teaching via demonstrations, which can be beneficial for education applications and faster training of IRL/BC learners across application domains. Some of the interesting directions for future work include: obtaining convergence bounds for CrossEnt-BC and other learner models, curriculum design for reinforcement learning agents, as well as evaluation of our curriculum strategy under approximate inference of the learner's policy, π_t^L, which was not explored in this work.

References

[1] S. Schaal. Learning from demonstration. In *NIPS*, pages 1040–1046, 1997.

[2] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[3] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[4] Maya Cakmak and Andrea L Thomaz. Eliciting good teaching from humans for machine learners. *Artificial Intelligence*, 217:198–215, 2014.

[5] Daphna Buchsbaum, Alison Gopnik, Thomas L Griffiths, and Patrick Shafto. Children's imitation of causal action sequences is influenced by statistical and pedagogical evidence. *Cognition*, 120(3):331–340, 2011.

[6] Patrick Shafto, Noah D Goodman, and Thomas L Griffiths. A rational account of pedagogical reasoning: Teaching by, and learning from, examples. *Cognitive psychology*, 71:55–89, 2014.

[7] D. A. Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

[8] Stuart Russell. Learning agents for uncertain environments. In *COLT*, pages 101–103, 1998.

[9] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *ICML*, page 1, 2004.

[10] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008.

[11] Abdeslam Boularias, Jens Kober, and Jan Peters. Relative entropy inverse reinforcement learning. In *AISTATS*, pages 182–189, 2011.

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

[13] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *ICML*, pages 49–58, 2016.

[14] Dorsa Sadigh, Anca D Dragan, Shankar Sastry, and Sanjit A Seshia. Active preference-based learning of reward functions. *RSS*, 2017.

[15] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 2018.

[16] Thomas J Walsh and Sergiu Goschin. Dynamic teaching in sequential decision making environments. *UAI*, 2012.

[17] Maya Cakmak and Manuel Lopes. Algorithmic and human teaching of sequential decision tasks. In *AAAI*, volume 26, 2012.

[18] Daniel S Brown and Scott Niekum. Machine teaching for inverse reinforcement learning: Algorithms and applications. In *AAAI*, 2019.

[19] Manuel Lopes, Francisco Melo, and Luis Montesano. Active learning for reward estimation in inverse reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 31–46. Springer, 2009.

[20] Kareem Amin, Nan Jiang, and Satinder Singh. Repeated inverse reinforcement learning. In *NeurIPS*, pages 1815–1824, 2017.

[21] Maxwell Svetlik, Matteo Leonetti, Jivko Sinapov, Rishi Shah, Nick Walker, and Peter Stone. Automatic curriculum graph generation for reinforcement learning agents. In *AAAI*, volume 31, 2017.

[22] Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *CORL*, pages 482–495, 2017.

[23] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *ICML*, 2018.

[24] Wojciech Czarnecki, Siddhant Jayakumar, Max Jaderberg, Leonard Hasenclever, Yee Whye Teh, Nicolas Heess, Simon Osindero, and Razvan Pascanu. Mix & match agent curricula for reinforcement learning. In *ICML*, 2018.

[25] Sanmit Narvekar and Peter Stone. Learning curriculum policies for reinforcement learning. In *AAMAS*, pages 25–33, 2019.

[26] Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Robert Fergus. Intrinsic motivation and automatic curricula via asymmetric self-play. In *ICLR*, 2018.

[27] Martin A Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Van de Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing solving sparse reward tasks from scratch. In *ICML*, 2018.

[28] Parameswaran Kamalaruban, Rati Devidze, Volkan Cevher, and Adish Singla. Interactive teaching algorithms for inverse reinforcement learning. In *IJCAI*, 2019.

[29] Jeffrey L Elman. Learning and development in neural networks: The importance of starting small. *Cognition*, 48(1):71–99, 1993.

[30] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, pages 41–48, 2009.

[31] Wojciech Zaremba and Ilya Sutskever. Learning to execute. *arXiv preprint arXiv:1410.4615*, 2014.

[32] Daphna Weinshall, Gad Cohen, and Dan Amir. Curriculum learning by transfer learning: Theory and experiments with deep networks. In *ICML*, 2018.

[33] Daphna Weinshall and Dan Amir. Theory of curriculum learning, with convex loss functions. *arXiv preprint arXiv:1812.03472*, 2018.

[34] Minoru Asada, Shoichi Noda, Sukoya Tawaratsumida, and Koh Hosoda. Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine learning*, 23(2-3):279–303, 1996.

[35] Yuxin Wu and Yuandong Tian. Training agent for first-person shooter game with actor-critic curriculum learning. In *ICLR*, 2016.

[36] Michael Bain. A framework for behavioural cloning. In *Machine Intelligence 15*, pages 103–129, 1995.

[37] Brian D Ziebart. *Modeling Purposeful Adaptive Behavior with the Principle of Maximum Causal Entropy*. PhD thesis, University of Washington, 2010.

[38] Weiyang Liu, Bo Dai, Ahmad Humayun, Charlene Tay, Chen Yu, Linda B Smith, James M Rehg, and Le Song. Iterative machine teaching. In *ICML*, 2017.

[39] Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. The principle of maximum causal entropy for estimating interacting processes. *IEEE Transactions on Information Theory*, 59(4):1966–1980, 2013.

[40] Andrew Y Ng and Stuart J. Russell. Algorithms for inverse reinforcement learning. In *ICML*, 2000.

[41] Sergey Levine, Zoran Popovic, and Vladlen Koltun. Feature construction for inverse reinforcement learning. *NIPS*, 23:1342–1350, 2010.

[42] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[43] Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? In *ICLR*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] In Sections 3 and 4 we present our curriculum strategy and provide theoretical analysis as described in the introduction. The experimental results mentioned in the abstract and introduction are detailed in Sections 5 and 6.
   (b) Did you describe the limitations of your work? [Yes] We discuss the limitations of our work in Section 7. We point out that we do not explore the performance of our approach under approximate inference of the learner's policy, π_t^L, and that convergence bounds for CrossEnt-BC learners have not been obtained yet. Both of these would be interesting directions for future work.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] As stated in Section 7, this work presents the theoretical underpinnings of curriculum design for teaching via demonstrations. This can be beneficial for education applications and faster training of IRL/BC learners. As such in the present form, we do not see any direct negative societal impacts of our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] We confirm that our paper conforms with the ethics review guidelines.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] In Section 4.1 we state our assumptions of a deterministic MDP and linear reward structure. Similarly we state our assumption of a linear scoring function for the CrossEnt-BC learner in Section 4.2. The simplifications considered are reiterated in the proofs.
   (b) Did you include complete proofs of all theoretical results? [Yes] Complete proofs of all theoretical results are included in the appendix.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code and instructions are included in the supplemental material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Training details are presented in the appendix and are also present in the code.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Error bars are included in all the result graphs, as can be seen in Figures 3, 4 and 5.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] The details are provided in the appendix.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]