Cost Constrained Imitation Learning

Qian Shao
Singapore Management University
Singapore, Singapore
qianshao.2020@phdcs.smu.edu.sg

Pradeep Varakantham
Singapore Management University
Singapore, Singapore
pradeepv@smu.edu.sg

Shih-Fen Cheng
Singapore Management University
Singapore, Singapore
sfcheng@smu.edu.sg

ABSTRACT

Imitation learning allows artificially intelligent systems to learn either the reward (or preference) model (or directly the behavioral policy) only from observing the behavior of an expert. Existing work in imitation learning and inverse reinforcement learning has focused on imitation primarily in unconstrained settings (e.g., no limit on fuel consumed by the vehicle). In many real-world domains, the behavior of an expert is governed not only by reward (or preference) but also by constraints. For instance, decisions of self-driving delivery vehicles are dependent not only on the route preferences/rewards (depending on past demand data) but also on the fuel in the vehicle. In such problems, imitation learning is challenging as decisions are not only dictated by the reward model but are also dependent on a cost constraint model. In this paper, we provide a reward-generative model to address imitation learning in cost-constrained environments. We demonstrate that the objective for imitation learning in cost-constrained environments can be succinctly derived. Finally, we empirically show that our approach is able to handle cost constraints exceedingly well and provides clear benefits over leading approaches on multiple benchmark problems from the literature.

KEYWORDS

imitation learning, constrained markov decision process, inverse reinforcement learning

1 INTRODUCTION

Imitation learning aims to replicate expert behaviors by directly observing human demonstrations, eliminating the need for designing explicit reward signals as in reinforcement learning (RL) [1]. This approach has been successfully applied in a variety of domains such as robotics [5], autonomous vehicles [10], and game AI [8]. Typically, this is achieved through techniques such as behavioral cloning [3], inverse reinforcement learning [13], and generative inverse reinforcement learning [7].

The previous research in the fields of imitation learning and inverse reinforcement learning has primarily concentrated on mimicking human behaviors in unconstrained environments, such as mimicking driving a vehicle without any limitations on fuel consumption by the vehicle. However, in many practical scenarios, experts consider not only rewards or preferences, but also limitations or constraints. For example, the decisions made by a self-driving delivery vehicle are not only based on route preferences or rewards, which are derived from past demand data, but also on the amount of fuel/power available in the vehicle. As another example, when an agent is being trained to drive a car on a race track, the expert demonstrations that the agent is mimicking involve high-speed driving and precision maneuvering, which are critical for success in a race. However, it is also essential for the agent to adhere to safety constraints, such as staying within the boundaries of the track and avoiding collisions with other vehicles. These safety constraints are different from the reward function, which may focus on achieving a fast lap time or winning the race. Therefore, the agent must strike a balance between the goal of imitating the expert demonstrations and the need to adhere to the safety constraints in order to successfully complete the task.

In scenarios where the decision-making process is influenced by both a reward model and a cost constraint model, the implementation of imitation learning becomes significantly more complex. This is because the decisions made are not solely based on the reward model, but also take into consideration the limitations imposed by the cost constraint model. To that end, we provide a new imitation learning problem in cost-constrained environments. The closest research to work provided in this paper is by Malik et al. [11], where cost constraints have to be learned from expert trajectories when the reward function is already provided. Our work is fundamentally different from their work, as we consider settings where the reward model is unknown (as it is usually a subjective preference), but the cost consumption and constraints are known (e.g., fuel consumed).

Contributions: Our key contributions are as follows:

- First, we formulate the cost-constrained imitation learning problem and provide a reward-generative model to address the challenge of imitation learning in cost-constrained environments. By utilizing this model, we are able to effectively address the issue of imitating expert behavior while ensuring that cost constraints are met during the exploration stage.
- We theoretically derive how the objective for imitation in cost-constrained environments can be succinctly derived using our proposed method. This proof serves as a solid foundation for the effectiveness of our model in addressing this specific problem.
- To further validate the effectiveness of our proposed method, we have conducted extensive evaluations on modified cost-constrained benchmarks from MuJoCo. The results of these evaluations show that our method is able to effectively imitate expert behavior while satisfying cost constraints.

2 BACKGROUND AND RELATED WORK

In this section, we describe the two important relevant models in this paper, namely Constrained MDPs and Imitation Learning. We also briefly review related work.

2.1 Constrained Markov Decision Process

Reinforcement Learning problems are characterized by an underlying Markov Decision Process (MDP), which is defined by the

Proc. of the Adaptive and Learning Agents Workshop (ALA 2023), Cruz, Hayes, Wang, Yates (eds.), May 29-30, 2023, London, UK, https://alaworkshop2023.github.io/. 2023.

tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$. Where $\mathcal{S}$ represents the set of states, $\mathcal{A}$ represents the set of actions. The reward function, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, provides a quantitative measure of how well the system is performing based on the current state and action. The transition function, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$, defines the probability of transitioning from one state to another, given the current state and action taken. Specifically, the probability of transitioning from state s to s', given that action a is taken, is represented by $\mathcal{P}(s'|s, a)$. A feasible set of policies, denoted as Π, contains all possible policies that can be implemented in the system. The objective of the MDP problem is to find an optimal policy, $\pi \in \Pi$, by maximizing the reward-based objective function, which is defined as follows:

$$\max_{\pi \in \Pi} \mathbb{E}_\pi [r(s, a)]. \tag{1}$$

In this work, we examine the scenario in which agents aim to optimize their rewards while adhering to policy-based cost constraints. This leads to an extension of the traditional MDP framework referred to as the Constrained Markov Decision Process (CMDP) [2]. The objective in a CMDP problem is succinctly formulated as:

$$\begin{aligned} &\max_{\pi \in \Pi} \mathbb{E}_\pi [r(s, a)] \\ &s.t. \quad \mathbb{E}_\pi [d(s, a)] \leq d_0. \end{aligned} \tag{2}$$

Where $d(s, a)$ is the cost associated with taking action a in state s and is independent of the reward function, $r(s, a)$. d_0 is the expected cost threshold for any selected policy.

2.2 Imitation Learning

Methods of Reinforcement Learning require clearly defined and observable reward signals, which provide the agent with feedback on their performance. However, in many real-world scenarios, defining these rewards can be very challenging. Imitation learning, on the other hand, offers a more realistic approach by allowing agents to learn behavior in an environment through observing expert demonstrations, without the need for getting access to a defined reward signal.

An effective method for addressing imitation learning challenges is Behavior Cloning (BC) [3]. This approach utilizes the states and actions of a demonstrator as training data, allowing the agent to replicate the expert's policy [14]. One of the advantages of this method is that it does not require the agent to actively interact with the environment, instead, it operates as a form of supervised learning, similar to classification or regression. Despite its simplicity, BC is known to suffer from a significant drawback, namely the compounding error caused by covariate shift [16]. This occurs when minor errors accumulate over time, ultimately resulting in a significantly different state distribution.

Another approach, Inverse Reinforcement Learning (IRL) [13] aims to identify the underlying reward function that explains the observed behavior of an expert. Once the reward function is determined, a standard Reinforcement Learning algorithm can be used to obtain the optimal policy. The reward function is typically defined as a linear [1, 13] or convex [19] combination of the state features, and the learned policy is assumed to have the maximum entropy [23] or maximum causal entropy [22]. However, many IRL methods are computationally expensive and may produce multiple possible formulations for the true reward function. To address these challenges, Generative Adversarial Imitation Learning (GAIL) [7] was proposed. GAIL directly learns a policy by using a discriminator to distinguish between expert and learned actions, with the output of the discriminator serving as the reward signal. With its state-of-the-art performance in various applications, we designate GAIL as the baseline for our algorithms.

We now describe the imitation learning problem and GAIL approach here as it serves as the basis for our method. The learner's goal is to find a policy, denoted as π, that performs at least as well as an expert policy, denoted as π_E, with respect to an unknown reward function, denoted as $r(s, a)$. For a given policy $\pi \in \Pi$, we define its occupancy measure, denoted as $\rho_\pi \in \Gamma$, as [15]

$$\rho_\pi(s, a) = \pi(a|s) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)$$

The occupancy measure represents the distribution of state-action pairs that an agent encounters when navigating the environment with the specified policy π. It is important to note that there is a one-to-one correspondence between the set of policies, Π, and the set of occupancy measures, Γ. Therefore, an imitation learning problem can be equivalently formulated as a matching learning problem between the occupancy measure of the learner's policy, ρ_π, and the occupancy measure of the expert's policy, ρ_{π_E}. In general, the objective can be succinctly represented as the task of finding a policy that closely matches the occupancy measure of the expert's policy, which is represented as:

$$\min_\pi -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E}), \tag{3}$$

where $H(\pi) \triangleq \mathbb{E}_\pi[-\log \pi(a|s)]$ is the causal entropy of the policy π, which is defined as the expected value of the negative logarithm of the probability of choosing an action a given a state s, under the distribution of the policy π. Additionally, the distance measure between the state-action distribution of the policy π, represented by ρ_π, and the expert's state-action distribution, represented by ρ_{π_E}, is represented by the symbol ψ^*. Specifically, the distance measure (Jensen-Shannon divergence) employed by the GAIL framework is defined as follows:

$$\psi^*(\rho_\pi - \rho_{\pi_E}) = \max_D \mathbb{E}_\pi[\log D(s, a)] + \mathbb{E}_{\pi_E}[\log(1 - D(s, a))] \tag{4}$$

The GAIL method utilizes a combination of imitation learning and generative adversarial networks, where $D \in (0, 1)^{\mathcal{S} \times \mathcal{A}}$ acts as the discriminator. Through this formalism, the method trains a generator, represented by π_θ, to generate state-action pairs that the discriminator attempts to distinguish from expert demonstrations. The optimal policy is achieved when the discriminator is unable to distinguish between the data generated by the generator and the expert data.

In our problem, as we aim to address the imitation learning problem within the constraints of an MDP, we have employed a unique distance measure that diverges from the traditional GAIL framework. This approach allows us to more effectively navigate the complexities of the constrained MDP setting and achieve our desired outcome.

3 COST CONSTRAINED IMITATION LEARNING

In this section, we first describe the problem of cost-constrained imitation learning and outline our approach to compute the policy that mimics expert behavior while satisfying the cost constraints. Formally, the problem is a combination of the CMDP problem (2) and the Imitation Learning problem (3) and can be succinctly characterized as:

$$\begin{aligned} &\min_{\pi} -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E}) \\ &s.t. \quad \mathbb{E}_\pi[d(s,a)] \leq \mathbb{E}_{\pi_E}[d(s,a)] \end{aligned} \tag{5}$$

Our approach to computing the policy that mimics the behavior of the expert within cost constraints is focused on computing a solution to the objective function of Equation (6) below. The theoretical justification for choosing this objective function is provided in the next section. Intuitively, the objective is composed of three optimizations:

- Minimize the distance between state, action distributions of new policy, π_θ and expert policy, π_E. This is transformed into the loss associated with a discriminator, D_ω, which discriminates between expert state, action pairs and the state, action pairs generated by the new policy, π_θ.
- Maximize the entropy of the new policy, π_θ to ensure none of the correct policies are ignored.
- Minimize the cost constraint violations corresponding to the new policy, π_θ.

$$\begin{aligned} L(\omega,\lambda,\theta) \triangleq \min_{\theta}\max_{\omega,\lambda} \quad & \mathbb{E}_{\pi_\theta}[\log D_\omega(s,a)] + \mathbb{E}_{\pi_E}\left[\log(1 - D_\omega(s,a))\right] \\ & + \lambda\left(\mathbb{E}_{\pi_\theta}[d(s,a)] - \mathbb{E}_{\pi_E}[d(s,a)]\right) - \lambda_1 H(\pi_\theta) \end{aligned} \tag{6}$$

where θ represents the parameters of the policy, λ_1 is the parameter corresponding to the causal entropy (since we maximize entropy similar to imitation learning) and finally, λ is the parameter corresponding to cost constraint. $H(\pi_\theta) \triangleq \mathbb{E}_{\pi_\theta}[-\log \pi_\theta(a|s)]$ is the casual entropy of policy π_θ.The given expert policy is represented by π_E, and a known cost function, represented by d, is also incorporated into the objective function.

Given the three optimization criterion, we do not choose one of the three but instead compute a saddle point $(\theta, \omega, \lambda)$ for (6). To accomplish this, we will employ a parameterized policy, represented by π_θ, with adjustable weights θ, as well as a discriminator network, represented by D_ω, which maps states and actions to a value between 0 and 1, and has its own set of adjustable weights ω. The Lagrangian multiplier, denoted by λ, is for penalizing the number of cost constraint violations.

To obtain the saddle point, we update the parameters of the policy, discriminator, and Lagrangian multiplier sequentially:

Updating ω: The gradient of (6) with respect to ω is calculated as:

$$\begin{aligned} &\nabla_\omega L(\omega,\lambda,\theta) = \\ &\quad \mathbb{E}_{\pi_\theta}[\nabla_\omega \log D_\omega(s,a)] + \mathbb{E}_{\pi_E}[\nabla_\omega \log(1 - D_\omega(s,a))] \end{aligned} \tag{7}$$

We utilize the Adam gradient step method [9] on the variable ω, targeting the maximization of (6) in relation to D.

Updating θ: To update policy parameters, we adopt Trust Region Policy Optimization (TRPO) method [17]. The theoretical foundation of the TRPO update process involves utilizing a specific algorithm to improve the overall performance of the policy by optimizing the parameters within a defined trust region:

$$\begin{aligned} \theta_{k+1} &= \arg\max_{\theta} \mathcal{L}(\theta_k, \theta) \\ s.t. \quad & \bar{D}_{KL}(\theta||\theta_k) \leq \delta \end{aligned} \tag{8}$$

The key challenge in applying the TRPO update process is the computation of the surrogate advantage, denoted by $\mathcal{L}(\theta_k, \theta)$. It is a metric that quantifies the relative performance of a new policy π_θ in comparison to an existing policy π_{θ_k}, based on data collected from the previous policy:

$$\mathcal{L}(\theta_k,\theta) = \mathop{\mathbb{E}}_{s,a\sim\pi_{\theta_k}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta_k}(a|s)}\left(A^{\pi_{\theta_k}}(s,a) - \lambda A_d^{\pi_{\theta_k}}(s,a)\right)\right]. \tag{9}$$

We do not have a reward function to compute the advantage and hence we utilize the output of the discriminator, represented by $\log D_\omega(s,a)$, as the reward signal. Subsequently, we employ the Generalized Advantage Estimation (GAE) method outlined in [18] to calculate the advantage of the reward, $A^{\pi_{\theta_k}}(s,a)$. Additionally, we also calculate the advantage pertaining to cost, denoted as $A_d^{\pi_{\theta_k}}(s,a)$, by utilizing the GAE method, as we have knowledge of the cost function.

The average KL-divergence, represented by $\bar{D}_{KL}(\theta||\theta_k)$, between policies across states visited by the previous policy can be computed as:

$$\bar{D}_{KL}(\theta||\theta_k) = \mathop{\mathbb{E}}_{s\sim\pi_{\theta_k}} D_{KL}\left(\pi_\theta(\cdot|s)||\pi_{\theta_k}(\cdot|s)\right) \tag{10}$$

Updating λ: We do an Adam gradient step on λ to increase (6), the gradient of (6) with respect to λ is calculated as:

$$\nabla_\lambda L(\omega,\lambda,\theta) = \left(\mathbb{E}_{\pi_\theta}[d(s,a)] - \mathbb{E}_{\pi_E}[d(s,a)]\right) \tag{11}$$

Algorithm 1 shows the pseudocode for our approach, CCIL (Cost Constrained Imitation Learning).

Algorithm 1 Cost Constrained Imitation Learning, CCIL

Input: expert trajectories $\tau_E \sim \pi_E$, initial parameters of policy, discriminator and Lagrangian multiplier θ_0,ω_0,λ_0, maximum cost d_0

Output: Learned policy π_θ

1: **for** $i = 0, 1, 2, \ldots$ **do**
2: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
3: Update ω_i to ω_{i+1} by ascending with gradients: $\triangle_{\omega_i} = \hat{\mathbb{E}}_{\tau_i}[\nabla_{\omega_i}\log(D_{\omega_i}(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_{\omega_i}\log(1 - D_{\omega_i}(s,a))]$
4: Take a policy step from θ_i to θ_{i+1}, using the TRPO update rule with the following objective: $\hat{\mathbb{E}}_{\tau_i}[\log(D_{\omega_{i+1}}(s,a))] + \lambda_i\hat{\mathbb{E}}_{\pi_{\theta_i}}[d(s,a)] - \lambda_1 H(\pi_{\theta_i})$
5: Update λ_i to λ_{i+1} by ascending with gradients: $\triangle_{\lambda_i} = \hat{\mathbb{E}}_{\tau_i}[d(s,a)] - \hat{\mathbb{E}}_{\tau_E}[d(s,a)]$
6: **end for**

4 THEORETICAL ANALYSIS

The objective function of the imitation learning problem can be represented using (3), and the distance measure in the GAIL framework is defined as (4).

Our proof is based on the GAIL framework, and the objective function of the cost-constrained imitation learning problem is formulated in (6). However, it is important to note that the form of the distance measure will differ from that of (4), as will be explained in the following theory.

Theorem 1. *The objective function of the cost-constrained imitation learning problem is:*

$$\min_{\pi\in\Pi} -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E}), \tag{12}$$

where $\psi^*(\rho_\pi - \rho_{\pi_E}) = \max_{D,\lambda} \mathbb{E}_\pi[\log(D(s,a))] + \mathbb{E}_{\pi_E}[\log(1-D(s,a))] + \lambda(\mathbb{E}_\pi[d(s,a)] - \mathbb{E}_{\pi_E}[d(s,a)])$

We will provide a proof sketch for the above theorem in two steps :

Step 1: Typically, optimal policy in an imitation learning setting is obtained by first solving the Inverse Reinforcement Learning (IRL) problem to get the optimal reward function r^* and then running an RL algorithm on the obtained reward function. In GAIL, these two steps were compressed into optimizing a ψ−regularized objective. **Our first step is to show this can be also done for Cost Constrained Imitation Learning problems.**

Step 2: Our second step is to derive the specific form of ψ^* for CCIL problems.

4.1 Step 1

Constrained Markov Decision Process (CMDP) is commonly solved by utilizing the Lagrangian relaxation technique [20]. Then CMDP is transformed into an equivalent unconstrained problem by incorporating the cost constraint into the objective function:

$$\max_{\lambda\geq 0} \min_{\pi\in\Pi} \mathbb{E}_\pi[-r(s,a)] + \lambda(\mathbb{E}_\pi[d(s,a)] - d_0) \tag{13}$$

In the aforementioned equation, our objective is to find the saddle point of the minimax problem. Since the reward function $r(s,a)$ is not provided, our goal is to determine the optimal policy by utilizing the expert policy π_E and the given cost functions $d(s,a)$. To accomplish this, we utilize the maximum casual entropy Inverse Reinforcement Learning (IRL) method [22][23] to solve the following optimization problem:

$$\begin{aligned}\max_{\substack{r\in\mathcal{R}\\ \lambda\geq 0}} &\left(\min_{\pi\in\Pi} -H(\pi) + \mathbb{E}_\pi[-r(s,a)] + \lambda(\mathbb{E}_\pi[d(s,a)] - d_0)\right) \\ &- \left(\mathbb{E}_{\pi_E}[-r(s,a)] + \lambda(\mathbb{E}_{\pi_E}[d(s,a)] - d_0)\right)\end{aligned} \tag{14}$$

Where $\mathcal{R}$ is a set of reward functions. Maximum casual entropy IRL aims to find a reward function $r \in \mathcal{R}$ that gives low rewards to the learner's policy while giving high rewards to the expert policy. The optimal policy can be found via a reinforcement learning procedure:

$$RL(r,\lambda) = \arg\min_{\pi\in\Pi} -H(\pi) + \mathbb{E}_\pi[\lambda d(s,a) - r(s,a)] - \lambda d_0 \tag{15}$$

We study policies generated through reinforcement learning, utilizing rewards learned through IRL on the most extensive set of reward functions, denoted as $\mathcal{R}$ in Eq.(14), which encompasses all functions mapping from $\mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ to $\mathbb{R}$. However, as the use of a large $\mathcal{R}$ can lead to overfitting in the IRL process, we employ a concave reward function regularizer [6], denoted as ψ, to define the IRL procedure:

$$\begin{aligned}IRL_\psi(\pi_E, d) = \quad &\arg\max_{\substack{r\in\mathbb{R}^{\mathcal{S}\times\mathcal{A}}\\ \lambda\geq 0}} \left(\min_{\pi\in\Pi} -H(\pi) + \mathbb{E}_\pi[\lambda d(s,a) - r(s,a)]\right) \\ &- \mathbb{E}_{\pi_E}[\lambda d(s,a) - r(s,a)] + \psi(r)\end{aligned} \tag{16}$$

Given $(\tilde{r},\tilde{\lambda}) \in IRL_\psi(\pi_E, d)$, our objective is to learn a policy defined by $RL(\tilde{r},\tilde{\lambda})$. To characterize $RL(\tilde{r},\tilde{\lambda})$, it is commonly beneficial to convert optimization problems involving policies into convex problems. We use occupancy measure ρ_π to accomplish this. After which we express the expected value of the reward and the expected value of the constraint as: $\mathbb{E}_\pi[r(s,a)] = \sum_{s,a} \rho_\pi(s,a) r(s,a)$ and $\mathbb{E}_\pi[d(s,a)] = \sum_{s,a} \rho_\pi(s,a) d(s,a)$ as described in Altman [2]. IRL can be reformulated as:

$$\begin{aligned}IRL_\psi(\pi_E, d) = &\arg\max_{\substack{r\in\mathbb{R}^{\mathcal{S}\times\mathcal{A}}\\ \lambda\geq 0}} \min_{\pi\in\Pi} -H(\pi) + \psi(r) + \\ &\sum_{s,a} (\rho_\pi(s,a) - \rho_{\pi_E}(s,a))[\lambda d(s,a) - r(s,a)]\end{aligned} \tag{17}$$

We then characterize $RL(\tilde{r},\tilde{\lambda})$, the policy learned by RL on the reward recovered by IRL as the optimal solution of Eq.(12).

Proposition 1. *(Theorem 2 of [19]) If $\rho \in \mathcal{D}$, then ρ is the occupancy measure for $\pi_\rho(a|s) \triangleq \rho(s,a)/\sum_{a'} \rho(s,a')$, and π_ρ is the only policy whose occupancy measure is ρ.*

Proposition 2. *(Lemma 3.1 of [7]) Let $\bar{H}(\rho) = -\sum_{s,a} \rho(s,a) \log(\rho(s,a)/\sum_{a'} \rho(s,a'))$. Then, $\bar{H}$ is strictly concave, and for all $\pi \in \Pi$ and $\rho \in \mathcal{D}$, we have $H(\pi) = \bar{H}(\rho_\pi)$ and $\bar{H}(\rho) = H(\pi_\rho)$.*

Proposition 3. *Let $(\tilde{r},\tilde{\lambda}) \in IRL_\psi(\pi_E, d)$, $\tilde{\pi} \in RL(\tilde{r},\tilde{\lambda})$, and*

$$\begin{aligned}\pi_A &\in \arg\min_\pi -H(\pi) + \psi^*(\rho_\pi - \rho_{\pi_E}) \\ &= \arg\min_\pi \max_{r,\lambda} -H(\pi) + \psi(r) + \\ &\sum_{s,a} (\rho_\pi(s,a) - \rho_{\pi_E}(s,a))[\lambda d(s,a) - r(s,a)]\end{aligned} \tag{18}$$

Then $\pi_A = \tilde{\pi}$.

Proof. Let ρ_A be the occupancy measure of π_A and $\tilde{\rho}$ be the occupancy measure of $\tilde{\pi}$. By using Proposition 1, we define $\bar{L} : \mathcal{D} \times \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \times \mathbb{R} \to \mathbb{R}$ by

$$\begin{aligned}\bar{L}(\rho, (r,\lambda)) = &-\bar{H}(\rho) + \psi(r) + \\ &\sum_{s,a} (\rho_\pi(s,a) - \rho_{\pi_E}(s,a))[\lambda d(s,a) - r(s,a)]\end{aligned} \tag{19}$$

The following relationship then holds:

$$\rho_A \in \arg\min_{\rho\in\mathcal{D}} \max_{r,\lambda} \bar{L}(\rho, (r,\lambda)) \tag{20}$$

$$(\tilde{r},\tilde{\lambda}) \in \arg\max_{r,\lambda} \min_{\rho\in\mathcal{D}} \bar{L}(\rho, (r,\lambda)) \tag{21}$$

$$\tilde{\rho} \in \arg\min_{\rho\in\mathcal{D}} \bar{L}(\rho, (\tilde{r},\tilde{\lambda})) \tag{22}$$

$\mathcal{D}$ is compact and convex, $\mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ is convex. Due to convexity of $-\bar{H}$,it follows that $\bar{L}(\rho,\cdot)$ is convex for all ρ. $\bar{L}(\cdot,(r,\lambda))$ is concave

for all (r, λ) (see proof in appendix A.1), Therefore, we can use minimax duality [12]:

$$\min_{\rho \in \mathcal{D}} \max_{\substack{r \in \mathcal{R} \\ \lambda}} \bar{L}(\rho, (c, \lambda)) = \max_{\substack{r \in \mathcal{R} \\ \lambda}} \min_{\rho \in \mathcal{D}} \bar{L}(\rho, (c, \lambda)) \tag{23}$$

Hence,from Eqs.(20) and (21), $(\rho_A, (\tilde{r}, \tilde{\lambda}))$ is a saddle point of $\bar{L}$, which implies that:

$$\rho_A \in \arg\min_{\rho \in \mathcal{D}} \bar{L}(\rho, (\tilde{r}, \tilde{\lambda})) \tag{24}$$

Because $\tilde{L}(\cdot, (r, \lambda))$ is strictly concave for all (r, λ), Eqs.(22) and (24) imply $\rho_A = \tilde{\rho}$. Since policies whose corresponding occupancy measure are unique(Proposition 2), finally we get $\pi_A = \tilde{\pi}$ □

Proposition 3 illustrates the process of IRL in finding the optimal reward function and Lagrangian multiplier, represented by (r^*, λ^*). By utilizing the output of IRL, reinforcement learning can be executed to obtain the optimal policy, represented by π^*. And we prove that π^* is the same as by directly solving the ψ-regularized imitation learning problem $\tilde{L}$. Furthermore, ψ-regularized imitation learning aims to identify a policy whose occupancy measure is similar to that of an expert, as measured by the convex function ψ^*. Subsequently, we deduce the form of ψ^*.

4.2 Step 2

In the GAIL paper [7], the authors present a cost regularizer, ψ_{GA}, that leads to an imitation learning algorithm, as outlined in Eq.(3), which aims to minimize the Jensen-Shannon divergence between the occupancy measures. Specifically, they convert a surrogate loss function, ϕ, which is used for binary classification of state-action pairs drawn from the occupancy measures ρ_π and ρ_{π_E}, into cost function regularizers ϕ, such that $\phi^*(\rho_\pi - \rho_{\pi_E})$ represents the minimum expected risk, $R_\phi(\rho_\pi, \rho_{\pi_E})$, for the function ϕ [7].

$$R_\phi(\rho_\pi, \rho_{\pi_E}) = \sum_{s,a} \max_{\gamma \in \mathbb{R}} \rho_\pi(s, a)\phi(\gamma) + \rho_{\pi_E}(s, a)\phi(-\gamma) \tag{25}$$

Here we use the same formula of surrogate loss function ϕ as in GAIL paper: $\psi_\phi(c) = \sum_{\rho_{\pi_E}} g_\phi(c(s, a))$, where $g_\phi(x) = -x + \phi(-\phi^{-1}(-x))$, ϕ is a strictly decreasing convex function (Proposition A.1 from Ho and Ermon [7]). Noted that in GAIL paper they adopt cost function $c(s, a)$ not reward function $r(s, a)$, then we write in this form: $\psi_\phi(-r) = \sum_{\rho_{\pi_E}} g_\phi(-r(s, a))$.

Then formulation of $\psi^*_\phi(\rho_\pi - \rho_{\pi_E})$ is represented as follows(see proof in Appendix A.2):

$$\begin{aligned} &\psi^*_\phi(\rho_\pi - \rho_{\pi_E}) \\ &= -R_\phi(\rho_\pi, \rho_{\pi_E}) + \max_\lambda \sum_{s,a} \lambda(\rho_\pi(s, a) - \rho_{\pi_E}(s, a))d(s, a) \end{aligned} \tag{26}$$

Using the logistic loss $\phi(\gamma) = \log(1+e^{-\gamma})$, the same form in GAIL paper, then $-R_\phi(\rho_\pi, \rho_{\pi_E}) = \max_{D \in (0,1)^{\mathcal{S} \times \mathcal{A}}} \sum_{s,a} \rho_\pi(s, a) \log D(s, a) + \rho_{\pi_E}(s, a) \log(1 - D(s, a))$. Therefore, we obtain the final form of $\psi^*(\rho_\pi - \rho_{\pi_E})$ as follows:

$$\begin{aligned} &\psi^*(\rho_\pi - \rho_{\pi_E}) = \max_{\substack{D \in (0,1)^{\mathcal{S} \times \mathcal{A}} \\ \lambda}} \sum_{s,a} \rho_\pi(s, a) \log D(s, a) + \\ &\rho_{\pi_E}(s, a) \log(1 - D(s, a)) + \lambda(\rho_\pi(s, a) - \rho_{\pi_E}(s, a))d(s, a) \\ &= \max_{\substack{D \in (0,1)^{\mathcal{S} \times \mathcal{A}} \\ \lambda}} \mathbb{E}_\pi[\log D(s, a)] + \mathbb{E}_{\pi_E}[\log(1 - D(s, a))] \\ &+ \lambda(\mathbb{E}_\pi[d(s, a)] - \mathbb{E}_{\pi_E}[d(s, a)]) \end{aligned} \tag{27}$$

Therefore, we prove Theorem 1 and the objective function of cost-constrained imitation learning is Eq.(6).

5 EXPERIMENTS

In this section, we compare the performance of our approach, CCIL in comparison to leading approaches for imitation learning, GAIL [7] and Behavioral Cloning (BC) [3]. This is to illustrate that a new approach is needed to mimic expert behavior when there are cost constraints in play. As we show, GAIL and BC can extensively violate the cost constraints.

5.1 Setup

Environments. We chose MuJoCo [21] due to its comprehensive collection of continuous control tasks, such as Ant, Walker2d, Swimmer, and Hopper, commonly used to evaluate Reinforcement Learning (RL) and Imitation Learning (IL) algorithms. Since existing environments do not have any cost constraints, we artificially introduced constraints on certain features of the state space:

- We imposed the speed limit as a cost indicator for the Swimmer, Walker2d, and Ant environments. If the speed/velocity of a performed action exceeds 1, the cost is denoted as 1; otherwise, we denote the cost as 0. Additionally, we utilized the default MuJoco environment settings as the reward function for these three environments.
- For the Hopper environment, we use the control cost as the cost indicator, penalizing the hopper for actions that are too large. If the control cost of a performed action exceeds 0.001, the cost is 1. The Hopper reward function consists of two parts: healthy reward and forward reward. Every time step that the hopper remained healthy, it received a fixed value 'healthy reward', and a 'foward reward' is also given for hopping forward.

The expert trajectories were generated by solving a forward-constrained RL problem, and the statistics of these trajectories are summarized in the last column of Table 1. We generated 10 expert trajectories for each environment.

Baselines and Codes. In order to evaluate the performance of our algorithm, we performed a comparison with two popular methods, namely Generative Adversarial Imitation Learning (GAIL) and Behavior Cloning (BC). It is worth noting that neither of these two methods considers cost constraints in their approaches. In the case of Behavioral Cloning, the expert trajectories dataset, which consists of state-action pairs, was divided into a 70% training data set and a 30% validation data set. The policy was then trained using supervised learning techniques. On the other hand, in the GAIL method, the policy network, value network, and discriminator network all employ the same architectures, comprising two hidden

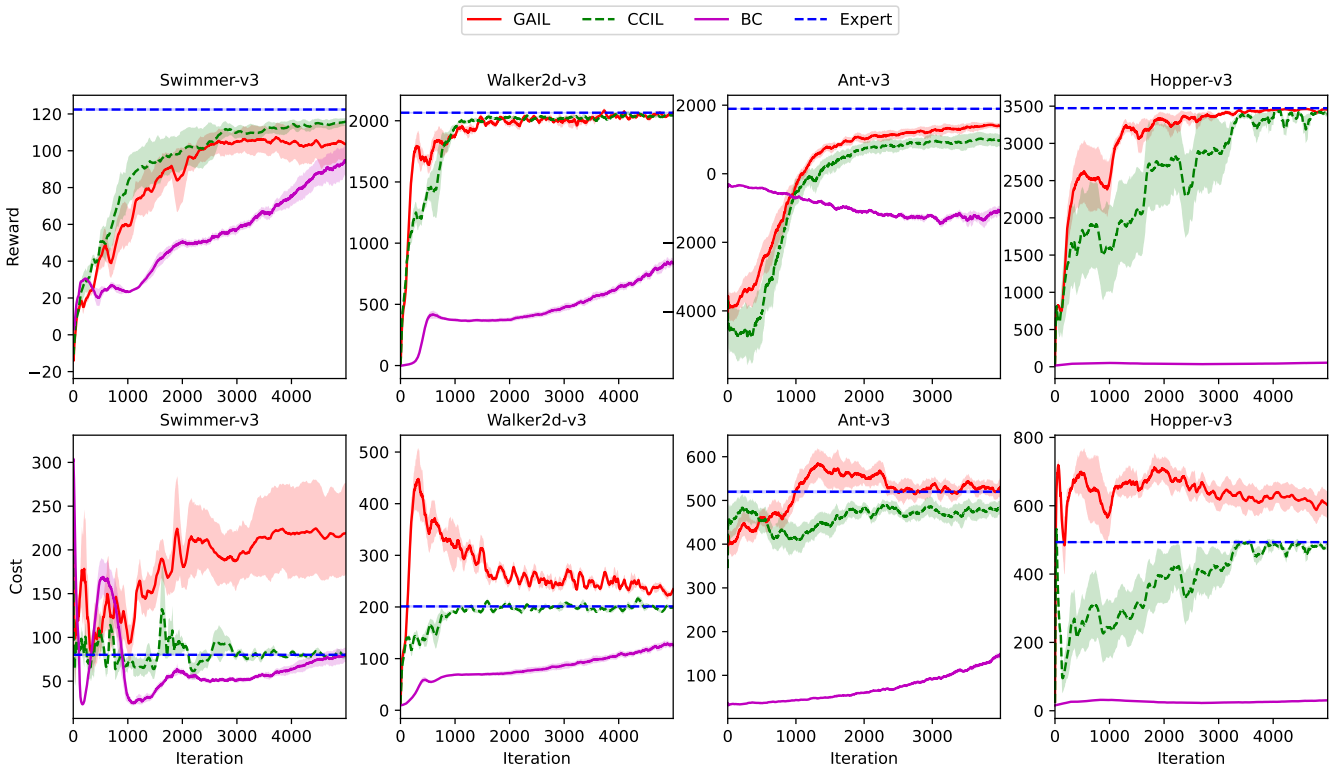

Figure 1: Performance over the training for GAIL, BC, and our approach (CCIL) all trained over 5 random seeds. The x-axes indicate the number of iterations, each iteration consists of 2000 timesteps interacting with the environment, and the y-axes indicate the performance of the agent, including average rewards/costs with standard deviations.

Environment		BC	GAIL	CCIL	Expert
Swimmer-v3	Reward	93.87±18.0	103.66±21.0	**115.68± 4.0**	122.41±2.41
	Cost	**79.52 ± 18.95**	219.24±114.73	**78.97 ±6.32**	80.10±3.78
Walker2d-v3	Reward	835.52±72.0	**2065.97±30.00**	2043.46±20.0	2065.71± 21.86
	Cost	**126.06 ± 13.54**	234.16±11.32	**198.99 ±4.15**	201.1±7.48
Ant-v3	Reward	-1072.5±336.0	**1403.35±180.00**	966.31±432.0	1895.15±57.81
	Cost	**147.64 ± 15.99**	530.75±47.78	**485.08 ±32.06**	519.9±11.28
Hopper-v3	Reward	54.36±16.0	**3456.06±12.0**	3393.90±85.0	3472.90±1.16
	Cost	**30.46 ± 5.23**	601.44±92.29	**486.85 ±26.33**	493.50±5.0

Table 1: Reward & Cost (mean ± std) an of best policy trained by GAIL, BC, and our approach (CCIL) in different MuJoCo games. The last column is the statistics of expert trajectories. In each column, we bold the best reward performance over all algorithms (higher is better), and bold the cost which is below the expert cost.

layers of 100 units each, with tanh nonlinearities being utilized in the layers.

Implementation. We employ a neural network architecture consistent with the one utilized in the GAIL method. However, our approach includes adding a cost value network and a Lagrangian penalty term, denoted by λ, which distinguishes our method from the GAIL method. The policy, value, and cost value network are optimized through gradient descent with the Adam optimizer [9]. The initial value of λ is set to 0.01 and also optimized using the Adam optimizer. We ran each algorithm for 5 different random seeds. The algorithms ran for 2000 time steps during each iteration, and the episode's total true reward and cost were recorded. The implementation of all codes was based on the OpenAI Baselines library [4].

5.2 Results

In order to assess the effectiveness of each algorithm, we used average episode true reward and average episode cost at each iteration as an evaluation. Here are the key observations from Figure 1:

- Even though BC also does not consider cost constraints, the cost of the policy obtained was generally lower than that of the expert cost. However, the reward obtained was well below the expert reward, except for the Swimmer environment.
- On the other hand, the GAIL method, which also neglects cost constraints, resulted in a scenario where the reward during the training process approached the expert reward. However, the cost was almost always higher than the expert cost.
- In contrast, our proposed method consistently maintained episode costs below the expert cost (except for the Swimmer environment, where it violated expert cost in initial episodes). Also, it achieved a true reward as close as possible to the expert reward.

Table 1 illustrates the performance of the optimal policy of all algorithms, which highlights that our proposed method almost achieved the highest reward while keeping all costs below the expert cost. The GAIL and BC methods fail to consider the cost constraints, which results in costs that exceed the expert cost for GAIL and low rewards for BC. Compared to these two methods, our method has a superior cost and reward optimization performance.

6 CONCLUSION

In this study, we address a novel challenge of solving the imitation learning problem within cost-constrained environments. To tackle this issue, we propose the Cost Constrained Imitation Learning method, which is both scalable and theoretically sound. We provide comprehensive theoretical justification for the objective utilized to handle imitation in cost-constrained environments. Our experiments demonstrate that our method can effectively imitate expert behavior while satisfying cost constraints, compared to other imitation learning methods that do not consider cost constraints.

REFERENCES

[1] Pieter Abbeel and Andrew Y Ng. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*. 1.

[2] Eitan Altman. 1999. *Constrained Markov decision processes: stochastic modeling*. Routledge.

[3] Michael Bain and Claude Sammut. 1995. A Framework for Behavioural Cloning.. In *Machine Intelligence 15*. 103–129.

[4] Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, Alex Nichol, Matthias Plappert, Alec Radford, John Schulman, Szymon Sidor, Yuhuai Wu, and Peter Zhokhov. 2017. OpenAI Baselines. https://github.com/openai/baselines.

[5] Bin Fang, Shidong Jia, Di Guo, Muhua Xu, Shuhuan Wen, and Fuchun Sun. 2019. Survey of imitation learning for robotic manipulation. *International Journal of Intelligent Robotics and Applications* 3, 4 (2019), 362–369.

[6] Chelsea Finn, Sergey Levine, and Pieter Abbeel. 2016. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*. PMLR, 49–58.

[7] Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning. *Advances in neural information processing systems* 29 (2016).

[8] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. 2017. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)* 50, 2 (2017), 1–35.

[9] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[10] Alex Kuefler, Jeremy Morton, Tim Wheeler, and Mykel Kochenderfer. 2017. Imitating driver behavior with generative adversarial networks. In *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 204–211.

[11] Shehryar Malik, Usman Anwar, Alireza Aghasi, and Ali Ahmed. 2021. Inverse constrained reinforcement learning. In *International Conference on Machine Learning*. PMLR, 7390–7399.

[12] P Warwick Millar. 1983. The minimax principle in asymptotic statistical theory. In *Ecole d'Eté de Probabilités de Saint-Flour XI—1981*. Springer, 75–265.

[13] Andrew Y Ng, Stuart Russell, et al. 2000. Algorithms for inverse reinforcement learning.. In *Icml*, Vol. 1. 2.

[14] Dean A Pomerleau. 1991. Efficient training of artificial neural networks for autonomous navigation. *Neural computation* 3, 1 (1991), 88–97.

[15] Martin L Puterman. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

[16] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 627–635.

[17] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International conference on machine learning*. PMLR, 1889–1897.

[18] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* (2015).

[19] Umar Syed, Michael Bowling, and Robert E Schapire. 2008. Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*. 1032–1039.

[20] Chen Tessler, Daniel J Mankowitz, and Shie Mannor. 2018. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074* (2018).

[21] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. MuJoCo: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 5026–5033. https://doi.org/10.1109/IROS.2012.6386109

[22] Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. 2010. Modeling interaction via the principle of maximum causal entropy. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*. 1255–1262.

[23] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. 2008. Maximum entropy inverse reinforcement learning.. In *Aaai*, Vol. 8. Chicago, IL, USA, 1433–1438.

A PROOFS

A.1 Prove concavity of $\bar{L}$

$\bar{L}(\cdot,(r,\lambda))$ is concave for all (r,λ).

Proof. We known that $\psi(r)$ is concave, suppose $\alpha \in [0,1]$.

$$\begin{aligned}
&\bar{L}(\cdot,(\alpha r_1+(1-\alpha)r_2,\alpha\lambda_1+(1-\alpha)\lambda_2)) = -\bar{H}(\rho)+\\
&\psi(\alpha r_1+(1-\alpha)r_2)+\\
&\sum_{s,a}(\rho_\pi-\rho_{\pi_E})[d(\alpha\lambda_1+(1-\alpha)\lambda_2)-(\alpha r_1+(1-\alpha)r_2)]\\
&\geq \alpha\psi(r_1)+(1-\alpha)\psi(r_2)+\alpha\sum_{s,a}(\rho_\pi-\rho_{\pi_E})(\lambda_1 d-r_1)\\
&+(1-\alpha)\sum_{s,a}(\rho_\pi-\rho_{\pi_E})(\lambda_2 d-r_2)
\end{aligned} \tag{28}$$

Therefore, $\bar{L}(\cdot,(\alpha r_1+(1-\alpha)r_2,\alpha\lambda_1+(1-\alpha)\lambda_2)) \geq \bar{L}(\cdot,(\alpha r_1,\alpha\lambda_1)+\bar{L}(\cdot,((1-\alpha)r_2,(1-\alpha)\lambda_2))$, $\bar{L}(\cdot,(r,\lambda))$ is concave for all (r,λ). □

A.2 Proof of $\psi^*_\phi(\rho_\pi-\rho_{\pi_E})$

We deduce the form of $\psi^*_\phi(\rho_\pi-\rho_{\pi_E})$ as:

$$\begin{aligned}
&\psi^*_\phi(\rho_\pi-\rho_{\pi_E}) =\\
&-R_\phi(\rho_\pi,\rho_{\pi_E})+\max_\lambda \lambda\sum_{s,a}(\rho_\pi(s,a)-\rho_{\pi_E}(s,a))d(s,a)
\end{aligned} \tag{29}$$

We will simplify notation by using the symbols ρ_π, ρ_{π_E}, r, and d to represent $\rho_\pi(s,a)$,$\rho_{\pi_E}(s,a)$,$r(s,a)$ and $d(s,a)$, respectively.

$$\begin{aligned}
&\psi^*_\phi(\rho_\pi-\rho_{\pi_E}) = \max_{\substack{r\in\mathcal{R}\\ \lambda}}\sum_{s,a}(\rho_\pi-\rho_{\pi_E})(\lambda d-r)-\psi_\phi(-r)\\
&=\max_{\substack{r\in\mathcal{R}\\ \lambda}}\sum_{s,a}(\rho_\pi-\rho_{\pi_E})(\lambda d-r)-\sum_{s,a}\rho_{\pi_E}g_\phi(-r)\\
&=\max_{r\in\mathcal{R}}\sum_{s,a}(\rho_\pi-\rho_{\pi_E})(-r)-\sum_{s,a}\rho_{\pi_E}(r+\phi(-\phi^{-1}(r)))\\
&+\max_\lambda\sum_{s,a}\lambda(\rho_\pi-\rho_{\pi_E})d\\
&=\max_{r\in\mathcal{R}}\sum_{s,a}\rho_\pi(-r)-\sum_{s,a}\rho_{\pi_E}\phi(-\phi^{-1}(r))\\
&+\max_\lambda\sum_{s,a}\lambda(\rho_\pi-\rho_{\pi_E})d
\end{aligned} \tag{30}$$

Then we make the change of variables $r \to \phi(\gamma)$, the above equation becomes:

$$\begin{aligned}
&\psi^*_\phi(\rho_\pi-\rho_{\pi_E}) =\\
&\sum_{s,a}\max_{\gamma\in\mathbb{R}}\rho_\pi(-\phi(\gamma))-\rho_{\pi_E}\phi(-\gamma)+\max_\lambda\lambda\sum_{s,a}(\rho_\pi-\rho_{\pi_E})d\\
&=-R_\phi(\rho_\pi,\rho_{\pi_E})+\max_\lambda\lambda\sum_{s,a}(\rho_\pi-\rho_{\pi_E})d
\end{aligned} \tag{31}$$