Learning to Maintain Safety Through Expert Demonstrations in Settings with Unknown Constraints: A Q-Learning Perspective.

George Papadopoulos
University of Piraeus
Piraeus, Greece
georgepap@unipi.gr

George A. Vouros
University of Piraeus
Piraeus, Greece
georgev@unipi.gr

ABSTRACT

Given a set of trajectories demonstrating the execution of a task safely in a constrained MDP with observable rewards but with unknown constraints and non-observable costs, we aim to find a policy that maximizes the likelihood of demonstrated trajectories trading the balance between being conservative and increasing significantly the likelihood of high-rewarding trajectories but with potentially unsafe steps. Having these objectives, we aim towards learning a policy that maximizes the probability of the most *promising* trajectories with respect to the demonstrations. In so doing, we formulate the “promise" of individual state-action pairs in terms of Q values, which depend on task-specific rewards as well as on the assessment of states' safety, mixing expectations in terms of rewards and safety. This entails a safe Q-learning perspective of the inverse learning problem under constraints: The devised Safe Q Inverse Constrained Reinforcement Learning (SafeQIL) algorithm is compared to state-of-the art inverse constraint reinforcement learning algorithms to a set of challenging benchmark tasks, showing its merits.

KEYWORDS

Q-Learning; Inverse Constrained RL; ICRL; SafeQIL

ACM Reference Format:
George Papadopoulos and George A. Vouros. 2026. Learning to Maintain Safety Through Expert Demonstrations in Settings with Unknown Constraints: A Q-Learning Perspective.. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/RJIB1203

1 INTRODUCTION

In this paper, we address the inverse problem of learning safe policies given expert trajectories following underlying constraints and demonstrating safe execution of tasks under observable rewards and non-observable costs. We term this as an inverse learning problem with respect to constraints, given that the set of constraints are unknown, and although we do not aim to approximate the set of constraints or cost functions that determine the set of expert demonstrations safe, we aim to assess the safety of states and learn policies that support agents to act safely not only by following the demonstrated trajectories, but also in states not in the support of those demonstrated. This perspective distinguishes this work from inverse constrained reinforcement learning approaches that aim to recover the minimal constraint sets or the least constraining constraints explaining the demonstrated behavior [11, 16].

Considering possible tradeoffs between rewards and constraints in executing a task, in settings with non-observable costs, agents may either learn very conservative policies that avoid states and actions not included in the demonstrated trajectories, or policies that increase significantly the likelihood of trajectories that cross states with high uncertainty about their safety but with high expected rewards. These will be the cases when, for instance, assessing the feasibility of agents' behavior at a trajectory level in a conservative manner, or when the values of state-action pairs not in the demonstrated trajectories are much larger than those in the demonstrated trajectories in terms of the expected reward. This means that in the first case the task execution will be punished as a whole and the agents shall not have ability to recover safety at any trajectory step, and in the later case the policy will prefer to cross states and perform high-rewarding actions in state-action space areas with high uncertainty about safety.

To address these phenomena, we express the likelihood of trajectories in terms of Q-values of individual state-action pairs, mixing expectations in terms of rewards and safety.

This approach aims towards policies that indicate the state-action pairs demonstrated to be the most likely ones, without increasing much the likelihood of trajectories that cross states that are not in the support of demonstrated state-action pairs (i.e., being probably unsafe), and without being conservative. The basic idea is that Q values mixing expected rewards and safety should indicate actions as highly promising when leading to subsequent states assessed to be safe. This supports agents to learn how to recover safety, targeting to safe states, even when they are in states of high uncertainty about their safety.

Although costs for violating constraints, or constraints-related rewards, are not revealed explicitly or approximated, to assess states' safety we use a discriminator function that estimates the probability of a state to be included in the distribution of states demonstrated. This brings this approach closer to other inverse constrained reinforcement learning approaches [11, 16], but with a different objective regarding the likelihood of trajectories and significant differences in performance.

Finally, it must be noted that mixing rewards and safety in Q values is different from reward-shaping approaches, where cost functions are assumed to be known: In contrast to that, we evaluate actions performed in states on their expectation to maintain safety, and distinctly to that, in terms of their expected rewards.

The contributions that this article makes are as follows:

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/RJIB1203

(a) It formulates the problem of learning a policy with respect to expert (safe) demonstrated trajectories as an inverse constrained reinforcement learning problem, whose objective function is rigorously specified in terms of Q-values of trajectory steps incorporating assessments on the safety of states, mixing expectations in terms of rewards and safety.

(b) It proposes the safe Q Inverse Constrained Reinforcement Learning (SafeQIL) algorithm[1].

(c) It presents evaluation results for SafeQIL in settings with constraints of increasing complexity. These are compared to results from state of the art imitation and inverse constrained reinforcement learning algorithms.

2 PRELIMINARIES AND MOTIVATION

A Markov Decision Process (MDP) is a tuple (S, A, R, P, γ), where S is the set of states, A is the set of actions available to an agent, $r : S \times A \rightarrow \mathbb{R}$ is the reward function, $P : S \times A \times S \rightarrow [0, 1]$ is the transition probability function $P(s_{t+1}|a_t, s_t)$ to a new state s_{t+1} after the execution of action a_t in state s_t, and $\gamma \in [0, 1]$ is the discount factor. In such a setting, an agent aims to learn the optimal policy $\pi^* : S \rightarrow \mathcal{P}(A)$ from states to probability distributions over actions, so as to maximize the performance measure

$$J(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)] + \alpha\mathcal{H}(\pi) \quad (1)$$

where τ denotes any trajectory generated using the policy π^*, with $s_0 \sim \mu$ (the distribution of initial states), $a_t \sim \pi(\cdot|s_t), s_{t+1} \sim P(\cdot|s_t, a_t)$, and $\mathcal{H}(\pi)$ represents the policy entropy weighted by α.

A constrained Markov decision process (CMDP) is an MDP with constraints that restrict the set of feasible policies for the MDP. Thus, the MDP is augmented with a set of constraint functions $C = \{C_i, i = 1..., m\}$, and corresponding limits $d_i, i = 1..., m$. Each $C_i : S \times A \rightarrow \mathbb{R}$ maps the execution of actions at states to costs. In such a setting, the set of feasible policies with respect to constraints is $\Pi_C = \{\pi \in \Pi|\forall i, J_{C_i}(\pi) \leq d_i\}$ [2] where:

$$J_{C_i}(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t C_i(s_t, a_t)]$$

denotes the i-th constraint-related discounted cost of policy π. In such a setting with known or predefined constraints, the constrained reinforcement learning objective is to learn the performance $J(\cdot)$ maximizing a feasible policy. Formally,

$$\pi^* = argmax_{\pi\in\Pi_C} J(\pi).$$

However, as it is well known, in many settings constraints can not be formulated or even expressed by human experts, being inherent in their expertise. Therefore, such information cannot be revealed to an agent in an explicit way, but can only be inferred through demonstrations. In such settings of unknown or implicit constraints, the inverse constrained reinforcement learning (ICRL) problem aims at fitting constraint functions $\tilde{C}_i$, given a set of sample trajectories $\mathcal{D}_E$ generated by an expert policy π_E, solving the optimization problem:

$$max_{\tilde{C}_i}(J_{\pi\in\Pi_{\tilde{C}_i}}(\pi) - J_{\tilde{C}_i}(\pi_E)).$$

Such a process, as shown in [13], iterates through constraint update (the inverse step) and policy update steps, given the set of expert trajectories $\mathcal{D}_E$. ICRL aims to model the expert policy π_E

$$\tilde{\pi_E} = argmax_{\pi\in\Pi_{\tilde{C}}} J(\pi)$$

using a constrained reinforcement learning method, where $\tilde{C}$ is the set of approximated constraint functions $\tilde{C}_i$.

Actually, given that the demonstrated trajectories may be explained by different sets of constraints, ICRL looks for the minimal set of constraint functions that maximizes the likelihood of demonstrations. Thus, it aims to assign a low cost to the trajectories generated by the expert policy, which must be feasible (i.e., within cost limits), and high cost to trajectories generated by any other feasible policy. Finding this set of constraints among the combinations of potential constraints is an intractable problem where greedy solutions can be used [19]. Instead of doing so, as proposed in [16], we may define a (learnable) function $\phi : S \times A \rightarrow [0, 1]$ that indicates the probability of an action performed in a state to be safe (i.e., within the distribution of expert demonstrations). This function may be used for calculating the probability of satisfying trajectory-level safety constraints $C(\tau) = 1 - \prod_{(s,a)\in\tau} \phi(s, a) = 1 - \mathbb{I}^\phi(\tau)$.

In doing so, this approach, although termed as an inverse approach, does not aim to reveal the set of constraints explaining the demonstrated trajectories, but to find the policy that increases in a direct manner the likelihood of demonstrated trajectories. This results into training this discrimination function ϕ_ω, parameterized by ω, with the following objective:

$$max_\omega logp(\mathcal{D}_E|\phi) = max_\omega log\big[\frac{1}{Z_\phi^{|\mathcal{D}_E|}} \prod_{i=1}^{|\mathcal{D}_E|} exp[r(\tau^i)]\mathbb{I}^\phi(\tau^i)\big]$$

$$\text{with } Z_\phi = \int \mathbb{I}^\phi(\tau)exp(r(\tau))\, d\tau, \text{ and } \mathbb{I}^\phi(\tau^i) = \prod_{t=1}^{T} \phi_\omega(s_t^i, a_t^i)$$

However, formulating the objective in this way implies the following:

- The assessment of the feasibility of any trajectory by means of $\mathbb{I}^\phi(\tau)$ is very strict, given that in case a single step (state-action pair) is not in the distribution of those demonstrated, this can be considered to be unsafe, and this may result into $C(\tau)$ close to1. This may result to a very conservative agent behavior.
- Highly rewarding trajectories, beyond those demonstrated, with state-action pairs to which ϕ_ω is approximately 0.5 may lead the agent to unsafe behavior, considering the trajectory as highly promising.
- Indeed, from a stepwise perspective, there may exist individual state-action pairs to which the agent is uncertain about their safety, but they do contribute significantly to the trajectory cumulative reward and thus, to the likelihood of trajectories. Such actions in states may actually be unsafe.

[1]Our implementation, the generated dataset, and the supplementary material are publicly available in: https://github.com/AILabDsUnipi/SafeQIL .

[2]Here, $d_i, i = 1, ..., m$ denote trajectory-level cost limits, although in alternative formulations they may denote stepwise limits. Alternatively, we may define trajectory level constraints of the form $C_i(\tau)$, where τ is a trajectory generated by a policy π. Then $J_{C_i}(\pi)$ is of the form $\mathbb{E}_{\tau\sim\pi}[C_i(\tau)]$

Therefore, considering the problem at the trajectory-level, the agent does not have the flexibility to choose individual actions that would allow it to maintain safety, increasing the likelihood of trajectories in terms of safety, even if some of the states crossed are potentially unsafe.

3 PROBLEM SPECIFICATION

To address the above mentioned issues of a trajectory-level inverse constrained reinforcement learning approach, we aim at maximizing the probability of demonstrated trajectories by considering the Q values of individual state-action pairs ($Q : S \times A \rightarrow \mathbb{R}$) comprising these trajectories, mixing the expectation regarding the performance of actions in states on rewards and safety. For the simplicity of presentation, with an abuse of notation, we subsequently use $\mathcal{P}^E$ to denote the distribution of demonstrated steps, i.e., state-action pairs, as well as the distribution of states crossed by any $\tau \in \mathcal{D}_E$. Similarly, we denote $supp^E$ the support set of state-action pairs, as well as the support set of states in any $\tau \in \mathcal{D}_E$.

To formulate the Q function and the final objective, we distinguish two types of reward values: The task-specific reward values (denoted r_d) provided by the environment per step (i.e., $r_d : S \times A \rightarrow \mathbb{R}$), and the safety rewards (denoted r_s) on states (i.e., $r_s : S \rightarrow \mathbb{R}$). The latter type of reward should be considered as a “constraint-abiding" bonus or as a penalty for being in an unsafe state. Without loss of generality, we can assume that for any $(s, a) \in S \times A$, it holds that $r_s(s) \leq r_d(s, a)$. For instance, we can assume, without loss of generality, that $r_s(\cdot) \leq 0$ and $r_d(\cdot) \geq 0$.

Now, the Q function is defined to be the expected sum of rewards when the agent performs action a_t in state s_t and in subsequent steps it acts according to the policy π, given the environment dynamics determined by P:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{\pi,P} \sum_{i=0}^{T-t-1} \gamma^i R_{t+i} \tag{2}$$

where,

$$R_t = [\mathbb{I}^S(s_t) r_d(s_t, a_t) + (1 - \mathbb{I}^S(s_t)) r_s(s_t)]$$

and $\mathbb{I}^S(s_t)$ specifies an assessment of whether the state s_t is safe. This can be a binary assessment of whether $(s, a) \in supp^E$ [3].

Given the above specification, the Q value of a state-action pair is an accumulation of a mixture of discounted “task specific" and “safety" rewards, where the latter are action-independent. This specifies the safety of a state and the promise of an action a_t performed in state s_t in terms of rewards. To see this, let us consider that starting from (s_t, a_t), a trajectory crosses only unsafe states. Then, $\mathbb{I}^S(s_i) = 0, \forall i \in \{t, t+1, ...\}$, and the $Q(s_t, a_t)$ value accumulates only penalties for these states. On the other hand, in case at some point $t' \geq t$ the agent performs an action that recovers safety and then it continues using a policy that maintains safety, then the Q value from that point and on accumulates task specific positive rewards. Based on this, we can prove the following theorem:

Theorem: When for any $(s, a) \in S \times A$ with $(s, a) \notin supp^E$ it holds that $r_s(s) \leq 0$, and $\mathbb{I}^S$ is binary, then the following holds for any policy π:

$$Q^\pi(s, a) \leq min_{(s^d,a^d) \in supp^E}\{Q^{\pi_d}(s^d, a^d)\} \tag{3}$$

where $s^d \in supp^E$ are states that are the closest ones (based on a measure of proximity) to $s \notin supp^E$ that the agent can reach in subsequent steps, and Q^{π_d} are the Q values under the demonstrated policy. It must be noted that this inequality holds in expectation according to the policy π and environment dynamics P.

Proof. To prove this, let s_t be a state not in $supp^E$. The policy π, starting from any state, does not guarantee that it results in trajectories that cross only safe states. Since $\mathbb{I}^S$ is binary and $s_t \notin supp^E$, then $\mathbb{I}^S(s_t) = 0$ and $R_t = r_s(s_t)$. Thus, it holds that:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{\pi,P}[r_s(s_t) + \sum_{i=1}^{T-t-1} \gamma^i R_{t+i}]$$

In case the demonstrated policy, π_d, starts from any state s_t^d where it applies $a_t^d \in A$, s.t. $(s_t^d, a_t^d) \in supp^E$, then

$$Q^{\pi_d}(s_t^d, a_t^d) = \mathbb{E}_{\pi^*,P}[r_d(s_t^d, a_t^d) + \sum_{i=1}^{T-t-1} \gamma^i R'_{t+i}]$$

Since $(s^d, a^d) \in supp^E$, then $\mathbb{I}^S(s^d, a^d) = 1$ and it holds that $R'_t = r_d(s_t^d, a_t^d)$. Now, the assumption that for any state s, $r_s(s) \leq 0$, implies that $R_t \leq r_d(s_t^d, a_t^d)$, for any $(s_t^d, a_t^d) \in supp^E$. Hence, $R_t \leq min_{(s^d,a^d) \in supp^E} r_d(s_t^d, a_t^d)$. This is the case for any subsequent state not in $supp^E$ visited by π starting from s_t in comparison to states visited by the demonstrated policy. For states in $supp^E$ visited by π, their Q values cannot be greater than those visited by the policy π_d, i.e., for $s_t^d \in supp^E$, it holds that $Q^\pi(s_t^d, \pi(s_t^d)) \leq Q^{\pi_d}(s_t^d, a_t^d)$. Therefore, the following holds, since, π_d crosses only safe states, while π may cross unsafe states:

$$Q^\pi(s_t, a_t) \leq min_{(s^d,a^d) \in supp^E}(r_d(s^d, a^d)) + \mathbb{E}_{\pi,P}[\gamma Q^\pi(s_{t+1}, a_{t+1})]$$
$$\leq min_{(s^d,a^d) \in supp^E}\{Q^{\pi_d}(s^d, a^d)\}$$

Inequality (3) formulates the idea that state-action pairs known to be safe should have the highest promise in terms of rewards (and safety). Unfortunately, this inequality is not guaranteed to hold in case $\mathbb{I}^S$ is probabilistic. Therefore, we need to (a) maximize the Q values of state-action pairs that are in the support of demonstrations (i.e., known to be safe), and to (b) maintain the Q values of actions in states that are not in the support of demonstrated states, lower than those in $supp^E$, *in expectation* to the policy, environment dynamics and the uncertainty about the safety of states. In so doing, we aim to prevent trajectories not in the demonstrations from being highly probable, and support agents to find actions that, when executed in probably unsafe states, can recover safety with high probability, with respect to the environment dynamics.

We can now formulate the objective of the inverse constrained reinforcement learning in terms of Q values. In stochastic, continuous domains with respect to the dynamics P, we know that we need to maximize:

$$\sum_t \mathbb{E}_{\pi,P}[R_t + \mathcal{H}(\pi(a_t|s_t))],$$

where the expectation is in the distribution of state-actions under the policy π and environment dynamics. This is optimized by choosing $\pi(a_t|s_t) = exp(Q(s_t, a_t) - V(s_t))$, with a soft maximization of

[3] It must be noted that a binary $\mathbb{I}^S$ does not necessarily assess whether the state is in the $supp^E$.

the value function: $V(s_t) = log \int exp(Q(s_t, a_t))da_t$. To evaluate the policy according to the maximum entropy objective, given a policy π, we update Q values following the Bellman update:

$$Q(s_t, a_t) = R_t + \gamma \mathbb{E}_{s_{t+1} \sim P} V(s_{t+1})$$

where the soft value function is as follows:

$$V(s_t) = \mathbb{E}_{a_t \sim \pi}[Q(s_t, a_t) - log\pi(a_t|s_t)],$$

transforming the policy entropy specified in (1) to a causal entropy on individual steps.

Therefore, considering that $\pi(a_t|s_t) \propto exp(Q(s_t, a_t))$, for the joint probability of actions (CQ) in a trajectory τ it holds that:

$$CQ(\tau) \propto \prod_{t=0}^{T} exp(Q(s_t, a_t)),$$

Thus, it holds that:

$$logp(\mathcal{D}_E) \propto log[\prod_{i=1}^{|\mathcal{D}_E|}(\prod_{t=0}^{T} exp(Q(s_t^i, a_t^i))]$$

We aim to maximize the likelihood of demonstrated trajectories by maximizing the following term:

$$max[\sum_{i=1}^{|\mathcal{D}_E|}\sum_{t=0}^{T} Q(s_t^i, a_t^i)] \tag{4}$$

subject to:

$$Q(s, a) \preceq min_{(s', a') \in supp^E}\{Q(s', a')\} \tag{5}$$

for any $(s, a) \in S \times A$ with $s \notin supp^E$. The symbol $\preceq$ denotes that the inequality (5) holds in expectation, depending on the assessment $\mathbb{I}^S$ on states, the policy π, and the environment dynamics P. This does not prevent pairs $(s, a) \notin \mathcal{P}^E$ from having Q values that are greater than the minimum Q value of the demonstrated steps.

4 SAFE Q-LEARNING

4.1 Objective function

Guided by the max-entropy formulation and the step-wise likelihood view in Section 3, we explicitly enforce the constraint (5) in expectation on states $s \notin \mathcal{P}^E$, and update Q-values on states in or out of $\mathcal{P}^E$. In so doing, we allow the agent to learn both from demonstrations and online interactions. Concretely, let B be a buffer of policy rollouts and D the demonstration buffer.

Getting samples from B, these may be in $\mathcal{P}^E$ or not. In the latter case, we need to enforce the constraint (5). To do this, for each state s_B in a sample $(s_B, a_B, r_d(s_B, a_B), s'_B, a'_B)$ from B, we search and pick a sample from D with the "closest" state s_D^* (we elaborate on this in Section 4.2), together with its accompanying action a_D^*, task reward $r_d^*(s_D^*, a_D^*)$, and next state $s_D'^*$. In addition, we get the next action $a_D'^*$ and use the estimated Q-target value of $(s_D'^*, a_D'^*)$,

$$\hat{Q}_{min}(s_B, a_B) = r_d^*(s_D^*, a_D^*) + \gamma Q(s_D'^*, a_D'^*),$$

as a local bound target for $Q(s_B, a_B)$[4]. Then, the objective for samples from B is as follows:

$$\mathcal{L}_Q^B = \sum_{(s_B, a_B, r_{d,B}, s'_B, a'_B) \sim B} [(1 - \mathbb{I}^S(s_B)) \cdot LC^B_{s_B \notin \mathcal{P}^E}(s_B, a_B) + \mathbb{I}^S(s_B) \cdot L^B_{s_B \in \mathcal{P}^E}(s_B, a_B, r_{d,B}, s'_B, a'_B)] \tag{6}$$

where:

$$LC^B_{s_B \notin \mathcal{P}^E}(s_B, a_B) = \Big(max\Big(Q(s_B, a_B), \hat{Q}_{min}(s_B, a_B)\Big) - \hat{Q}_{min}(s_B, a_B)\Big)^2$$

and:

$$L^B_{s_B \in \mathcal{P}^E}(s_B, a_B, r_{d,B}, s'_B, a'_B) = \big(Q(s_B, a_B) - [r_d(s_B, a_B) + \gamma Q(s'_B, a'_B)]\big)^2$$

where the first term in brackets in Equation 6 enforces the constraint. However, we also have to include the safety reward term in case $s_B \notin \mathcal{P}^E$:

$$L^B_{s_B \notin \mathcal{P}^E}(s_B, a_B, s'_B, a'_B) = \big(Q(s_B, a_B) - [r_s(s_B) + \gamma Q(s'_B, a'_B)]\big)^2$$

So, now the objective becomes:

$$\mathcal{L}_Q^B = \sum_{(s_B, a_B, r_{d,B}, s'_B, a'_B) \sim B} [(1 - \mathbb{I}^S(s_B)) \cdot LC^B_{s_B \notin \mathcal{P}^E}(s_B, a_B) + (1 - \mathbb{I}^S(s_B)) \cdot L^B_{s_B \notin \mathcal{P}^E}(s_B, a_B, s'_B, a'_B) + \mathbb{I}^S(s_B) \cdot L^B_{s_B \in \mathcal{P}^E}(s_B, a_B, r_{d,B}, s'_B, a'_B)] \tag{7}$$

To recap, the above objective covers: (a) the task-specific reward term and the safety reward term, both weighted by the safety estimation, as proposed in Equation 2, and (b) it enforces the constraint of Equation 5. Based on previous work [4] and on our preliminary experiments, we found it beneficial to introduce a bias in Q-values towards the demonstrations' expected reward. Therefore, we incorporate the term $\mathcal{L}_Q^D$ that explicitly updates the Q-values based on samples from D comprising the demonstrated state-action pairs (s_D, a_D), the next state-action pairs (s'_D, a'_D), and the corresponding received task reward $r_d(s_D, a_D)$:

$$\mathcal{L}_Q^D = \sum_{(s_D, a_D, r_{d,D}, s'_D, a'_D) \sim D} L^D(s_D, a_D, r_d(s_D, a_D), s'_D, a'_D)$$

where:

$$L^D(s_D, a_D, r_d(s_D, a_D), s'_D, a'_D) = \big(Q(s_D, a_D) - [r_d(s_D, a_D) + \gamma Q(s'_D, a'_D)]\big)^2$$

This term is not weighted by the discriminator's estimate since the samples come from the demonstrations' distribution. So, the final objective, given N samples from B and N samples from D, is formulated as follows:

$$\mathcal{L}_Q = \frac{1}{2N}[\mathcal{L}_Q^B + \mathcal{L}_Q^D] \tag{8}$$

[4]In this section, we deliberately omit the entropy term. In addition, we omit details on how exactly s_D^*, a_D^*, $r_{d,D}^*$, $s_D'^*$, $a_D'^*$ are found, and whether we use the next action a'_D from the buffer or from the current policy (as, for instance, in SAC). These details are included in Section 4.2.

4.2 The SafeQIL algorithm

In this section, we provide a practical algorithm using the proposed objective and SAC as the backbone algorithm.

We maintain a stochastic policy (actor) $\pi_\theta(a|s)$, two critics Q_{ϕ_1}, Q_{ϕ_2} and two target critics $Q_{\bar{\phi}_1}$, $Q_{\bar{\phi}_2}$ as well as, a discriminator ϕ_ω, that estimates the probability for a state to be in $\mathcal{P}^E$. Also, the safety reward/penalty $r_s(s)$ is calculated based on the discriminator's estimate $\phi_\omega(s)$. Specifically, we choose $r_s(s) = log(\phi_\omega(s))$ which maps the discriminator's estimate (range $[0,1]$) to a negative safety reward (range $[-\infty, 0]$).

Online rollouts are stored in B and demonstrations in D. During the update of Q networks (critics), we get N samples from B and N samples from D, and each critic is updated separately using Equation 8. An important detail about Equation 8 requiring clarification is about how we retrieve (s^*_D, a^*_D, r^*_D, s'^*_D,a'^*_D) from D given a state s_B. Actually, we get the index of the sample s^*_D that minimizes the cosine similarity to s_B:

$$i^*_D = \arg\min\left(\mathrm{CosSim}(\mathbf{s_B}, \mathbf{S_D})\right) \tag{9}$$

Having the index i^*_D, it is straightforward to slice buffer D and get the corresponding sample for the calculation of $\hat{Q}_{min}$.

Continuing on the details about the entropy and Q-networks weights, assuming that we update critic j:

$$LC^B_{j,s_B \notin \mathcal{P}^E}(s_B, a_B) = \tag{10}$$
$$\left(max\left(Q_{\phi_j}(s_B, a_B), \hat{Q}_{min}(s_B, a_B)\right) - \hat{Q}_{min}(s_B, a_B)\right)^2$$

where (with an abuse of notation for simplicity of presentation):

$$\hat{Q}_{min}(s_B, a_B) = r^*_d(s_B, a_B) + \gamma \cdot \min_{i=1,2} Q_{\bar{\phi}_i}(s'^*_D, a'^*_D)$$

The next action a' is predicted based on the current policy for $L^B_{s_B \notin \mathcal{P}^E}(s_B, a_B, r_{d,B}, s'_B, a'_B)$ (i.e., the term for states out of $\mathcal{P}^E$), while for $L^D(s_D, a_D, r_d(s_D, a_D), s'_D, a'_D)$ (i.e., the term for states in $\mathcal{P}^E$) we use the one stored in D. In so doing, we omit the entropy term in both terms $L^B_{s_B \notin \mathcal{P}^E}(s_B, a_B, r_{d,B}, s'_B, a'_B)$ and $L^D(s_D, a_D, r_d(s_D, a_D), s'_D, a'_D)$ aligned with $LC^B_{j,s_B \notin \mathcal{P}^E}(s_B, a_B)$. These terms for the critic j can be expressed as follows:

$$L^B_{j,s_B \notin \mathcal{P}^E}(s_B, a_B, s'_B, a'_B) = \tag{11}$$
$$\left(Q_{\phi_j}(s_B, a_B) - [r_s(s_B) + \gamma \cdot \min_{i=1,2} Q_{\bar{\phi}_i,\ a' \sim \pi_\theta(\cdot|s'_B)}(s'_B, a')]\right)^2$$

and

$$L^D_j(s_D, a_D, r_d(s_D, a_D), s'_D, a'_D) = \tag{12}$$
$$\left(Q_{\phi_j}(s_D, a_D) - [r_d(s_D, a_D) + \gamma \cdot \min_{i=1,2} Q_{\bar{\phi}_i}(s'_D, a'_D)]\right)$$

Regarding $L^B_{s_B \in \mathcal{P}^E}(s_B, a_B, r_{d,B}, s'_B, a'_B)$, we don't perform any change to the backbone SAC algorithm, to maintain its strengths, so this term for the critic j is expressed as follows:

$$L^B_{j,s_B \in \mathcal{P}^E}(s_B, a_B, r_{d,B}, s'_B, a'_B) = \tag{13}$$
$$(Q_{\bar{\phi}_j}(s_B, a_B) -$$
$$[r_d(s_B, a_B) + \gamma\left(\min_{i=1,2} Q_{\bar{\phi}_i,\ a' \sim \pi_\theta(\cdot|s'_B)}(s'_B, a') - \alpha \log \pi_\theta(a'|s'_B)\right)])^2$$

where α is the entropy coefficient. To update the policy, we maximize the soft objective:

$$J_{\pi_\theta} = \frac{1}{N} \sum_{s_B \sim B} \left(\min_{i=1,2} Q_{\bar{\phi}_i,\ a \sim \pi_\theta(\cdot|s_B)}(s_B, a) - \alpha \log \pi_\theta(a|s_B)\right) \tag{14}$$

while the auto-tuned entropy coefficient is updated using the loss defined in [3]:

$$\mathcal{L}_\alpha = \frac{1}{N} \sum_{s_B \sim B} \left(-\left(\alpha \log \pi_\theta(a|s_D) + \bar{\mathcal{H}}\right)\right) \tag{15}$$

where $\bar{\mathcal{H}}$ is a hyperparameter. Each target critic j is updated using the Polyak averaging with the corresponding parameter η:

$$\bar{\phi}_j \leftarrow \eta \bar{\phi}_j + (1-\eta)\phi_j \tag{16}$$

Finally, at each update of all the aforementioned, we also update the discriminator using the Logistic Loss as used in DAC [7], the Gradient Penalty regularization of [2], N samples from B and N samples from D:

$$\mathcal{L}_\omega = \tag{17}$$
$$\frac{1}{2N}\left(\sum_{s_B \sim B}[-\log(1-\phi_\omega(s_B))] + \sum_{s_D \sim D}[-\log(\phi_\omega(s_D))]\right)$$
$$+ \lambda_{\mathrm{gp}} \cdot \frac{1}{2N} \sum_{(s_B, s_D)} \left(\|\nabla_{\hat{s}} \phi_\omega(\hat{s})\|_2 - 1\right)^2$$

where:

$$\hat{s} = \epsilon\, s_D + (1-\epsilon)\, s_B, \qquad \epsilon \sim \mathcal{U}(0,1).$$

We summarize all the above in Algorithm 1. In summary, the method couples a support-aware constraint with max-entropy policy learning: demonstrations provide local upper bounds for the value function of state-action pairs not in the distribution of demonstrations, the discriminator softly gates updates so that online interactions improve performance on in-distribution states, while a safety reward drives recovery off-distribution. This yields a modification of standard soft actor–critic training that curbs unsafe extrapolation, preserves performance in states in the support of demonstrations, and, in special cases, it can be reduced to SAC if all states are in-distribution of demonstrations.

5 EXPERIMENTS

5.1 Experimental settings

Tasks. We evaluate SafeQIL on 4 tasks from Safety-Gymnasium: SafetyPointGoal1-v0, SafetyPointCircle2-v0, SafetyCarButton1-v0, and SafetyCarPush2-v0, covering navigation and object interaction under safety constraints. Figure 1 provides representative snapshots of the 4 experimental settings to illustrate task geometry and typical safety-related objects. This set of tasks spans increasing difficulty and safety density, that is, from boundary-only constraints (SafetyPointCircle2-v0) to multi-obstacle navigation (SafetyPointGoal1-v0) and interaction-heavy control (SafetyCarButton1-v0/SafetyCarPush2-v0), providing a broad stress-test for algorithms that must (i) improve in-distribution while (ii) remaining conservative at out-of-distribution states.

Demonstrations. To train the algorithms, we collect human-generated demonstration datasets for each task via keyboard control, producing (state, action, reward, cost, next state, next action)

Algorithm 1 SafeQIL

Require: Demonstration buffer D
Initialize Policy π_θ; Critics Q_{ϕ_1}, Q_{ϕ_2}; Target critics $Q_{\bar{\phi}_1}, Q_{\bar{\phi}_2}$; Entropy coefficient α; Discriminator ϕ_ω; Replay buffer $B \leftarrow \varnothing$
for each environment step **do**
  Observe s_t; Sample and apply $a_t \sim \pi_\theta(\cdot|s_t)$; Get $(r_{d,t}, s_{t+1})$
  Push $(s_t, a_t, r_{d,t}, s_{t+1})$ into B
  for update step $u = 1, \ldots, U$ **do**
    Sample N tuples $(s_B, a_B, r_{d,B}, s'_B) \sim B$
    Sample N tuples $(s_D, a_D, r_{d,D}, s'_D) \sim D$
    For each s_B get index i^*_D (Eq. 9)
    For each i^*_D retrieve $(r^*_{d,D}, s'^*_D, a'^*_D)$ slicing D
    Sample $a'_B \sim \pi_\theta(\cdot|s'_B)$
    Update discriminator ϕ_ω (Eq. 17)
    Update critics Q_{ϕ_1}, Q_{ϕ_2} (Eq. 8, 10, 11, 12, 13)
    Update policy π_θ (Eq. 14)
    Update entropy coefficient α (Eq. 15)
    Update target critics $Q_{\bar{\phi}_1}, Q_{\bar{\phi}_2}$ (Eq. 16)

tuples. To the best of our knowledge, existing offline safe-RL benchmarks [14] provide algorithm-generated datasets across 38 Safety-Gymnasium / BulletSafetyGym / MetaDrive tasks, but there is no publicly available human-generated demonstration set for Safety-Gymnasium tasks.

Baselines. We compare SafeQIL against 3 strong baselines:

- ICRL [16] learns an explicit constraint function from demonstrations and then optimizes a policy under the inferred constraints. Our implementation relies on the official ICRL implementation [15].

- VICRL [11] models a posterior distribution over constraints via variational inference, and our implementation is based on the official VICRL implementation [12].

- SAC-GAIL: a GAIL-style discriminator provides the reward while the actor and critic are trained with SAC. We implement this in the spirit of off-policy AIL, such as DAC [6], which couples a GAIL discriminator with an off-policy actor-critic. This baseline tests whether pure imitation suffices on our tasks, as opposed to SafeQIL's value-shaping with state-level pessimism.

ICRL and VICRL were extensively tuned using the complete set of configurations from their original papers/implementations (3 and 9 settings, respectively) on SafetyPointGoal1-v0. Because the best ICRL configuration (Setting 2) involves a low number of environment steps, we also evaluate ICRL using the best VICRL parameters (Setting 7), tuning the regularization coefficient per task for both configurations. We report results for both, denoted as ICRL^2 and ICRL^7. For SafeQIL and SAC-GAIL, we utilized the Stable Baselines3 [18] SAC implementation. We use default hyperparameters for these baselines, tuning only the regularization coefficient per task. All the hyperparameter settings are provided in the Appendix E.

Finally, we include results from the unconstrained backbone algorithms (SAC and PPO), as well as the Human Demonstrator, to provide a comprehensive comparison.

Evaluation protocol. For each task, every method is trained with the same set of 40 human-generated demonstration trajectories. After training, we run 40 evaluation episodes per method. We measure episodic reward (higher is better) and safety cost (lower is better) on the 4 Safety-Gymnasium tasks. We repeat this with 3 independent random seeds and report mean and standard deviation (in the form mean ± std) over seeds, following recommended reporting practices for reliability.

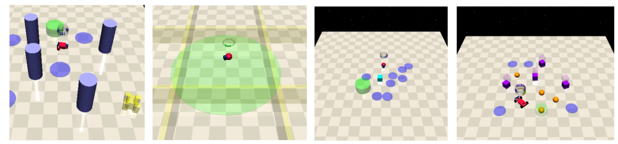

Figure 1: Snapshots of: (left) SafetyCarPush2-v0, (middle-left) SafetyPointCircle2-v0, (middle-right) SafetyPointGoal1-v0, (right) SafetyCarButton1-v0.

5.2 Experimental results

Tables 1–4 and Figure 2, as well as Figures 3-5 in Appendix B summarize results. The best algorithm is decided (highlighted) based on the following process: If all the algorithms exceed the unconstrained SAC cost, the one with the minimum cost is selected. Otherwise, we highlight the algorithm with the best trade-off between cost reduction and reward performance relative to the baseline. Hyperparameters of SafeQIL and baseline algorithms have been tuned towards improving this trade-off. The detailed methodology for quantifying the trade-off is described in Appendix A along with detailed results.

SafetyPointGoal1-v0. In this navigation task, the unconstrained SAC baseline incurs a cost of 49.15 ± 2.21. Surprisingly, the inverse constraint baselines fail to improve upon this baseline, with ICRL and VICRL incurring higher costs (62.60 and 62.97, respectively). SafeQIL emerges as the best algorithm, achieving a cost of 34.22 ± 2.71, which corresponds to a 30.4% reduction in cost relative to the unconstrained SAC baseline. While SAC-GAIL also reduces the cost (to 44.80), SafeQIL's reduction is significantly deeper. Although SafeQIL's reward (5.27) is lower than the unconstrained baselines, it is the only method that improves safety margins over the baseline while maintaining positive task performance.

Table 1: SafetyPointGoal1-v0: Comparative results.

Algorithm	Reward ↑	Cost ↓
SafeQIL	**5.27 ± 1.85**	**34.22 ± 2.71**
ICRL^2	23.21 ± 1.28	62.60 ± 9.87
ICRL^7	-0.50 ± 4.93	42.80 ± 35.43
VICRL	21.87 ± 1.20	62.97 ± 10.81
SAC-GAIL	7.17 ± 1.65	44.80 ± 18.60
SAC	27.47 ± 0.21	49.15 ± 2.21
PPO^2	23.64 ± 1.64	60.99 ± 7.99
PPO^7	26.70 ± 0.11	57.17 ± 1.68
Human Demonstrator	11.39 ± 1.55	0.00 ± 0.00

SafetyPointCircle2-v0. The unconstrained SAC baseline incurs a massive cost of 392.20 ± 4.38 in this task. Both SafeQIL and VICRL

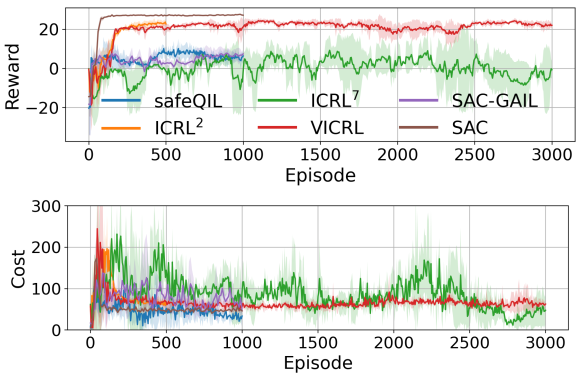

Figure 2: Learning curves for SafetyPointGoal1-v0.

successfully identify the boundary constraints, significantly outperforming ICRL[2]. VICRL achieves the best safety performance with a cost of 5.49 ± 0.81, representing a 98% reduction relative to the unconstrained baseline. SafeQIL is very close behind, achieving a cost of 29.28 ± 8.41, which corresponds to a 92% cost reduction. Crucially, SafeQIL offers a far better trade-off compared to ICRL[7] and SAC-GAIL: although these baselines achieve similar safety levels (reducing cost by 91%), they suffer severe reward degradation, retaining only 14% and 20% of the baseline reward, respectively. In contrast, SafeQIL retains 46% of the baseline reward, effectively matching VICRL's performance, outperforming the other constrained baselines by more than double in terms of task performance.

Table 2: SafetyPointCircle2-v0: Comparative results.

Algorithm	Reward ↑	Cost ↓
SafeQIL	27.06 ± 4.10	29.28 ± 8.41
ICRL[2]	28.36 ± 17.37	177.88 ± 6.70
ICRL[7]	8.21 ± 17.07	34.07 ± 16.74
VICRL	**26.29 ± 0.69**	**5.49 ± 0.81**
SAC-GAIL	11.89 ± 9.55	33.95 ± 28.02
SAC	58.81 ± 0.70	392.20 ± 4.38
PPO[2]	43.24 ± 12.01	250.40 ± 217.83
PPO[7]	32.57 ± 1.26	408.74 ± 2.76
Human Demonstrator	24.28 ± 3.98	0.00 ± 0.00

SafetyCarButton1-v0. This interaction-heavy task proves difficult for all algorithms. The unconstrained SAC baseline achieves a high reward but incurs a high cost of 299.27 ± 20.61. SafeQIL effectively bridges the gap, reducing the cost to 70.11 ± 72.96, which represents a 76% safety improvement over the baseline. While VICRL achieves an even higher cost reduction of 88% (34.65 ± 23.36), it suffers from a complete collapse in task performance (Reward = −14.12), failing to solve the task. SafeQIL avoids this collapse (Reward = −3.81), offering the most efficient trade-off by maintaining a performing policy that enforces safety.

SafetyCarPush2-v0. In this manipulation task, similarly to the Button task, the unconstrained SAC baseline incurs a high cost of 245.36 ± 28.52. A critical comparison arises between SafeQIL and SAC-GAIL. SAC-GAIL achieves the most efficient trade-off, reducing the cost by 64% (87.97 ± 10.57) while maintaining a positive reward of 0.76 ± 0.92. However, for applications prioritizing strict safety over task efficiency, SafeQIL demonstrates better performance, lowering the cost further to 57.20 ± 30.02 (a 76% reduction). Although this comes at the expense of a slightly negative reward (−0.62), SafeQIL reduces the safety violations by 30 more points compared to SAC-GAIL, providing a much tighter safety bound. In contrast, VICRL matches SafeQIL's cost (55.80) but suffers a catastrophic reward drop to −9.05, effectively failing the task. Finally, ICRL[2] and ICRL[7] fail to provide meaningful safety, reducing the cost only by 27% (178.81 ± 60.69) and 17% (203.10 ± 37.24), respectively.

Table 3: SafetyCarButton1-v0: Comparative results.

Algorithm	Reward ↑	Cost ↓
SafeQIL	**-3.81 ± 3.05**	**70.11 ± 72.96**
ICRL[2]	0.31 ± 2.23	222.01 ± 173.61
ICRL[7]	-15.83 ± 12.98	145.94 ± 23.54
VICRL	-14.12 ± 10.83	34.65 ± 23.36
SAC-GAIL	0.14 ± 4.72	98.15 ± 81.54
SAC	19.49 ± 3.50	299.27 ± 20.61
PPO[2]	11.70 ± 1.90	460.47 ± 6.61
PPO[7]	11.71 ± 1.16	431.77 ± 40.23
Human Demonstrator	14.11 ± 5.67	0.00 ± 0.00

Table 4: SafetyCarPush2-v0: Comparative results.

Algorithm	Reward ↑	Cost ↓
SafeQIL	-0.62 ± 0.65	57.20 ± 30.02
ICRL[2]	0.00 ± 1.73	203.10 ± 37.24
ICRL[7]	1.15 ± 0.49	178.81 ± 60.69
VICRL	-9.05 ± 8.58	55.80 ± 71.45
SAC-GAIL	**0.76 ± 0.92**	**87.97 ± 10.57**
SAC	1.50 ± 0.01	245.36 ± 28.52
PPO[2]	0.44 ± 0.52	316.50 ± 52.84
PPO[7]	0.72 ± 0.64	238.63 ± 23.33
Human Demonstrator	7.46 ± 1.77	0.00 ± 0.00

Overall, SafeQIL consistently lowers safety costs relative to the unconstrained baseline, with reductions ranging from 30% to 92%. It outperforms ICRL in safety across all tasks, and provides a more stable reward-cost trade-off compared to VICRL, which tends to over-constrain the policy to the point of task failure in complex manipulation environments. Finally, compared to SAC-GAIL, SafeQIL demonstrates a more robust safety behavior. While SAC-GAIL offers a competitive baseline in terms of reward, SafeQIL consistently achieves significantly tighter worst-case safety bounds when taking into account also the analysis in Appendix A.

5.3 Ablation study

To validate the contribution of SafeQIL's components, we conducted an ablation study on the challenging SafetyPointGoal1-v0 benchmark. Table 5 summarizes the results for the key variations. The original formulation achieves the most effective balance of reward (5.27 ± 1.85) and cost (34.22 ± 2.71). We found that decoupling the upper-bound selection from state similarity ("w/o cosine similarity") results in highly unstable costs (30.25 ± 19.10) and reduced reward, confirming that state-specific bounds are crucial for consistent learning. Similarly, removing the explicit upper-bound constraint ("w/o constraint term") leads to high cost variance (±14.03), indicating that relying solely on demonstration bias is insufficient to guarantee consistent safety. For further analysis, Appendix C details the full ablation of SafeQIL's components, and Appendix D evaluates the impact of demonstration dataset size.

Table 5: Ablation results for key SafeQIL's variations.

Algorithm	Reward ↑	Cost ↓
Original	**5.27 ± 1.85**	**34.22 ± 2.71**
w/o cosine similarity	3.74 ± 2.52	30.25 ± 19.10
w/o constraint term	4.34 ± 4.41	29.02 ± 14.03

6 RELATED WORK

This section reviews works that connect most closely to SafeQIL: (i) learning constraints from demonstrations, and (ii) offline RL approaches that enforce pessimism on out-of-distribution (OOD) actions.

Learning constraints from demonstrations. Safety and constraints' abiding behavior is a long-term topic of interest in the RL realm. It was formulated at first by [1] through CMDPs. More recently, the family of Inverse Constraint RL (ICRL) algorithms has shown promising results on learning to respect constraints in settings with an unknown cost function and without cost labels provided. ICRL [16] aims to approximate the cost function and employs a binary classifier to differentiate between the demonstrated trajectories and those generated by the policy under learning. A later work formulates constraint inference variationally, introducing a benchmark and a VI-based estimator referred to as VICRL [11]. Confidence-aware variants of ICRL [20] explicitly encode a desired confidence level over inferred constraints, seeking estimates that are at least as strict as ground truth with high probability — aiming to address the conservatism/coverage trade-off inherent in constraint inference. Continuing, Multi-Modal ICRL (MMICRL) [17] relaxes the single-expert assumption by modeling demonstrations as mixtures from heterogeneous experts who may adhere to distinct constraint sets. It infers multiple constraint modes and shows improved recovery and control performance in both discrete and continuous domains. Uncertainty-Aware ICRL (UAICRL) [22] tackles the ambiguity inherent in constraint inference by modeling both aleatoric and epistemic uncertainty, yielding risk-sensitive constraints via distributional Bellman updates and demonstrating robustness when demonstrations are limited or stochastic. Finally, CoCoRL [10] bypasses reward labels and learns a convex safe set directly from demonstrations with unknown rewards, providing safety guarantees and transfer to new tasks. A key distinction from the ICRL family is that we do not infer an explicit constraint set or solve a constrained MDP. Instead, we regularize the value function directly: demonstrations define support, a discriminator gates in/out-of-distribution weighting, and a state-level upper bound limits over-optimism off-support while preserving SAC-style improvement on-demonstrations-support. This sidesteps the ambiguity and calibration issues inherent in constraint identification while still leveraging demonstrations to shape safe behavior.

Offline RL with pessimistic value learning. Offline RL addresses distribution shift by making critics conservative off-demonstrations-support. CQL [9] lowers Q-values on OOD actions to obtain lower-bound critics and policy-improvement guarantees. BRAC [21] constrains the policy toward the behavior policy, while IQL [8] uses expectiles to avoid querying OOD actions. Finally, model-based methods, such as MOPO [24], MOReL [5], and COMBO [23], inject pessimism through dynamics or value regularization. Aligned with this conservative principle, our method is pessimistic off-demonstrations-support but defines “support” via demonstrations together with the online replay buffer. Importantly, the pessimism operates at the state level: we bound Q-values for out-of-distribution states using a discriminator-weighted, demonstration-informed upper bound, so every action at those states is conservatively valued, while standard SAC updates drive improvement on in-distribution states. This contrasts with action-centric penalties such as CQL's conservative regularizer and complements state-space pessimism in model-based approaches like MOReL/MOPO.

7 CONCLUSIONS

We introduced SafeQIL, a simple, model-free method for learning to respect constraints from demonstrations while continuing to improve online. The key idea is to make the critic pessimistic at out-of-distribution states by enforcing a local, demonstration-limited upper bound on $Q(s, \cdot)$ and by gating updates with a discriminator that estimates whether a state lies on the demonstration support. On in-distribution states, the policy still learns with a standard max-entropy objective (SAC), preserving the sample-efficiency and stability benefits of off-policy actor–critic training.

SafeQIL consistently reduces safety-violation costs compared to explicit constraint-inference baselines (ICRL/VICRL) in 4 Safety-Gymnasium tasks, while maintaining competitive performance on navigation tasks. The method trades some reward for safety on interaction-heavy tasks, an expected outcome of our state-level pessimism design. These results support our central claim: shaping values to be conservative only where the data indicate low support yields strong safety without discarding the performance advantages of max-entropy RL.

Limitations include reliance on the coverage and quality of demonstrations, possible miscalibration of the discriminator in hard out-of-distribution regions, and the simplicity of the closest state retrieval for the anchor. These can lead to over- or under-conservatism in poorly covered parts of the state space.

Future work aims at (i) learning task-aware state embeddings for robust closest-demo retrieval, and (ii) exploring model-based rollouts for safer recovery planning.

ACKNOWLEDGMENTS

This research work was supported by the Hellenic Foundation for Research and Innovation (HFRI) under the 5th Call for HFRI PhD Fellowships (Fellowship Number: 20769).

REFERENCES

[1] Eitan Altman. 1999. Constrained Markov Decision Processes. https://api.semanticscholar.org/CorpusID:14906227

[2] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. 2017. Improved training of wasserstein gans. *Advances in neural information processing systems* 30 (2017).

[3] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*. Pmlr, 1861–1870.

[4] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. 2018. Deep q-learning from demonstrations. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[5] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. 2020. MOReL: Model-Based Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 21810–21823. https://proceedings.neurips.cc/paper_files/paper/2020/file/f7efa4f864ae9b88d43527f4b14f750f-Paper.pdf

[6] Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. 2018. Discriminator-Actor-Critic: Addressing Sample Inefficiency and Reward Bias in Adversarial Imitation Learning. arXiv:1809.02925 [cs.LG] https://arxiv.org/abs/1809.02925

[7] Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. 2019. Discriminator-Actor-Critic: Addressing Sample Inefficiency and Reward Bias in Adversarial Imitation Learning. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Hk4fpoA5Km

[8] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. 2021. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169* (2021).

[9] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems* 33 (2020), 1179–1191.

[10] David Lindner, Xin Chen, Sebastian Tschiatschek, Katja Hofmann, and Andreas Krause. 2024. Learning Safety Constraints from Demonstrations with Unknown Rewards. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*.

[11] Guiliang Liu, Yudong Luo, Ashish Gaurav, Kasra Rezaee, and Pascal Poupart. 2023. Benchmarking Constraint Inference in Inverse Reinforcement Learning. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=vINj_Hv9szL

[12] Guiliang Liu, Yudong Luo, Ashish Gaurav, Kasra Rezaee, and Pascal Poupart. 2023. ICRL-benchmarks-public: Benchmarking Constraint Inference in Inverse Reinforcement Learning. https://github.com/Guiliang/ICRL-benchmarks-public.

[13] Guiliang Liu, Sheng Xu, Shicheng Liu, Ashish Gaurav, Sriram Ganapathi Subramanian, and Pascal Poupart. 2025. A Comprehensive Survey on Inverse Constrained Reinforcement Learning: Definitions, Progress and Challenges. arXiv:2409.07569 [cs.LG] https://arxiv.org/abs/2409.07569

[14] Zuxin Liu, Zijian Guo, Haohong Lin, Yihang Yao, Jiacheng Zhu, Zhepeng Cen, Hanjiang Hu, Wenhao Yu, Tingnan Zhang, Jie Tan, et al. 2023. Datasets and benchmarks for offline safe reinforcement learning. *arXiv preprint arXiv:2306.09303* (2023).

[15] Shehryar Malik. 2021. ICRL: Inverse Constrained Reinforcement Learning. https://github.com/shehryar-malik/icrl.

[16] Shehryar Malik, Usman Anwar, Alireza Aghasi, and Ali Ahmed. 2021. Inverse Constrained Reinforcement Learning. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 7390–7399. https://proceedings.mlr.press/v139/malik21a.html

[17] Guanren Qiao, Guiliang Liu, Pascal Poupart, and Zhiqiang Xu. 2023. Multi-Modal Inverse Constrained Reinforcement Learning from a Mixture of Demonstrations. In *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 60384–60396. https://proceedings.neurips.cc/paper_files/paper/2023/file/bdc48324d6158a7edef88d673855a3f4-Paper-Conference.pdf

[18] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. 2021. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of machine learning research* 22, 268 (2021), 1–8.

[19] Dexter R. R. Scobee and S. Shankar Sastry. 2020. Maximum Likelihood Constraint Inference for Inverse Reinforcement Learning. arXiv:1909.05477 [cs.LG] https://arxiv.org/abs/1909.05477

[20] Sriram Ganapathi Subramanian, Guiliang Liu, Mohammed Elmahgiubi, Kasra Rezaee, and Pascal Poupart. 2024. Confidence Aware Inverse Constrained Reinforcement Learning. arXiv:2406.16782 [cs.LG] https://arxiv.org/abs/2406.16782

[21] Yifan Wu, George Tucker, and Ofir Nachum. 2020. Behavior Regularized Offline Reinforcement Learning. https://openreview.net/forum?id=BJg9hTNKPH

[22] Sheng Xu and Guiliang Liu. 2024. Uncertainty-aware Constraint Inference in Inverse Constrained Reinforcement Learning. In *The Twelfth International Conference on Learning Representations*. https://openreview.net/forum?id=ILYjDvUM6U

[23] Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. 2021. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems* 34 (2021), 28954–28967.

[24] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. 2020. MOPO: Model-based Offline Policy Optimization. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 14129–14142. https://proceedings.neurips.cc/paper_files/paper/2020/file/a322852ce0df73e204b7e67cbbef0d0a-Paper.pdf