IQ-Learn: Inverse soft-Q Learning for Imitation

Anonymous Author(s)
Affiliation
Address
email

Abstract

In many sequential decision-making problems (e.g., robotics control, game playing, sequential prediction), human or expert data is available containing useful information about the task. However, imitation learning (IL) from a small amount of expert data can be challenging in high-dimensional environments with complex dynamics. Behavioral cloning is a simple method that is widely used due to its simplicity of implementation and stable convergence but doesn't utilize any information involving the environment's dynamics. Many existing methods that exploit dynamics information are difficult to train in practice due to an adversarial optimization process over reward and policy approximators or biased, high variance gradient estimators. We introduce a method for dynamics-aware IL which avoids adversarial training by learning a single Q-function, implicitly representing both reward and policy. On standard benchmarks, the implicitly learned rewards show a high positive correlation with the ground-truth rewards, illustrating our method can also be used for inverse reinforcement learning (IRL). Our method, Inverse soft-Q learning (IQ-Learn) obtains state-of-the-art results in offline and online imitation learning settings, surpassing existing methods both in the number of required environment interactions and scalability in high-dimensional spaces.

1 Introduction

Imitation of an expert has long been recognized as a powerful approach for sequential decision-making [21, 1], with applications as diverse as healthcare [27], autonomous driving [28], and playing complex strategic games [6]. In the imitation learning (IL) setting, we are given a set of expert trajectories, with the goal of learning a policy which induces behavior similar to the expert's. The learner has no access to the reward, and no explicit knowledge of the dynamics.

The simple behavioural cloning [24] approach simply maximizes the probability of the expert's actions under the learned policy, approaching the IL problem as a supervised learning problem. While this can work well in simple environments and with large quantities of data, it ignores the sequential nature of the decision-making problem, and small errors can quickly compound when the learned policy departs from the states observed under the expert. A natural way of introducing the environment dynamics is by framing the IL problem as an Inverse RL (IRL) problem, aiming to learn a reward function under which the expert's trajectory is optimal, and from which the learned imitation policy can be trained [1]. This framing has inspired several approaches which use rewards either explicitly or implicitly to incorporate dynamics while learning an imitation policy [13, 7, 23, 18]. However, these dynamics-aware methods are typically hard to put into practice due to unstable learning which can be sensitive to hyperparameter choice or minor implementation details [17].

In this work, we introduce a dynamics-aware imitation learning method which has stable, non-adversarial training, allowing us to achieve state-of-the-art performance on imitation learning benchmarks. Our key insight is that much of the difficulty with previous IL methods arises from the IRL-motivated representation of the IL problem as a min-max problem over reward and policy [13, 1].

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

Table 1: A comparison of various algorithms for imitation learning. "Convergence Guarantees" refers to if a proof is given that the algorithm converges to the correct policy with sufficient data. We consider an algorithm "directly optimized" if it consists of an optimization algorithm (such as gradient descent) applied to the parameters of a single function

	Method	Reference	Dynamics Aware	Non-Adversarial Training	Convergence Guarantees	Non-restrictive Reward	Direct Optimization
Online	Max Margin IRL	[21, 1]	✓	✓	✓	×	×
	Max Entropy IRL	[30]	✓	✓	✓	×	×
	GAIL/AIRL	[13, 7]	✓	×	✓	✓	×
	ASAF	[3]	✓	✓	✓	×	✓
	SQIL	[23]	✓	✓	×	×	✓
	Ours (Online)	–	✓	✓	✓	✓	✓
Offline	Max Margin IRL	[20, 16]	✓	✓	✓	×	×
	Max Likelihood IRL	[14]	✓	✓	✓	×	×
	Max Entropy IRL	[12]	✓	✓	✓	×	×
	ValueDICE	[18]	✓	×	×	×	×
	Behavioral Cloning	[24]	×	✓	✓	×	✓
	Regularized BC	[22]	✓	✓	✓	×	✓
	EDM	[15]	✓	✓	×	✓	✓
	Ours (Offline)	–	✓	✓	✓	✓	✓

This introduces a requirement to separately model the reward and policy, and train these two functions jointly, often in an adversarial fashion. Drawing on connections between RL and energy-based models [9, 10], we propose learning a *single model for the Q-value*. The Q-value then implicitly defines both a reward and policy function. This turns a difficult min-max problem over policy and reward functions into a simpler minimization problem over a single function, the Q-value. Since our problem has a one-to-one correspondence with the min-max problem studied in adversarial IL [13], we maintain the generality and guarantees of these previous approaches, resulting in a meaningful reward that may be used for inverse reinforcement learning. Furthermore, our method may be used to minimize a variety of statistical divergences between the expert and learned policy. We show that we recover several previously-described approaches as special cases of particular divergences, such as the regularized behavioural cloning of [22], and the conservative Q-learning of [19].

In our experiments, we find that our method is performant even with very sparse data - surpassing prior methods using *one expert demonstration* in the completely offline setting - and can scale to complex image-based tasks like Atari reaching expert performance. Moreover, our learnt rewards are highly predictive of the original environment rewards.

Concretely, our contributions are as follows:

- We present a modified Q-learning update rule for imitation learning that can be implemented on top of soft-Q learning or soft actor-critic (SAC) algorithms in fewer than **15** lines of code.
- We introduce a simple framework to minimize a wide range of statistical distances: Integral Probability Metrics (IPMs) and f-divergences, between the expert and learned distributions.
- We empirically show state-of-art results in a variety of imitation learning settings: online and offline IL. On the complex Atari suite, we outperform prior methods by **3-7x** while requiring **3x** less environment steps.
- We characterize our learnt rewards and show a high positive correlation with the ground-truth rewards, justifying the use of our method for Inverse Reinforcement Learning.

2 Background

Preliminaries We consider environments represented as a Markov decision process (MDP), which is defined by a tuple $(\mathcal{S}, \mathcal{A}, p_0, \mathcal{P}, r, \gamma)$. $\mathcal{S}, \mathcal{A}$ represent state and action spaces, p_0 and $\mathcal{P}(s'|s,a)$ represent the initial state distribution and the dynamics, $r(s,a)$ represents the reward function, and $\gamma \in (0,1)$ represents the discount factor. $\mathbb{R}^{\mathcal{S}\times\mathcal{A}} = \{x : \mathcal{S}\times\mathcal{A} \to \mathbb{R}\}$ will denote the set of all functions in the state-action space and $\overline{\mathbb{R}}$ will denote the extended real numbers $\mathbb{R} \cup \{\infty\}$. Section 3 and 4 will work with finite state and action spaces S and A, but our algorithms and experiments

later in the paper use continuous environments. Π is the set of all stationary stochastic policies that take actions in A given states in S. We work in the γ-discounted infinite horizon setting, and we will use an expectation with respect to a policy $\pi \in \Pi$ to denote an expectation with respect to the trajectory it generates: $\mathbb{E}_\pi[r(s,a)] \triangleq \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$, where $s_0 \sim p_0$, $a_t \sim \pi(\cdot|s_t)$, and $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$ for $t \geq 0$. For a policy $\pi \in \Pi$, we define its occupancy measure $\rho_\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as $\rho_\pi(s,a) = \pi(a|s) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)$. We refer to the expert policy as π_E and its occupancy measure as ρ_E. In practice, π_E is unknown and we have access to a sampled dataset of demonstrations. For brevity, we refer to ρ_π as ρ for a learnt policy in the paper.

Soft Q-functions For a reward $r \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ and $\pi \in \Pi$, the soft Bellman operator $\mathcal{B}^\pi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ defined as $(\mathcal{B}^\pi Q)(s,a) = r(s,a) + \gamma \mathbb{E}_{s'\sim P(s,a)} V^\pi(s')$ with $V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[Q(s,a) - \log \pi(a|s)\right]$. The soft Bellman operator is contractive [9] and defines a unique soft Q-function for r, given as $Q = \mathcal{B}^\pi Q$.

Max Entropy Reinforcement Learning For a given reward function $r \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$, maximum entropy RL [10, 4] aims to learn a policy that maximizes the expected cumulative discounted reward along with the entropy in each state: $\max_{\pi\in\Pi} \mathbb{E}_\pi[r(s,a)] + H(\pi)$. Where $H(\pi) \triangleq \mathbb{E}_\pi[-\log \pi(a|s)]$ is the discounted causal entropy of the policy π. The optimal policy satisfies [29, 4]:

$$\pi^*(a|s) = \frac{1}{Z_s} \exp\left(Q(s,a)\right), \tag{1}$$

where Q is the soft Q-function and Z_s is the normalization factor given as $\sum_{a'} \exp\left(Q\left(s,a'\right)\right)$.

Q satisfies the soft-Bellman equation:

$$Q(s,a) = r(s,a) + \gamma \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}\left[\log \sum_{a'} \exp(Q(s',a'))\right] \tag{2}$$

In continuous action spaces, Z_s becomes intractable and soft actor-critic methods like SAC [9] can be used to learn an explicit policy.

Max Entropy Inverse Reinforcement Learning Given demonstrations sampled using the policy π_E, maximum entropy Inverse RL aims to recover the reward function in a family of functions $\mathcal{R}$ that rationalizes the expert behavior by solving the optimization problem: $\max_{r\in\mathcal{R}} \min_{\pi\in\Pi} \mathbb{E}_{\pi_E}[r(s,a)] - (\mathbb{E}_\pi[r(s,a)] + H(\pi))$, where the expected reward of π_E is empirically approximated. It looks for a reward function that assigns high reward to the expert policy and a low reward to other policies, while searching for the best policy for the reward function in an inner loop.

The Inverse RL objective can be reformulated in terms of its occupancy measure, and with a convex reward regularizer $\psi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \overline{\mathbb{R}}$ [13]

$$\max_{r\in\mathcal{R}} \min_{\pi\in\Pi} L(\pi, r) = \mathbb{E}_{\rho_E}[r(s,a)] - \mathbb{E}_\rho[r(s,a)] - H(\pi) - \psi(r) \tag{3}$$

In general, we can exchange the max-min resulting in an objective that minimizes the statistical distance parameterized by ψ, between the expert and the policy [13]

$$\min_{\pi\in\Pi} \max_{r\in\mathcal{R}} L(\pi, r) = \min_{\pi\in\Pi} d_\psi(\rho, \rho_E) - H(\pi), \tag{4}$$

with $d_\psi \triangleq \psi^*(\rho_E - \rho)$, where ψ^* is the convex conjugate of ψ.

3 Inverse soft Q-learning (IQ-Learn) Framework

A naive solution to the IRL problem in (Eq. 3) involves (1) an outer loop learning rewards and (2) executing RL in an inner loop to find an optimal policy for them. However, we know that this optimal policy can be obtained analytically in terms of soft Q-functions (Eq. 1). Interestingly, as we will show later, the rewards can also be represented in terms of Q (Eq. 2). Together, these observations suggest it might be possible to directly solve the IRL problem by optimizing only over the Q-function.

To motivate the search of an imitation learning algorithm that depends only on the Q-function, we characterize the space of Q-functions and policies obtained using Inverse RL. We will study $\pi \in \Pi$,

$r \in \mathcal{R}$ and Q-functions $Q \in \Omega$ where $\mathcal{R} = \Omega = \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$. We assume Π is convex, compact and that $\pi_E \in \Pi$[1]. We define $V^\pi(s) = \mathbb{E}_{a\sim\pi(\cdot|s)}\left[Q(s,a) - \log \pi(a|s)\right]$.

We start with analysis developed in [13]: The regularized IRL objective $L(\pi, r)$ given by Eq. 3, is concave in the policy and convex in rewards. And has a unique saddle point where it is optimized.

To characterize the Q-functions it is useful to transform the optimization problem over rewards to a problem over Q-functions. We can get a one-to-one correspondence between r and Q:

Define the inverse soft bellman operator $\mathcal{T}^\pi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ such that

$$(\mathcal{T}^\pi Q)(s,a) = Q(s,a) - \gamma \mathbb{E}_{s'\sim P(s,a)} V^\pi(s'),$$

Lemma 3.1. *The inverse soft bellman operator $\mathcal{T}^\pi$ is bijective, and $(\mathcal{T}^\pi)^{-1} = \mathcal{B}^\pi$.*

The proof of this lemma is in Appendix A.1. For a policy π, we are thus justified in changing between rewards and their corresponding soft-Q functions. We can freely transform functions from the reward-policy space: $\Pi \times \mathcal{R}$ to the Q-policy space: $\Pi \times \Omega$, giving us the lemma:

Lemma 3.2. *If $L(\pi, r) = \mathbb{E}_{\rho_E}[r(s,a)] - \mathbb{E}_\rho[r(s,a)] - H(\pi) - \psi(r)$ and $\mathcal{J}(\pi, Q) = \mathbb{E}_{\rho_E}[(\mathcal{T}^\pi Q)(s,a)] - \mathbb{E}_\rho[(\mathcal{T}^\pi Q)(s,a)] - H(\pi) - \psi(\mathcal{T}^\pi Q)$, then for all policies $\pi \in \Pi$, $L(\pi, r) = \mathcal{J}(\pi, (\mathcal{T}^\pi)^{-1} r)$ for all $r \in \mathcal{R}$, and $\mathcal{J}(\pi, Q) = L(\pi, \mathcal{T}^\pi Q)$, for all $Q \in \Omega$.*

Lemma 3.1 and 3.2 allow us to adapt the Inverse RL objective $L(\pi, r)$ to learning Q through $\mathcal{J}(\pi, Q)$.

Simplifying our new objective (using Lemma A.3 in Appendix):

$$\mathcal{J}(\pi, Q) = \mathbb{E}_{s,a\sim\rho_E}[Q - \gamma \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)} V^\pi(s')] - (1-\gamma)\mathbb{E}_{s_0\sim p_0}[V^\pi(s_0)] - \psi(\mathcal{T}^\pi Q), \quad (5)$$

We are now ready to study $\mathcal{J}(\pi, Q)$, the Inverse RL optimization problem in the Q-policy space. As the regularizer ψ depends on both Q and π, a general analysis over all functions in $\mathbb{R}^{\mathcal{S}\times\mathcal{A}}$ becomes too difficult. We restrict ourselves to regularizers induced by a convex function $g : \mathbb{R} \to \overline{\mathbb{R}}$ such that

$$\psi_g(r) = \mathbb{E}_{\rho_E}[g(r(s,a))] \quad (6)$$

This allows us to simplify our analysis to the set of all real functions while retaining generality[2]. We further motivate this choice in Section 4.

Proposition 3.3. *In the Q-policy space, there exists a unique saddle point (π^*, Q^*) that optimizes $\mathcal{J}$. i.e. $Q^* = \operatorname{argmax}_{Q\in\Omega} \min_{\pi\in\Pi} \mathcal{J}(\pi, Q)$ and $\pi^* = \operatorname{argmin}_{\pi\in\Pi} \max_{Q\in\Omega} \mathcal{J}(\pi, Q)$. Furthermore, π^* and $r^* = \mathcal{T}^{\pi^*} Q^*$ are the solution to the Inverse RL objective $L(\pi, r)$.*

Thus we have, $\max_{Q\in\Omega} \min_{\pi\in\Pi} \mathcal{J}(\pi, Q) = \max_{r\in\mathcal{R}} \min_{\pi\in\Pi} L(\pi, r)$.

This tells us, even after transforming to Q-functions we have retained the saddle point property of the original IRL objective and optimizing $\mathcal{J}(\pi, Q)$ recovers this saddle point. In the Q-policy space, we can get an additional property:

Proposition 3.4. *For a fixed Q, $\operatorname{argmin}_{\pi\in\Pi} \mathcal{J}(\pi, Q)$ is the solution to max entropy RL with rewards $r = \mathcal{T}^\pi Q$. Thus, this forms a manifold in the Q-policy space, that satisfies*

$$\pi_Q(a|s) = \frac{1}{Z_s} \exp(Q(s,a)),$$

with normalization factor $Z_s = \sum_a \exp Q(s,a)$ and π_Q defined as the π corresponding to Q.

Proposition 3.3 and 3.4 are telling us that if we know Q, then the inner optimization problem in terms of policy is trivial, and obtained in a closed form! Thus, we can recover an objective that only requires learning Q:

$$\max_{Q\in\Omega} \min_{\pi\in\Pi} \mathcal{J}(\pi, Q) = \max_{Q\in\Omega} \mathcal{J}\left(\pi_Q, Q\right) \quad (7)$$

Furthermore, we have:

Proposition 3.5. *Let $\mathcal{J}^*(Q) = \mathcal{J}\left(\pi_Q, Q\right)$. Then $\mathcal{J}^*$ is concave in Q.*

Thus, this new optimization objective is well-behaved and is maximized only at the saddle point.

[1] The full policy class satisfies all these assumptions

[2] Averaging over the expert occupancy allows the regularizer to adjust to arbitrary experts

In Appendix C, we expand on our analysis and characterize the behavior for different choices of regularizer ψ, while giving proofs of all our propositions. Figure 1 summarizes the properties for the IRL objective: there exists a optimal policy manifold depending on Q, allowing optimization along it (using $\mathcal{J}^*$) to converge to the saddle point. We further present analysis of IL methods that learn Q-functions like SQIL [23] and ValueDICE [18] and find subtle fallacies affecting their learning.

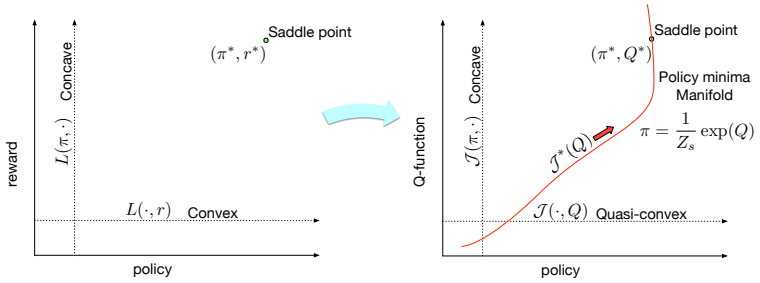

Figure 1: Properties of IRL objective in reward-policy space and Q-policy space.

Note that although the same analysis holds in the reward-policy space, the optimal policy manifold depends on Q, which isn't trivially known unlike when in the Q-policy space.

4 Approach

In this section, we develop our inverse soft-Q learning (IQ-Learn) algorithm, such that it recovers the optimal soft Q-function for a MDP from a given expert distribution. We start by learning energy-based models for the policy similar to soft Q-learning and later learn an explicit policy similar to actor-critic methods.

4.1 General Inverse RL Objective

For designing a practical algorithm using regularizers of the form ψ_g (from Eq. 6), we define g using a concave function $\phi : \mathcal{R}_\psi \to \mathbb{R}$, such that $g(x) = \begin{cases} x - \phi(x) & \text{if } x \in \mathcal{R}_\psi \\ +\infty & \text{otherwise} \end{cases}$ with the rewards constrained in R_ψ.

For this choice of ψ, the Inverse RL objective $L(\pi, r)$ takes the form of Eq. 4 with a distance measure:

$$d_\psi(\rho, \rho_E) = \max_{r \in \mathcal{R}_\psi} \mathbb{E}_{\rho_E}[\phi(r(s,a))] - \mathbb{E}_\rho[r(s,a)], \tag{8}$$

This forms a general learning objective that allows the use of a wide-range of statistical distances including Integral Probability Metrics (IPMs) and f-divergences (see Appendix B). [3]

4.2 Choice of Statistical Distances

While choosing a practical regularizer, it can be useful to obtain certain properties on the reward functions we recover. Some (natural) nice properties are: having rewards bounded in a range, learning smooth functions or enforcing a norm-penalty.

In fact, we find these properties correspond to the Total Variation distance, the Wasserstein-1 distance and the χ^2-divergence respectively. The regularizers and the induced statistical distances are summarized in Table 2:

Table 2: Enforced reward property, corresponding regularizer ψ and statistical distance ($R_{\text{max}}, K, \alpha \in \mathbb{R}^+$)

Reward Property	ψ	d_ψ
Bound range	$\psi = 0$ if $\lvert r\rvert \le R_{\text{max}}$ and $+\infty$ otherwise	$2R_{\text{max}} \cdot \text{TV}(\rho, \rho_E)$
Smoothness	$\psi = 0$ if $\lVert r\rVert_{\text{Lip}} \le K$ and $+\infty$ otherwise	$K \cdot W_1(\rho, \rho_E)$
L2 Penalization	$\psi(r) = \alpha r^2$	$\frac{1}{4\alpha} \cdot \chi^2(\rho, \rho_E)$

We find that these choice of regularizers[4] work very well in our experiments. In Appendix B, we further give a table for the well known f-divergences, the corresponding ϕ and the learnt reward estimators, along with a result ablation on using different divergences. Compared to χ^2, we find other f-divergences like Jensen-Shannon result in similar performances but are not as readily interpretable.

[3] We recover IPMs when using identity ϕ and restricted reward family $\mathcal{R}$

[4] The additional scalar terms scale the entropy regularization strength and can be ignored in practice

4.3 Inverse soft-Q update (Discrete control)

Optimization along the optimal policy manifold gives the concave objective (Prop 3.5):

$$\max_{Q \in \Omega} \mathcal{J}^*(Q) = \mathbb{E}_{\rho_E}[\phi(Q(s,a) - \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} V^*(s'))] - (1-\gamma)\mathbb{E}_{\rho_0}[V^*(s_0)], \tag{9}$$

with $V^*(s) = \log \sum_a \exp Q(s,a)$.

For each Q, we get a corresponding reward $r(s,a) = Q(s,a) - \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)}[\log \sum_{a'} \exp Q(s',a')]$. This correspondence is unique (Lemma A.1 in Appendix), and every update step can be seen as finding a better reward for IRL.

Note that estimating $V^*(s)$ exactly is only possible in discrete action spaces. Our objective forms a variant of soft-Q learning: to learn the optimal Q-function given an expert distribution.

4.4 Inverse soft actor-critic update (Continuous control)

In continuous action spaces, it might not be possible to exactly obtain the optimal policy π_Q, which forms an energy-based model of the Q-function, and we use an explicit policy π to approximate π_Q.

For any policy π, we have a objective (from Eq. 5):

$$\mathcal{J}(\pi, Q) = \mathbb{E}_{\rho_E}[\phi(Q - \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} V^\pi(s'))] - (1-\gamma)\mathbb{E}_{\rho_0}[V^\pi(s_0)] \tag{10}$$

For a fixed Q, soft actor-critic (SAC) update: $\min_\pi \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi(\cdot|s)}[Q(s,a) - \log \pi(a|s)]$, brings π closer to π_Q while always minimizing Eq. 10 (Lemma A.4 in Appendix). Here $\mathcal{D}$ is the distribution of previously sampled states, or a replay buffer.

Thus, we obtain the modified actor-critic update rule to learn Q-functions from the expert distribution:

1. For a fixed π, optimize Q by maximizing $\mathcal{J}(\pi, Q)$.
2. For a fixed Q, apply SAC update to optimize π towards π_Q.

This differs from ValueDICE [18], where the actor is updated adverserially and the objective may not always converge (Appendix C).

5 Practical Algorithm

Pseudocode in Algorithm 1, shows our Q-learning and actor-critic variants, with differences with conventional RL algorithms in red (we optimize -$\mathcal{J}$ to use gradient descent). We can implement our algorithm IQ-Learn in **15** lines of code on top of standard implementations of (soft) DQN [10] for discrete control or soft actor-critic (SAC) [9] for continuous control, with a change on the objective for the Q-function. Default hyperparameters from [10, 9] work well, except for tuning the entropy regularization. Target networks were helpful for continuous control. We elaborate details in Appendix D.

Algorithm 1 Inverse soft Q-Learning (both variants)

1: Initialize Q-function Q_θ, and optionally a policy π_ϕ
2: **for** step t in {1...N} **do**
3: Train Q-function using objective from Equation 9:
$\theta_{t+1} \leftarrow \theta_t - \alpha_Q \nabla_\theta[-\mathcal{J}(\theta)]$
(Use V^* for Q-learning and V^{π_ϕ} for actor-critic)
4: (only with actor-critic) Improve policy π_ϕ with SAC style actor update:
$\phi_{t+1} \leftarrow \phi_t - \alpha_\pi \nabla_\phi \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi_\phi(\cdot|s)}[Q(s,a) - \log \pi_\phi(a|s)]$
5: **end for**

Algorithm 2 Recover policy and reward

1: Given trained Q-function Q_θ, and optionally a trained policy π_ϕ
2: Recover policy π:
(Q-learning) $\pi := \frac{1}{Z} \exp Q_\theta$
(actor-critic) $\pi := \pi_\phi$
3: For state $\mathbf{s}$, action $\mathbf{a}$ and $\mathbf{s}' \sim \mathcal{P}(\cdot|\mathbf{s}, \mathbf{a})$
4: Recover reward $r(\mathbf{s}, \mathbf{a}, \mathbf{s}') = Q_\theta(\mathbf{s}, \mathbf{a}) - \gamma V^\pi(\mathbf{s}')$

5.1 Training methodology

Corollary 2.1 in Appendix A states $\mathbb{E}_{(s,a) \sim \mu}[V^\pi(s) - \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} V^\pi(s')] = (1-\gamma)\mathbb{E}_{s \sim p_0}[V^\pi(s)]$, where μ is any policy's occupancy. We use this to stabilize training instead of using Eq. 9 directly.

Online: Instead of directly estimating $\mathbb{E}_{p_0}[V^\pi(s_0)]$ in our algorithm, we can sample (s, a, s') from a replay buffer and get a single-sample estimate $\mathbb{E}_{(s,a,s')\sim\text{replay}}[V^\pi(s) - \gamma V^\pi(s')]$. This removes the issue where we are only optimizing Q in the inital states resulting in overfitting of $V^\pi(s_0)$, and improves the stability for convergence in our experiments. We find sampling half from the policy buffer and half from the expert distribution gives the best performances. Note that this is makes our learning online, requiring environment interactions.

Offline: Although $\mathbb{E}_{p_0}[V^\pi(s_0)]$ can be estimated offline we still observe an overfitting issue. Instead of requiring policy samples we use only expert samples to estimate $\mathbb{E}_{(s,a,s')\sim\text{expert}}[V^\pi(s) - \gamma V^\pi(s')]$ to sufficiently approximate the term. This methodology gives us state-of-art results for offline IL.

5.2 Recovering rewards

Instead of the conventional reward function $r(s, a)$ on state and action pairs, our algorithm allows recovering rewards for each transition (s, a, s') using the learnt Q-values as follows:

$$r(s, a, s') = Q(s, a) - \gamma V^\pi(s') \tag{11}$$

Now, $\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[Q(s, a) - \gamma V^\pi(s')] = Q(s, a) - \gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}[V^\pi(s')] = \mathcal{T}^\pi Q(s, a)$. This is just the reward function $r(s, a)$ we want. So by marginalizing over next-states, our expression correctly recovers the reward over state-actions. Thus, Eq. 11 gives the reward over transitions.

Our rewards require s' which can be sampled from the environment, or by using a dynamics model.

5.3 Implementation of Statistical Distances

Implementing TV and W_1 distances is fairly trivial and we give details in Appendix B. For the χ^2**-divergence**, we note that it corresponds to $\phi(x) = x - \frac{1}{4\alpha}x^2$. On substituting in Eq. 9, we get

$$\max_{Q\in\Omega} \mathbb{E}_{\rho_E}[(Q(s, a) - \gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}V^*(s'))] - (1 - \gamma)\mathbb{E}_{p_0}[V^*(s_0)] - \tfrac{1}{4\alpha}\mathbb{E}_{\rho_E}[(Q(s, a) - \gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}V^*(s'))^2]$$

In a fully offline setting, this can be further simplified as (using the offline methodology in Sec 5.1):

$$\min_{Q\in\Omega} \; -\mathbb{E}_{\rho_E}[(Q(s, a) - V^*(s))] + \frac{1}{4\alpha}\mathbb{E}_{\rho_E}[(Q(s, a) - \gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}V^*(s'))^2] \tag{12}$$

This is interestingly the same as the Q-learning objective in CQL [19], an state-of-art method for offline RL (using 0 rewards), and shares similarities with regularized behavior cloning [23] [5].

5.4 Learning state-only reward functions

Previous works like AIRL [7] propose learning rewards that are only function of the state, and claim that these form of reward functions generalize between different MDPs. We find our method can predict state-only rewards by using the policy and expert state-marginals with a modification to Eq. 9:

$$\max_{Q\in\Omega} \mathcal{J}^*(Q) = \mathbb{E}_{s\sim\rho_E(s)}[\mathbb{E}_{a\sim\pi(\cdot|s)}[\phi(Q(s, a) - \gamma\mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a)}V^*(s'))]] - (1 - \gamma)\mathbb{E}_{p_0}[V^*(s_0)]$$

Interestingly, our objective no longer depends on the the expert actions π_E and can be used for IL using only observations. For the sake of brevity, we expand on this in Section 1 in Appendix A.

6 Related Work

Classical IL: Imitation learning has a long history, with early works using supervised learning to match a policy's actions to those of the expert [11, 25]. A significant advance was made with the formulation of IL as the composition of RL and IRL [21, 1, 30], recovering the expert's policy by inferring the expert's reward function, then finding the policy which maximizes reward under this reward function. These early approaches required a hand-designed featurization of the MDP, limiting their applicability to complex MDPs.

[5] The simplification to get Eq. (12) is not applicable in the online IL setting where our method differs

Online IL: More recent work aims to leverage the power of modern machine learning approaches to learn good featurizations and extend IL to complex settings. Recent work generally falls into one of two settings: online or offline. In the online setting, the IL algorithm is able to interact with the environment to obtain dynamics information. GAIL [13] takes the nested RL/IRL formulation of earlier work , optimizing over all reward functions with a convex regularizer. This results in the objective in Eq. (3), with a max-min adversarial problem similar to a GAN [8]. A variety of further work has built on this adversarial approach [17, 7, 2]. A separate line of work aims to simplify the problem in Eq. (3) by using a fixed r or π. In SQIL [23], r is chosen to be the 1-0 indicator on the expert demonstrations, while ASAF [3] takes the GAN approach and uses a discriminator (with role similar to r) of fixed form, consisting of a ratio of expert and learner densities.

Offline IL: In the offline setting, the learner has no access to the environment. The simple behavioural cloning (BC) [24] approach is offline, but doesn't use any dynamics information. ValueDICE [18] is a dynamics-aware offline approach with an objective somewhat similar to ours, motivated from minimization of a variational representation of the KL-divergence between expert and learner policies. ValueDICE requires adversarial optimization to learn the policy and Q-functions, with a biased gradient estimator for training. We show a way to recover a unbiased gradient estimate for the KL-divergence in Appendix C. The EDM method [15] incorporates dynamics via learning an explicit energy based model for the expert state occupancy, although some theoretical details have been called into question (see [26], appendix D). Finally, the very recent AVRIL approach [5] uses a variational method to solve a probabilistic formulation of IL, finding a posterior distribution over r and π.

7 Experiments

7.1 Experimental Setup

We compare IQ-Learn ("IQ") to prior work on a diverse collection of RL tasks and environments - ranging from low-dimensional control tasks: CartPole, Acrobot, LunarLander - to more challenging continuous control MuJoCo tasks: HalfCheetah, Hopper, Walker and Ant. Furthermore, we test on the visually challenging Atari Suite with high-dimensional image inputs. We compare on offline IL - with no access to the the environment while training, and online IL - with environment access. We show results on W_1 and χ^2 as our statistical distances, as we found them more effective than TV distance. In all cases, we train until convergence and average over multiple seeds. Hyperparameter settings and training details are detailed in Appendix D.

7.2 Benchmarks

Offline IL We compare to the state-of-art IL methods EDM and AVRIL, following the same experimental setting as [5]. Furthermore, we compare with ValueDICE which also learns Q-functions, albeit with drawbacks such as adversarial optimization. We also experimented with SQIL, but found that it was not competitive in the offline setting. Finally, we utilize BC as an additional IL baseline.

Online IL We use MuJoCo and Atari environments and compare against state-of-art online IL methods: ValueDICE, SQIL and GAIL. We only show results on χ^2 as W_1 was harder to stabilize on complex environments[6]. Using target updates stabilizes the Q-learning on MuJoCo. For brevity, further online IL results are shown in the Appendix D.

7.3 Results

Offline IL We present results on the three offline control tasks in Figure 2. On all tasks, IQ strongly outperforms prior works we compare to in performance and sample efficiency. Using just *one expert trajectory*, we achieve expert performance on Acrobot and reach near expert on Cartpole.

Mujoco Control We present our results on the MuJoCo tasks using a single expert demo in Table 3. IQ achieves expert-level performance in all the tasks while outperforming prior methods like ValueDICE and GAIL. We did not find SQIL competitive in this setting, and skip it for brevity.

[6] χ^2 and W_1 can be used together to still have a convex regularization and is more stable

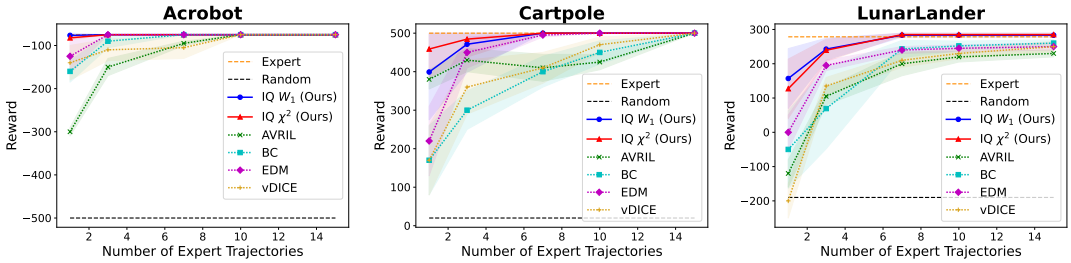

Figure 2: **Offline IL results.** We plot the average environment returns vs the number of expert trajectories.

Atari We present our results on Atari using 20 expert demos in Figure 3. We reach expert performance on Space Invaders while being near expert on Pong and Breakout. Compared to prior methods like SQIL, IQ obtains **3-7x** normalized score[7] and converges in ∼300k steps, being **3x** faster compared to Q-learning based RL methods that take more than 1M steps to converge. Other popular methods like GAIL and ValueDICE perform near random even with 1M env steps.

Table 3: **Mujoco Results.** We show our performance on MuJoCo control tasks using a single expert trajectory.

Task	GAIL	ValueDICE	IQ (Ours)	Expert
Hopper	3252.5	3312.1	**3546.4**	3532.7
Half-Cheetah	3080.0	3835.6	**5076.6**	5098.3
Walker	4013.7	3842.6	**5134.0**	5274.5
Ant	2299.1	1806.3	**4362.9**	4700.0

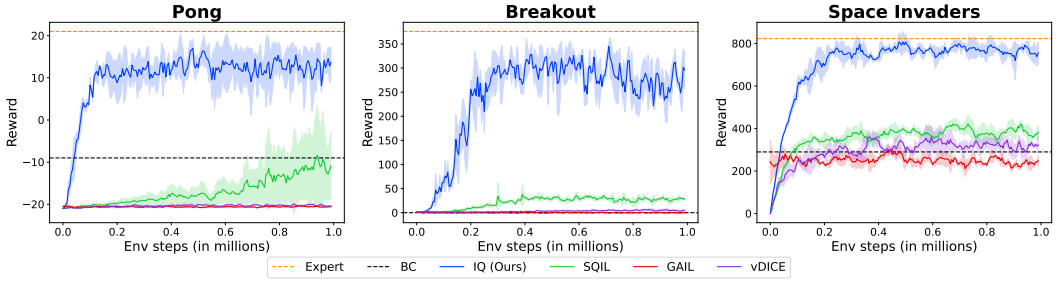

Figure 3: **Atari Results**. We show the returns vs the number of env steps. (Averaged over 5 seeds)

7.4 Recovered Rewards

IQ has the added benefit of recovering rewards and can be used for IRL. On Hopper task, our learned rewards have a Pearson correlation of **0.99** with the true rewards. In Figure 4, we visualize our recovered rewards in a simple grid environment. We elaborate details in Appendix D.

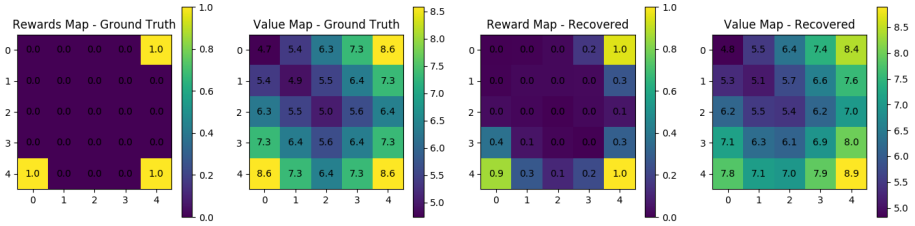

Figure 4: **Reward Visualization.** We use a discrete GridWorld environment with 5 possible actions: up, down, left, right, stay. Agent starts in a random state. (With 30 expert demos)

8 Discussion and Outlook

We present a new principled framework for learning soft-Q functions for IL and recovering the optimal policy and the reward, building on past works in IRL [30]. Our algorithm IQ-Learn outperforms prior methods with very sparse expert data and scales to complex image-based environments. We also recover rewards highly correlated with actual rewards. It has applications in autonomous driving and complex decision-making, but proper considerations need to be taken into account to ensure safety and reduce uncertainty, before any deployment. Finally, human or expert data can have errors that can propogate. A limitation of our method is that our recovered rewards depend on the environment dynamics, preventing trivial use on reward transfer settings. One direction of future work could be to learn a reward model from the trained soft-Q model to make the rewards explicit.

[7] normalizing rewards obtained from random behavior to 0 and expert to 1

References

[1] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. *International conference on Machine learning (ICML)*, 2004. 1, 2, 7

[2] Nir Baram, Oron Anschel, and Shie Mannor. Model-based adversarial imitation learning. *stat*, 1050:7, 2016. 8

[3] Paul Barde, Julien Roy, Wonseok Jeon, Joelle Pineau, Christopher Pal, and Derek Nowrouzezahrai. Adversarial soft advantage fitting: Imitation learning without policy optimization. *Advances in neural information processing systems (NeurIPS)*, 2020. 2, 8

[4] M. Bloem and N. Bambos. Infinite time horizon maximum causal entropy inverse reinforcement learning. *53rd IEEE Conference on Decision and Control*, pages 4911–4916, 2014. 3

[5] Alex J. Chan and Mihaela van der Schaar. Scalable bayesian inverse reinforcement learning, 2021. 8

[6] G Alphastar DeepMind. Mastering the real-time strategy game starcraft ii, 2019. 1

[7] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adverserial inverse reinforcement learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rkHywl-A-. 1, 2, 7, 8

[8] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. 8

[9] T. Haarnoja, Aurick Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, 2018. 2, 3, 6

[10] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. 2017. 2, 3, 6

[11] G HAYES. A robot controller using learning by imitation. In *Proc. 2nd Int. Symposium on Intelligent Robotic Systems, LIFTA-IMAG, Grenoble, France*, 1994. 7

[12] Michael Herman, Tobias Gindele, Jörg Wagner, Felix Schmitt, and Wolfram Burgard. Inverse reinforcement learning with simultaneous estimation of rewards and dynamics. *International conference on artificial intelligence and statistics (AISTATS)*, 2016. 2

[13] Jonathan Ho and S. Ermon. Generative adversarial imitation learning. In *NIPS*, 2016. 1, 2, 3, 4, 8

[14] Vinamra Jain, Prashant Doshi, and Bikramjit Banerjee. Model-free irl using maximum likelihood estimation. *AAAI Conference on Artificial Intelligence (AAAI)*, 2019. 2

[15] Daniel Jarrett, Ioana Bica, and Mihaela van der Schaar. Strictly batch imitation learning by energy-based distribution matching. *Advances in neural information processing systems (NeurIPS)*, 2020. 2, 8

[16] Edouard Klein, Matthieu Geist, and Olivier Pietquin. Batch, off-policy and model-free apprenticeship learning. *European Workshop on Reinforcement Learning (EWRL)*, 2011. 2

[17] Ilya Kostrikov, Kumar Krishna Agrawal, Debidatta Dwibedi, Sergey Levine, and Jonathan Tompson. Discriminator-actor-critic: Addressing sample inefficiency and reward bias in adversarial imitation learning. In *International Conference on Learning Representations*, 2018. 1, 8

[18] Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. Imitation learning via off-policy distribution matching. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Hyg-JC4FDr. 1, 2, 5, 6, 8

[19] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. 2020. URL https://arxiv.org/abs/2006.04779. 2, 7

[20] Donghun Lee, Srivatsan Srinivasan, and Finale Doshi-Velez. Truly batch apprenticeship learning with deep successor features. *International Joint Conference on Artificial Intelligence (IJCAI)*, 2019. 2

[21] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. *International conference on Machine learning (ICML)*, 2000. 1, 2, 7

[22] Bilal Piot, Matthieu Geist, and Olivier Pietquin. Boosted and reward-regularized classification for apprenticeship learning. *International conference on Autonomous agents and multi-agent systems (AAMAS)*, 2014. 2

[23] Siddharth Reddy, A. Dragan, and S. Levine. Sqil: Imitation learning via reinforcement learning with sparse rewards. *arXiv: Learning*, 2020. 1, 2, 5, 7, 8

[24] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. *International conference on artificial intelligence and statistics (AISTATS)*, 2010. 1, 2, 8

[25] Claude Sammut, Scott Hurst, Dana Kedzier, and Donald Michie. Learning to fly. In *Proceedings of the Ninth Conference on Machine Learning*, pages 385–393. Elsevier, 1992. 7

[26] Gokul Swamy, Sanjiban Choudhury, Zhiwei Steven Wu, and J Andrew Bagnell. Of moments and matching: Trade-offs and treatments in imitation learning. *arXiv preprint arXiv:2103.03236*, 2021. 8

[27] Lu Wang, Wenchao Yu, Xiaofeng He, Wei Cheng, Martin Renqiang Ren, Wei Wang, Bo Zong, Haifeng Chen, and Hongyuan Zha. Adversarial cooperative imitation learning for dynamic treatment regimes. In *Proceedings of The Web Conference 2020*, pages 1785–1795, 2020. 1

[28] Jinyun Zhou, Rui Wang, Xu Liu, Yifei Jiang, Shu Jiang, Jiaming Tao, Jinghao Miao, and Shiyu Song. Exploring imitation learning for autonomous driving with feedback synthesizer and differentiable rasterization. *arXiv preprint arXiv:2103.01882*, 2021. 1

[29] Brian D Ziebart. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. 2010. 3

[30] Brian D. Ziebart, Andrew L. Maas, J. Bagnell, and A. Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008. 2, 7, 9

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 8
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 8
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendix D
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No] We refer to standard RL codebases that are under the MIT license
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We release parts of our codebase
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]