Policy Optimization with Demonstrations

Bingyi Kang [1] **Zequn Jie** [2] **Jiashi Feng** [1]

Abstract

Exploration remains a significant challenge to reinforcement learning methods, especially in environments where reward signals are sparse. Recent methods of learning from demonstrations have shown to be promising in overcoming exploration difficulties but typically require considerable high-quality demonstrations that are difficult to collect. We propose to effectively leverage available demonstrations to guide exploration through enforcing occupancy measure matching between the learned policy and current demonstrations, and develop a novel Policy Optimization from Demonstration (POfD) method. We show that POfD induces implicit dynamic reward shaping and brings provable benefits for policy improvement. Furthermore, it can be combined with policy gradient methods to produce state-of-the-art results, as demonstrated experimentally on a range of popular benchmark sparse-reward tasks, even when the demonstrations are few and imperfect.

1. Introduction

Reinforcement Learning (RL) solves sequential decision making problems based on the experiences collected by interacting with environments. Thanks to the advances in deep learning, various neural network powered RL methods have made significant progress for multiple applications, including Atari games (Mnih et al., 2015; Van Hasselt et al., 2016; Mnih et al., 2016), robot locomotion tasks (Schulman et al., 2015; 2017) and the game of Go (Silver et al., 2016; 2017). However, exploration problems, that how to gain more experience with novel strategies to improve the performance in the long run, are still challenging in deep RL methods. When the reward signals are sparse and rare, existing methods still struggle to explore effectively to learn meaningful policies. This is because most of them rely on heuristic exploration strategies, *e.g.*, ϵ-greedy for value based methods (Van Hasselt et al., 2016) and noise-based exploration for policy gradient methods (Sutton et al., 2000), which are undirected and incapable of finding interesting states explicitly.

Some recent works (Nair et al., 2017; Houthooft et al., 2016; Plappert et al., 2017; Pathak et al., 2017) have been devoted to tackling the exploration problems in RL. They are basically rooted in the following two ideas. 1) Reshape the original reward function by encouraging the agent to visit states never seen before, driven by intrinsic curiosity (Pathak et al., 2017) or information gain (Houthooft et al., 2016). 2) Use demonstration trajectories sampled from an expert policy to guide the learning procedure, by either putting the demonstrations into a replay memory (Nair et al., 2017; Hester et al., 2017; Večerík et al., 2017) or using them to pretrain the policy in a supervised manner (Silver et al., 2016). Learning from demonstrations has shown promising performance in overcoming exploration difficulties in sparse-reward environments. However, existing schemes cannot fully leverage the power of the demonstration data, limited by only treating them in the same way as self-generated data, and usually require a tremendous number of high-quality demonstrations which are difficult to collect at scale.

To address this problem, we propose to combine above two ideas by developing a principled method that rewards demonstration-like actions more during interaction with environments, which thus encourages exploration for meaningful states when feedback is sparse. The intuition is that, when the reward signal is not available, the agent should mimic the demonstrated behavior in early learning stages for exploration. After acquiring sufficient skills, the agent can explore new states on its own. This is actually a dynamic intrinsic reward mechanism that can be introduced for reshaping native rewards in RL. To this end, we propose a novel Policy Optimization from Demonstration (POfD) method, which can acquire knowledge from demonstration data to boost exploration, even though the data are scarce and imperfect. We realize our idea with three technical novelties. 1) We reformulate the policy optimization objective by adding a demonstration-guided exploration term, which measures the divergence between the current policy and the expert one, forcing expert-alike exploration. We theoretically analyze the benefits brought by POfD to vanilla

[1]Department of Electrical and Computer Engineering, National University of Singapore, Singapore [2]Tencent AI Lab, China. Correspondence to: Bingyi Kang <kang@u.nus.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

policy gradient ones, in terms of improvement over the expected return. 2) We convert the proposed objective to a new one defined on *occupancy measure* (Ho & Ermon, 2016) for better exploiting demonstrations and establish an optimization-friendly lower bound. 3) We eventually draw a connection between optimizing the derived lower bound and generative adversarial training (Goodfellow et al., 2014), and show that the optimization can thus easily proceed in a manner of alternating between two sub-procedures. One is fitting a discriminator measuring the similarity between expert data and self-generated data to reshape the reward; the other one is updating the policy with the gradient from the reshaped reward function. POfD is general and compatible with most policy gradient methods. We also show that existing replay memory based learning from demonstration methods (Hester et al., 2017; Večerík et al., 2017) can be interpreted as degenerated cases of our method int terms of how to leverage the demonstration data.

We evaluate our POfD on physical locomotion tasks based on Mujoco (Todorov et al., 2012) in sparse-reward environments. We compare POfD against 5 state-of-the-art baselines. The experiments clearly demonstrate that POfD surpasses all the well-established baselines and performs very well in these environments. Its performance is even comparable with that achieved by policy gradient methods in oracle dense-reward environments.

2. Related Work

Recently, learning from demonstration (LfD) (Schaal, 1997) has received increasing attention as a promising way to overcome exploration difficulties in RL (Subramanian et al., 2016). Most LfD methods adopt value-based RL algorithms, which are off-policy in nature. For instance, DQfD (Hester et al., 2017) introduces LfD into DQN (Mnih et al., 2015) by adding demonstration data into the replay buffer. It uses a refined priority replay mechanism (Schaul et al., 2015) and gives additional priority to the demonstration data. However, DQfD is limited to applications with discrete action spaces. DDPGfD (Večerík et al., 2017) extends LfD to continuous action domains, which is built upon DDPG (Lillicrap et al., 2015) similar to DQfD. Both DQfD and DDPGfD suffer the problem of under-exploiting demonstration data, as detailed in Sec. 5. Some existing methods leverage demonstration data in different ways. Kim et al. (2013); Piot et al. (2014) are based on policy iteration and use demonstration data to shape the value function. Moreover, they require the value of demonstrated state-action pairs to be larger than the others with a margin. Those methods would suffer performance decline when only imperfect demo data are given. Aravind S. Lakshminarayanan (2016) considered LfD under settings where the demonstration data are in paucity. They proposed a hybrid framework that learns the policy in which the probability of taking demonstrated actions is maximized. Recently, Brys et al. (2015) introduced a reward reshaping mechanism to encourage taking actions close to the demonstrated ones. They share a similar motivation with us but their method has a different course. They defined a potential function based on multi-variate Gaussian to model the distribution of state-actions. Different from our POfD, all the above methods require a significantly large amount of perfect demonstration data to achieve satisfactory performance.

Imitation learning aims at mimicking expert behaviors, which can be realized in multiple ways. Most recent successful imitation learning algorithms are based on Inverse Reinforce Learning (IRL) (Ng et al., 2000), which targets at recovering the reward function of a given task from samples, without knowing the dynamics. Following this line, Syed & Schapire (2008); Syed et al. (2008) casted IRL problems as a two-player zero-sum game which is then solved by alternating between fitting the reward function and selecting the policy. However, their capacity is limited to small-scale problems. Very recently, Generative Adversarial Imitation Learning (GAIL) (Ho & Ermon, 2016) is shown to be very effective in high-dimensional continuous control problems. Instead of fitting the underlying reward function as traditional IRL algorithms, GAIL uses a discriminator to distinguish whether a state-action pair is from the expert or the learned policy. Meanwhile, GAIL optimizes the policy to confuse the discriminator. Though effective for imitation learning, these algorithms cannot leverage the valuable feedback given by the environments and usually suffer sharp performance decline when the expert data are imperfect. Our POfD overcomes such inherent limitations by learning from feedbacks and can learn well-performing policies even though the expert data are imperfect.

A similar and contemporary idea has also been introduced in (Li et al., 2017; Zhu et al., 2018) to learn an agent with hybrid imitation learning and reinforcement learning reward. However, they only treat it as a heuristic method and provide intuitive explanations. In contrast, we consider a novel practical setting and develop the reward reshaping mechanism theoretically with monotonic improvement guarantee.

3. Background

3.1. Preliminaries

We consider the standard Markov Decision Process (MDP) (Sutton & Barto, 1998). MDP is defined by a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma \rangle$, where $\mathcal{S}$ and $\mathcal{A}$ are the state space and the action space respectively, $\mathcal{P}(s'|s,a)$ is the transition distribution of taking action a at state s, $r(s,a)$ is the reward function, and $\gamma \in (0,1)$ is the discount factor.

Given a stochastic policy $\pi(a|s) = p(a|s;\pi)$ mapping from

states to action probabilities, the performance of π is usually evaluated by its expected discounted reward $\eta(\pi)$:

$$\eta(\pi) = \mathbb{E}_\pi\left[r(s,a)\right] = \mathbb{E}_{(s_0,a_0,s_1,\dots)}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\right], \tag{1}$$

where $(s_0, a_0, s_1, \dots)$ is a trajectory generated by executing policy π, *i.e.*, $s_0 \sim p_0$, $a_t \sim \pi(\cdot|s_t)$ and $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$. Similarly, following the standard definitions, the value function V_π and action value function Q_π can be written as $V_\pi(s) = \mathbb{E}_\pi[r(\cdot,\cdot)|s_0{=}s]$ and $Q_\pi(s,a) = \mathbb{E}_\pi[r(\cdot,\cdot)|s_0{=}s, a_0{=}a]$. Subtracting $Q_\pi(s,a)$ by $V_\pi(s)$ gives the advantage function $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$ that reflects the expected additional reward that the agent will get after taking action a in state s.

Reinforcement Learning (RL) (Sutton & Barto, 1998) is a set of algorithms trying to infer a policy achieving maximal reward $\eta(\pi)$ with regard to some form of reward signals $r(s,a)$ from trajectories $\mathcal{D} = \{\tau_i\}$ generated by executing a current policy (on-policy methods) or some other policy (off-policy methods). Each trajectory consists of a sequence of state transitions $\tau = \{(s_0,a_0),(s_1,a_1),\dots,(s_T,a_T)\}$.

The occupancy measure, defined as follows, characterizes the distribution of action-state pairs when executing policy π. It will play an important role in our analysis.

Definition 1. *(Occupancy measure) Let $\rho_\pi(s) : \mathcal{S} \to \mathbb{R}$ denote the unnormalized distribution of state visitation by following policy π in the environment:*

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

Then the unnormalized distribution of state-action pairs $\rho_\pi(s,a) = \rho_\pi(s)\pi(a|s)$ is called occupancy measure *of policy π.*

Substituting $\rho_\pi(s,a)$ into Eqn. (1), we have the following equivalent formulation for the expectation over policy π:

$$\begin{aligned}\mathbb{E}_\pi\left[r(s,a)\right] &= \sum_{t=0}^{\infty}\sum_s P(s_t=s|\pi)\sum_a \pi(a|s)\gamma^t r(s,a)\\ &= \sum_s \rho_\pi(s)\sum_a \pi(a|s) r(s,a)\\ &= \sum_{s,a}\rho_\pi(s,a) r(s,a),\end{aligned} \tag{2}$$

which provides an alternative way to compute the expected discounted return. An important property of the occupancy measure is that it uniquely specifies a policy, as described in the following lemma.

Lemma 1. *(Theorem 2 of (Syed et al., 2008)) Suppose ρ is the occupancy measure for $\pi_\rho(a|s) \triangleq \frac{\rho(s,a)}{\sum_{a'}\rho(s,a')}$. Then π_ρ is the only policy whose occupancy measure is ρ.*

3.2. Problem Definition

In practice, most RL tasks and environments do not provide a well-designed reward function. Instead, only a little *sparse* feedback indicating whether the goal is reached is available. Existing RL algorithms usually fail to explore and collect useful information in such sparse-reward settings. In this work, we are interested in solving such a challenging problem by developing a method capable of learning from both (sparse) rewards and expert demonstrations.

In particular, in addition to sparse rewards from environments as in traditional RL settings, the agent is also provided with a few (and possibly imperfect) demonstrations $\mathcal{D}^E = \{\tau_1, \tau_2, \dots, \tau_N\}$, where the i-th trajectory $\tau_i = \{(s_0^i, a_0^i), (s_1^i, a_1^i), \dots, (s_T^i, a_T^i)\}$ is generated from executing an unknown expert policy π_E in the environment. We aim to develop a method that can boost exploration through effectively leveraging $\mathcal{D}^E$ in such settings and maximize $\eta(\pi)$ in Eqn. (1).

Throughout the paper, we use π_E to denote the expert policy that gives the relatively good $\eta(\pi)$, and use $\hat{\mathbb{E}}_\mathcal{D}$ to denote empirical expectation estimated from the demonstrated trajectories $\mathcal{D}^E$. We have the following reasonable and necessary assumption on the quality of the expert policy π_E.

Assumption 1. *In early learning stages, we assume acting according to expert policy π_E will provide higher advantage value with a margin as least δ over current policy π,* i.e.*,*

$$\mathbb{E}_{a_E\sim\pi_E, a\sim\pi}[A_\pi(s,a_E) - A_\pi(s,a)] \geq \delta.$$

We do not need to assume the expert policy π_E to be advantageous over all the policies, as our proposed POfD will learn from both rewards and demonstrations and can possibly learn a policy better than π_E through exploration on its own in later learning stages.

4. Method

4.1. Policy Optimization with Demonstrations

Suppose π_θ is a θ-parameterized policy and is differentiable. Policy gradient methods optimize the expected return $\eta(\pi_\theta)$ by updating θ with the gradient of $\eta(\pi_\theta)$ w.r.t. θ (Sutton et al., 2000; Williams, 1992). They usually start optimizing the policy π_θ by random exploration, which may cause slow convergence when the state and action spaces have high dimensionality, and may even lead to failure when the environment feedback is sparse. We propose to address this problem by forcing the policy to explore in the nearby region of the expert policy π_E (as shown in Fig.1), which is specified by several demonstrated trajectories $\mathcal{D}^E$.

To this end, besides maximizing the expected return $\eta(\pi_\theta)$ through learning from sparse feedback during interaction with environments, we also encourage the policy π

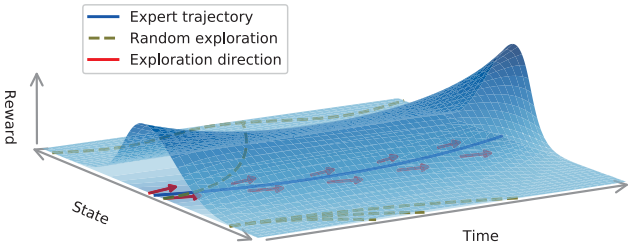

Figure 1. Our proposed POfD explores in the high-reward regions (red arrows), with the aid of demonstrations (the blue curve). It thus performs better than random explorations (olive green dashed curves) in sparse-reward environments.

to explore by "following" the demonstrations $\mathcal{D}^E$. We therefore introduce demonstration-guided exploration term $\mathcal{L}_M(\pi_\theta, \pi_E) = D_{JS}(\pi_\theta, \pi_E)$, which is defined over Jensen-Shannon divergence between current policy π_θ and the expert one π_E, to the vanilla objective $\eta(\pi_\theta)$. This gives a new learning objective:

$$\mathcal{L}(\pi_\theta) = -\eta(\pi_\theta) + \lambda_1 D_{JS}(\pi_\theta, \pi_E),$$

where λ_1 is a trading-off parameter. However, the above policy divergence measure between π_θ and π_E is infeasible as π_E is unknown. Fortunately, leveraging the one-to-one correspondence between the policy and occupancy measure as given by Lemma 1, we can instead define a divergence over the occupancy measures $\rho_\pi(s,a)$ and $\rho_{\pi_E}(s,a)$, which is easier to optimize through adversarial training on demonstrations, as we will show later. Based on their occupancy measure $\mathcal{L}_M \triangleq D_{JS}(\rho_\theta, \rho_E)$, where ρ_θ and ρ_E are short for ρ_{π_θ} and ρ_{π_E}, our proposed demonstration guided learning objective is

$$\mathcal{L}(\pi_\theta) = -\eta(\pi_\theta) + \lambda_1 D_{JS}(\rho_\theta, \rho_E). \tag{3}$$

We here slightly abuse D_{JS} to apply it to the unnormalized distribution ρ. Later, we will establish a lower bound to it without needing to directly optimize it.

4.2. Benefits of Exploration with Demonstrations

To better understand the new objective (3), we here prove that introducing the guiding term $D_{JS}(\rho_\theta, \rho_E)$ boosts the advantage value for the learned policy, and brings non-trivial benefits in terms of policy improvement for policy gradient methods. To see this, we introduce following the useful expression at first (Kakade & Langford, 2002) that is commonly used in policy gradient methods (Schulman et al., 2015; 2017; Kakade, 2002),

$$\eta(\pi) = \eta(\pi_{\text{old}}) + \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t A_{\pi_{\text{old}}}(s,a)\right]. \tag{4}$$

In the above, the expected return $\eta(\pi)$ of policy π is expressed in terms of the advantage over the policy π_{old} in the previous iteration. Eqn. (4) can be rewritten as

$$\eta(\pi) = \eta(\pi_{\text{old}}) + \sum_s \rho_\pi(s)\sum_a \pi(a|s)A_{\pi_{\text{old}}}(s,a). \tag{5}$$

To alleviate difficulties brought by complex dependency of $\rho_\pi(s)$ over π, policy gradient methods usually optimize the following surrogate objective, which is a local approximation to $\eta(\pi)$ up to first order:

$$J_{\pi_{\text{old}}}(\pi) = \eta(\pi_{\text{old}}) + \sum_s \rho_{\pi_{\text{old}}}(s)\sum_a \pi(a|s)A_{\pi_{\text{old}}}(s,a),$$

where ρ_π is replaced by $\rho_{\pi_{\text{old}}}$ and we ignore the change in state distribution due to policy update. Policy gradient methods are guaranteed to improve $\eta(\pi)$ monotonically by optimizing the above surrogate $J_{\pi_{\text{old}}}(\pi)$ with a sufficiently small update step $\pi_{\text{old}} \to \pi$ such that $D_{KL}^{\max}(\pi, \pi_{\text{old}})$ is bounded (Schulman et al., 2015; 2017; Kakade, 2002). In particular, TRPO (Schulman et al., 2015) imposes hard constraints with fixed penalty on $D_{KL}^{\max}(\pi, \pi_{\text{old}})$ while PPO (Schulman et al., 2017) and natural policy gradient (Kakade, 2002) use $D_{KL}^{\max}(\pi, \pi_{\text{old}})$ for regularization with fixed or adaptive weights.

POfD additionally imposes a regularization $D_{JS}(\pi_\theta, \pi_E)$ between π_θ and π_E in order to encourage explorations around regions demonstrated by the expert π_E. We formally show the benefits from leveraging the expert demonstration in this way by giving the following theorem.

Theorem 1. *Let* $\alpha = D_{KL}^{\max}(\pi_{old}, \pi) = \max_s D_{KL}(\pi(\cdot|s), \pi_{old}(\cdot|s))$, $\beta = D_{JS}^{\max}(\pi_E, \pi) = \max_s D_{JS}(\pi(\cdot|s), \pi_E(\cdot|s))$, *and* π_E *is an expert policy satisfying Assumption 1. Then we have*

$$\eta(\pi) \geq J_{\pi_{old}}(\pi) - \frac{2\gamma(4\beta\epsilon_E + \alpha\epsilon_\pi)}{(1-\gamma)^2} + \frac{\delta}{1-\gamma},$$

where $\epsilon_E = \max_{s,a}|A_{\pi_E}(s,a)|, \epsilon_\pi = \max_{s,a}|A_\pi(s,a)|$.

We provide proof in the supplement. The above theorem implies the benefits of adding matching regularization. Let $M_i(\pi) = J_{\pi_i}(\pi) - C_{\pi_E} D_{JS}^{\max}(\pi, \pi_E) - C_\pi D_{KL}^{\max}(\pi, \pi_i) + \hat{\delta}$ where $C_{\pi_E} = \frac{8\gamma\epsilon_E}{(1-\gamma)^2}, C_\pi = \frac{2\gamma\epsilon_\pi}{(1-\gamma)^2}, \hat{\delta} = \frac{\delta}{1-\gamma}$. Then,

$$\begin{aligned}
&\eta(\pi_{i+1}) \geq M_i(\pi_{i+1}),\\
&\eta(\pi_i) = M_i(\pi_i) + C_{\pi_E} D_{JS}^{\max}(\pi_i, \pi_E) - \hat{\delta},\\
&\eta(\pi_{i+1}) - \eta(\pi_i)\\
&\qquad \geq M_i(\pi_{i+1}) - M_i(\pi_i) - C_{\pi_E} D_{JS}^{\max}(\pi_i, \pi_E) + \hat{\delta}.
\end{aligned}$$

The above result is reminiscent of classic monotonic improvement guarantees for policy gradient methods (Schulman et al., 2015; 2017; Kakade, 2002). However, POfD brings another factor to the improvement $-C_{\pi_E} D_{JS}^{\max}(\pi_i, \pi_E) + \hat{\delta}$. This implies that following the demonstrations (*i.e.*, having small $D_{JS}^{\max}(\pi_i, \pi_E)$) will fully utilize the advantage $\hat{\delta}$ and bring improvement with a margin over pure policy gradient methods.

4.3. Optimization

In this subsection, we introduce a practical optimization algorithm for Eqn. (3), which is compatible with any policy gradient methods. In particular, instead of performing optimization on the difficult Jensen-Shannon divergence directly, we optimize its lower bound given as follows.

Theorem 2. *Let* $h(u) = \log(\frac{1}{1+e^{-u}})$, $\bar{h}(u) = \log(\frac{e^{-u}}{1+e^{-u}})$ *and* $U(s,a) : \mathcal{S} \times A \to \mathbb{R}$ *be an arbitrary function. Then we have*

$$D_{JS}(\rho_\pi, \rho_E) \geq \sup_U \left(\mathbb{E}_{\rho_\pi}[h(U(s,a))] + \mathbb{E}_{\rho_E}[\bar{h}(U(s,a))]\right) + \log 4.$$

We defer the proof to supplement due to space limit. With the above theorem, the occupancy measure matching objective $\mathcal{L}_M$ can be written as

$$\mathcal{L}_M \triangleq \sup_{D \in (0,1)^{\mathcal{S}\times\mathcal{A}}} \mathbb{E}_{\pi_\theta}[\log(D(s,a))] + \mathbb{E}_{\pi_E}[\log(1 - D(s,a))],$$

where $D(s,a) = \frac{1}{1+e^{-U(s,a)}} : \mathcal{S} \times \mathcal{A} \to (0,1)$ is an arbitrary mapping function followed by a sigmoid activation function for scaling. The supremum ranging over $D(s,a)$ thus represents the optimal binary classification loss of distinguishing the current policy π_θ and the expert policy π_E w.r.t. the state-action pairs sampled from ρ_θ and ρ_E.

To avoid potential overfitting risks, we introduce causal entropy $-H(\pi_\theta)$ as another regularization term, similar to (Ziebart et al., 2008; Ziebart, 2010). Therefore, the overall objective of our proposed POfD is formulated as

$$\min_\theta \mathcal{L} = -\eta(\pi_\theta) - \lambda_2 H(\pi_\theta) + \lambda_1 \sup_{D \in (0,1)^{\mathcal{S}\times\mathcal{A}}} \mathbb{E}_{\pi_\theta}[\log(D(s,a))] + \mathbb{E}_{\pi_E}[\log(1 - D(s,a))].$$

It is actually a minimax problem closely related to the learning target of Generative Adversarial Networks (GANs) (Goodfellow et al., 2014). GANs aim to train a generative model G to produce samples indistinguishable from the ones from real distributions for a well-trained discriminative model D. In our case, the true distribution is the expert policy $\pi_E(a|s)$, or equivalently the expert occupancy measure $\rho_E(s,a)$. The generator to learn is the policy model $\pi_\theta(a|s)$. Suppose D is parameterized by w. By labeling expert state-action pairs as true ("1") and policy state-action pairs as false ("0"), we get the following objective,

$$\min_\theta \max_w \mathcal{L} = -\eta(\pi_\theta) - \lambda_2 H(\pi_\theta) + \lambda_1 \left(\mathbb{E}_{\pi_\theta}[\log(D_w(s,a))] + \mathbb{E}_{\pi_E}[\log(1 - D_w(s,a))]\right). \tag{6}$$

Moreover, the minimax objective (6) can be simplified by substituting Eqn. (1) and Eqn. (2) into it, resulting in a dynamic reward reshaping mechanism over the original reward signal:

$$\min_\theta \max_w -\mathbb{E}_{\pi_\theta}[r'(s,a)] - \lambda_2 H(\pi_\theta) + \lambda_1 \mathbb{E}_{\pi_E}[\log(1 - D_w(s,a))], \tag{7}$$

where $r'(s,a) = r(a,b) - \lambda_1 \log(D_w(s,a))$ is the reshaped reward function. This function augments the environment reward with demonstration information. When the environment feedback is sparse or exploration is insufficient, the augmented reward can force the policy to generate similar trajectories as the expert policy π_E. In other words, the divergence of π and π_E is minimized. Therefore, our algorithm is able to explore the environment more efficiently.

The above objective can be optimized efficiently by alternately updating policy parameters θ and discriminator parameters w. First, trajectories are sampled by executing the current policy π_θ and mixed with demonstration data to train the discriminator by SGD. The gradient is given by

$$\mathbb{E}_\pi[\nabla_w \log(D_w(s,a))] + \mathbb{E}_{\pi_E}[\nabla_w \log(1 - D_w(s,a))].$$

Then, fixing the discriminator D_w, *i.e.*, a fixed reward function $r'(s,a)$, the policy π_θ is optimized with a chosen policy gradient method. In particular, by policy gradient theorem (Sutton et al., 2000), the reshaped policy gradient is:

$$\nabla_\theta \mathbb{E}_{\pi_\theta}[r'(s,a)] = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s) Q'(s,a)],$$
$$\text{where} \quad Q'(\bar{s},\bar{a}) = \mathbb{E}_{\pi_\theta}[r'(s,a)|s_0 = \bar{s}, a_0 = \bar{a}].$$

The gradient for causal entropy regularization is given by

$$\nabla_\theta \mathbb{E}_{\pi_\theta}[-\log \pi_\theta(a|s)] = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(a|s) Q^H(s,a)],$$
$$\text{where } Q^H(\bar{s},\bar{a}) = \mathbb{E}_{\pi_\theta}[-\log \pi_\theta(a|s)|s_0 = \bar{s}, a_0 = \bar{a}].$$

The optimization details are summarized in Alg. 1. It is compatible with any policy gradient methods, *e.g.*, TRPO (Schulman et al., 2015) and PPO (Schulman et al., 2017).

5. Discussion on Existing LfD Methods

DQfD (Hester et al., 2017) and DDPGfD (Večerík et al., 2017) are two latest LfD methods. They both leverage demonstrations to aid exploration in RL, aiming to improve RL in terms of either convergence speed or performance. Here we provide a new perspective that interprets DQfD and DDPGfD through occupancy measure matching, which thus connects them with our POfD.

DQfD The DQfD method is built upon Deep Q-Networks (DQN) (Mnih et al., 2015). As a Q-learning algorithm, DQN models the Q value with a w-parameterized neural network $Q_w(s,a)$. The objective for optimizing Q_w is

$$J_{\text{DQN}} = \mathbb{E}[(R_t(n) - Q_w(s_t,a_t))^2], \text{ where } R_t(n) = \sum_{i=t}^{t+n-1} \gamma^{i-t} r_i + \max_a \gamma^n Q_w(s_{t+n},a). \tag{8}$$

Algorithm 1 Policy optimization with demonstrations

Input: Expert demonstrations $\mathcal{D}_E = \{\tau_1^E, \dots, \tau_N^E\}$, initial policy and discriminator parameters θ_0 and w_0, regularization weights λ_1, λ_2, maximal iterations I.
for $i = 1$ to I **do**
Sample trajectories $\mathcal{D}_i = \{\tau\}, \tau \sim \pi_{\theta_i}$.
Sample expert trajectories $\mathcal{D}_i^E \subset \mathcal{D}^E$.
Update discriminator parameters from w_i to w_{i+1} with the gradient

$$\hat{\mathbb{E}}_{\mathcal{D}_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\mathcal{D}_i^E}[\nabla_w \log(1 - D_w(s,a))]$$

Update the rewards in $\mathcal{D}_i$ with

$$r'(s,a) = r(a,b) - \lambda_1 \log(D_{w_i}(s,a)), \forall (s,a,r) \in \mathcal{D}_i$$

Update the policy with policy gradient method (*e.g.*, TRPO, PPO) using the following gradient

$$\hat{\mathbb{E}}_{\mathcal{D}_i}[\nabla_\theta \log \pi_\theta(a|s) Q'(s,a)] - \lambda_2 \nabla_\theta H(\pi_{\theta_i})$$

end for

DQfD takes advantage of demonstration data by putting them into a replay memory $\mathcal{D}$ and keeping them throughout the Q-learning process. Here we show minimizing Eqn. (8) with expert demonstration and self-generated off-policy data is actually equivalent to imposing an occupancy measure matching regularization to the original DQN objective [1].

Let $\mathcal{D}^E$ denote the replay memory containing expert data only. Then the objective (8) for DQfD can be separated as

$$\begin{aligned} J_{\text{DQfD}} = \hat{\mathbb{E}}_{\mathcal{D}}[(R_t(n) - Q_w(s_t,a_t))^2] + \\ \alpha \hat{\mathbb{E}}_{\mathcal{D}^E}[(R_t(n) - Q_w(s_t,a_t))^2], \end{aligned} \tag{9}$$

where α is specified by the ratio of samples from $\mathcal{D}$ and $\mathcal{D}^E$. Based on Eqn. (2), $Q_w(s,a) = \mathbb{E}[r(\cdot,\cdot)|s_0{=}s, a_0{=}a] = \rho_\pi(s,a) r(s,a)$, and $Q^E(s,a) = \rho_E(s,a) r(s,a)$. Thus

$$\begin{aligned} &\hat{\mathbb{E}}_{\mathcal{D}^E}[(R_t(n) - Q_w(s_t,a_t))^2] \\ &= \hat{\mathbb{E}}_{\mathcal{D}^E}\left[\left(\hat{\rho}_E(s,a) - \rho_\pi(s,a)\right)^2 r^2(s,a)\right], \end{aligned} \tag{10}$$

which can be interpreted as a regularization forcing current policy's occupancy measure to match the expert's empirical occupancy measure, weighted by the potential reward we will get from that state-action pair. For high reward state-action pairs, there is a greater penalty on the discrepancy between $\hat{\rho}_E$ and ρ_π.

[1] DQfD has two training stages and an extra supervised loss $J_E(Q) = \max_{a \in A}[A(s,a) + \ell(a_E, a) - Q(s, a_E)]$. In the pre-training stage, DQfD trains on the demonstrations with J_E. In the RL training stage, J_E becomes less significant. Since we are interested in the Q-learning part of DQfD and how it utilizes demonstration data, we focus on analyzing objective (8).

DDPGfD Similar to DQfD, DDPGfD leverages the demonstration data by putting them into the replay memory. But DDPGfD is based on an actor-critic framework (Lillicrap et al., 2015). Aside from a Q-network Q_w, DDPGfD introduces another policy network parameterized by θ to model a deterministic policy $\pi_\theta(s)$. Its Q-network is optimized off-policy with Eqn. (8), while its policy network is optimized directly with the following gradient of Q-value $Q_w(s,a)$ w.r.t. the action $a = \pi_\theta(s)$:

$$\nabla_\theta J_{\text{DDPGfD}} \approx \mathbb{E}_{s,a}[\nabla_a Q_w(s,a) \nabla_\theta \pi_\theta(s)], a = \pi_\theta(s).$$

The above equation shows that the update of policy π_θ is not directly dependent on the demonstration data $\mathcal{D}_E$, but depends on the learned Q-network Q_w solely. Since DDPGfD shares the same objective function for Q_w as DQfD, as well as the same way of leveraging demonstrations as shown in Eqn. (9) and Eqn. (10). Thus we can draw a similar conclusion, *i.e.* demonstrations in DDPGfD induce an occupancy measure matching regularization.

Although the above replay memory based LfD methods can benefit RL algorithms to some extent in sparse-reward environments, they can not sufficiently exploit the demonstration data due to following limitations. First, such a paradigm utilizes expert trajectories only by treating them as learning reference, whose effect may be significantly underexploited when demonstrations are few, as verified by our experiments. Second, to be compatible with collected data during training, the demonstrated trajectories are required to be associated with rewards for each state transition. However, the rewards in demonstrations may differ from the ones used for learning the policy in the current environment (Ziebart et al., 2008), or they may be unavailable.

By reformulating the original objective (9) into (10), we find that, instead of mixing expert data with self-generated data, putting an occupancy measure matching regularization can avoid the requirement of rewards in demonstrations, as what POfD does. In this way, LfD will no longer be restricted to the off-policy RL setting.

6. Experiments

In this section, we aim at investigating 1) whether POfD can aid exploration when provided with a few demonstrations, even though the demonstrations are imperfect, and 2) whether POfD can succeed and achieve high empirical return, especially when feedback of the environment is extremely sparse. To comprehensively assess our method, we conduct extensive experiments on eight widely used physical control tasks, ranging from low-dimensional ones such as cartpole (Barto et al., 1983) and mountain car (Moore, 1990) to high-dimensional and naturally sparse environments based on OpenAI Gym (Brockman et al., 2016) and Mujoco (Todorov et al., 2012). The specifications of envi-

ronments we used are given in Table 1.

Settings In view of the property of each individual environment, we apply four ways for sparsifying their built-in dense rewards for evaluating performance of different methods in sparse-reward environments, detailed as follows. *TYPE1*: a reward of $+1$ is given when the agent reaches the terminal state, and otherwisely 0. This is also employed by (Houthooft et al., 2016) for MountainCar. *TYPE2*: a reward of $+1$ is given when the agent survives for a while (*e.g.*, 200 steps for CartPole). *TYPE3*: a reward of $+1$ is given for every time the agent moves forward over a specific number of units in Mujoco environments. *TYPE4*: specially designed for InvertedDoublePendulum, a reward $+1$ is given when the second pole stays above a specific height of 0.89. Instead of providing dense rewards to state-action pairs at every step, the environment directly tells the agent the target state, *e.g.* reaching a goal and moving forward, through providing sparse rewards. This is more practical and hence we do not consider extensive reward engineering. For clearer distinction, we call the original simulator *dense environments*, while the ones with altered reward are called *sparse environments*.

Without specification, we use only *one single imperfect* trajectory as demonstrations. To collect demonstrations, we train an agent insufficiently by running TRPO (Schulman et al., 2015) in the corresponding dense environment. Then, an imperfect trajectory is randomly sampled by executing the agent. Our POfD is evaluated using two metrics. First, for each sparse environment, the method is run for 5 times with different random initialization and we investigate training curves to understand how POfD facilitates exploration. Second, empirical returns in numerical values are calculated by averaging over 500 cumulated rewards for the learned policy. We compare POfD against five strong baselines, including 1) training the policy with TRPO (Schulman et al., 2015) in *dense* environments which is referred to as *expert*; 2) training the policy with TRPO in sparse environments; 3) applying GAIL (Ho & Ermon, 2016) to learn the policy from demonstrations, under the same setting as our POfD; 4) DQfD and 5) DDPGfD, which are the state-of-the-art LfD algorithms for discrete and continuous actions respectively.

Implementation Details Due to space limit, we defer implementation details to the supplementary material.

Results We first test the performance of POfD in sparse control environments with discrete actions, including *CartPole* and *MountainCar*. The learning curves are plotted in Fig. 2, while the averaged cumulated rewards are given in Table 1. From the numerical results in Table 1, we can see that our method achieves performance comparable with the policy learned under dense environments, *e.g.*, -98.38 *vs* -98.75 for CartPole, even though POfD only has access to sparse rewards and a single demonstration trajectory that is much worse than the expert. The learning curves in Fig. 2 demonstrate that both TRPO and DQfD fail to explore sufficiently to obtain informative feedback from the sparse environments. The demonstration data insufficiency severely limits the learning ability of DQfD which usually requires as many demonstration data as self-generated ones. Furthermore, GAIL succeeds in CartPole but converges to the imperfect demonstration data in MontainCar. Our POfD outperforms GAIL by a significant margin in terms of both convergence rate and final performance.

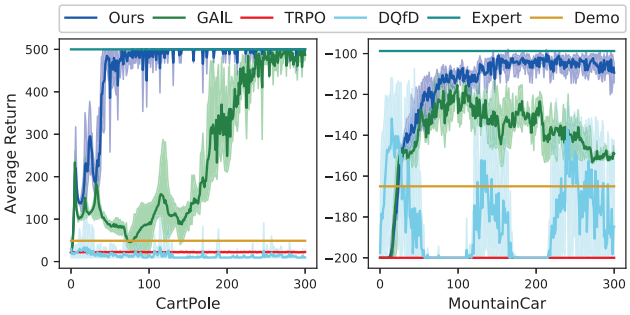

Figure 2. Learning curves of our POfD versus strong baselines under sparse environments with discrete action space.

Then POfD as well as the baselines are evaluated in sparse locomotion control tasks, including *Hopper*, *HalfCheetah*, *Walker2d* and *InvertedDoublePendulum*. Similarly, the numerical results are listed in Table 1 while the learning course is visualized in Fig. 3. Throughout the four experiments, POfD achieves expert-level performance in terms of cumulated rewards. Moreover, POfD surpasses TRPO in dense environments substantially in multiple environments. For example, observing results of Walker2D, POfD obtains an averaged return of 7687.47, while the expert is only 6717.08. The margin becomes more remarkable considering the return of demonstration data we use is only 1701.13, strongly proving that our intrinsic reward reshaping mechanism benefits the exploration a lot. Observing the learning process of different methods, it is clear that TRPO consistently fails to explore the environments when the feedback is sparse, except for HalfCheetah. This may be because there is no terminal state in HalfCheetah, thus a random agent can perform reasonably well as long as the time horizon is sufficiently long. This can be observed in the figure where the improvement of TRPO emerges very late (after 400 iterations). DDPGfD and GAIL share some common drawback: they both converge to the imperfect demonstration data as training proceeds. For HalfCheetah, GAIL fails to converge and DDPGfD converges to an even worse point. This is well expected because the policy and value networks are prone to over-fitting when the data are few. Thus the

Table 1. Environment specifications and results. All results are measured in the corresponding dense environment expect for Reacher which is evaluated in a naturally sparse environment.

Environment	$\mathcal{S}$	$\mathcal{A}$	Sparsification	Empirical Return		
				Demonstration	Expert	Ours
MountainCar-v1	$\mathbb{R}^2$	$\{0, 1, 2\}$	TYPE1	-165.0	-98.75	$\mathbf{-98.35 \pm 9.35}$
CartPole-v0	$\mathbb{R}^4$	$\{0, 1\}$	TYPE2	49	**500**	$\mathbf{500 \pm 0}$
Hopper-v1	$\mathbb{R}^{11}$	$\mathbb{R}^3$	TYPE3 (1 unit)	793.86	3571.38	$\mathbf{3652.23 \pm 263.62}$
HalfCheetah-v1	$\mathbb{R}^{17}$	$\mathbb{R}^6$	TYPE3 (15 unit)	1827.77	4463.46	$\mathbf{4771.15 \pm 646.96}$
Walker2d-v1	$\mathbb{R}^{17}$	$\mathbb{R}^6$	TYPE3 (1 unit)	1701.13	6717.08	$\mathbf{7687.47 \pm 394.97}$
DoublePendulum-v1	$\mathbb{R}^{11}$	$\mathbb{R}$	TYPE4	520.23	8399.86	$\mathbf{9116.08 \pm 1290.74}$
Humanoid-v1	$\mathbb{R}^{376}$	$\mathbb{R}^{17}$	TYPE3 (1 unit)	2800.05	9575.40	$\mathbf{9823.43 \pm 2015.48}$
Reacher-v1	$\mathbb{R}^{11}$	$\mathbb{R}^2$	TYPE1	0.73	0.75	$\mathbf{0.86 \pm 0.34}$

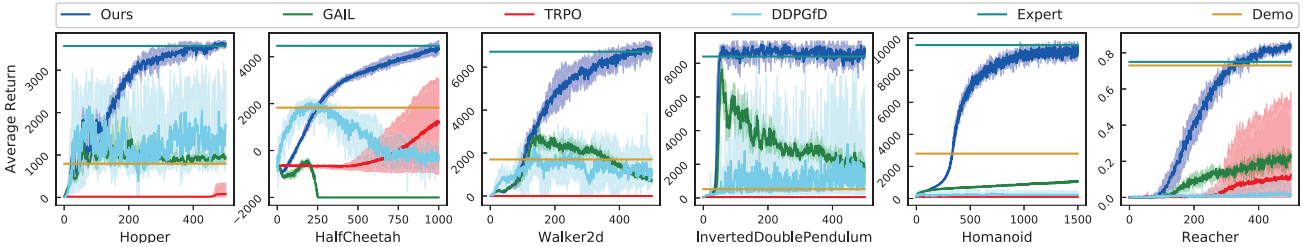

Figure 3. Learning curves of our POfD versus baselines under sparse environments with continuous action space.

training process of GAIL and DDPGfD is severely biased by the imperfect data. Eventually, our proposed method can effectively explore the environment with the help of demonstration-based intrinsic reward reshaping, and succeeds consistently across different tasks both in terms of learning stability and convergence speed.

Humanoid is a locomotion task about teaching a human-like robot to walk naturally. This task has a state space of dimension as high as 376, rendering it much harder than the above investigated tasks. The experimental results in Table 1 indicate that POfD still achieves expert-level performance. The learning curves in Fig. 3 demonstrate that all the other three baseline methods, *i.e.*, TRPO, GAIL and DDPGfD, fail to learn good policies from such a reward-sparse environment. In particular, the LfD baselines, *i.e.*, GAIL and DDPGfD, cannot even reach the demonstration performance (the yellow horizontal line in the figure). In stark contrast, POfD converges fast and stably to the expert level performance. Such results strongly support that POfD can generalize well to challenging tasks with high-dimensional state spaces.

The last environment we use for evaluation is *Reacher*. It is an environment where rewards are naturally sparse and is difficult to solve. The target of this task is to control the robot arm to touch the target red ball. The environment reward is sparse: every time the arm reaches the ball and holds for a while (*e.g.*, 5 time steps), it receives a reward of $+1$; otherwise it gets zero reward. For every instantiation of the environment, the location of the target ball is randomly generated. This increases the task difficulty significantly. In Gym implementation, it is significantly simplified by crafting a distance-to-target based reward function and we only use it to generate demonstrations. In the experiments, we randomly select 15 trajectories as demonstration data. The numerical results in Table 1 show that the performance of our POfD (0.86) is much better than the expert (0.75). The learning curves in Fig 3 also demonstrate that all the other baseline methods fail on this task.

7. Conclusion

We considered the problems of reinforcement learning within sparse-reward environments, and proposed a general learning from demonstration method, *i.e.*, POfD to gain stronger exploration ability. POfD is compatible with any policy gradient methods. We explicitly analyzed how POfD improves the policy by leveraging the benefits of demonstration for exploration. A simple dynamic reward reshaping based optimization algorithm for POfD was provided that connects to the generative adversarial training and can be applied efficiently. The POfD was shown to be effective in encouraging the agent to explore around the nearby region of the expert policy and learning better policies, through extensive experimental results. To our best knowledge, POfD is the first one that can acquire knowledge from few and imperfect demonstration data to aid exploration in environments with sparse feedback.

Acknowledgements

This work was partially supported by NUS startup R-263-000-C08-133, MOE Tier-I R-263-000-C21-112, NUS IDS R-263-000-C67-646, ECRA R-263-000-C87-133 and MOE Tier-II R-263-000-D17-112.

References

Aravind S. Lakshminarayanan, Sherjil Ozair, Y. B. Reinforcement learning with few expert demonstrations. In *NIPS workshop*, 2016.

Barto, A. G., Sutton, R. S., and Anderson, C. W. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym, 2016.

Brys, T., Harutyunyan, A., Suay, H. B., Chernova, S., Taylor, M. E., and Nowé, A. Reinforcement learning from demonstration through shaping. In *IJCAI*, pp. 3352–3358, 2015.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Sendonaris, A., Dulac-Arnold, G., Osband, I., Agapiou, J., et al. Learning from demonstrations for real world reinforcement learning. *arXiv preprint arXiv:1704.03732*, 2017.

Ho, J. and Ermon, S. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, pp. 4565–4573, 2016.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., and Abbeel, P. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*, pp. 1109–1117, 2016.

Kakade, S. and Langford, J. Approximately optimal approximate reinforcement learning. In *ICML*, volume 2, pp. 267–274, 2002.

Kakade, S. M. A natural policy gradient. In *Advances in neural information processing systems*, pp. 1531–1538, 2002.

Kim, B., Farahmand, A.-m., Pineau, J., and Precup, D. Learning from limited demonstrations. In *Advances in Neural Information Processing Systems*, pp. 2859–2867, 2013.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Li, Y., Song, J., and Ermon, S. Infogail: Interpretable imitation learning from visual demonstrations. In *Advances in Neural Information Processing Systems*, pp. 3815–3825, 2017.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, 2015.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937, 2016.

Moore, A. W. Efficient memory-based learning for robot control. 1990.

Nair, A., McGrew, B., Andrychowicz, M., Zaremba, W., and Abbeel, P. Overcoming exploration in reinforcement learning with demonstrations. *arXiv preprint arXiv:1709.10089*, 2017.

Ng, A. Y., Russell, S. J., et al. Algorithms for inverse reinforcement learning. In *Icml*, pp. 663–670, 2000.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, volume 2017, 2017.

Piot, B., Geist, M., and Pietquin, O. Boosted bellman residual minimization handling expert demonstrations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 549–564. Springer, 2014.

Plappert, M., Houthooft, R., Dhariwal, P., Sidor, S., Chen, R. Y., Chen, X., Asfour, T., Abbeel, P., and Andrychowicz, M. Parameter space noise for exploration. *arXiv preprint arXiv:1706.01905*, 2017.

Schaal, S. Learning from demonstration. In *Advances in neural information processing systems*, pp. 1040–1046, 1997.

Schaul, T., Quan, J., Antonoglou, I., and Silver, D. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1889–1897, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

Subramanian, K., Isbell Jr, C. L., and Thomaz, A. L. Exploration from demonstration for interactive reinforcement learning. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pp. 447–456. International Foundation for Autonomous Agents and Multiagent Systems, 2016.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Syed, U. and Schapire, R. E. A game-theoretic approach to apprenticeship learning. In *Advances in neural information processing systems*, pp. 1449–1456, 2008.

Syed, U., Bowling, M., and Schapire, R. E. Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*, pp. 1032–1039. ACM, 2008.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 5026–5033. IEEE, 2012.

Van Hasselt, H., Guez, A., and Silver, D. Deep reinforcement learning with double q-learning. In *AAAI*, volume 16, pp. 2094–2100, 2016.

Večerík, M., Hester, T., Scholz, J., Wang, F., Pietquin, O., Piot, B., Heess, N., Rothörl, T., Lampe, T., and Riedmiller, M. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv:1707.08817*, 2017.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Zhu, Y., Wang, Z., Merel, J., Rusu, A., Erez, T., Cabi, S., Tunyasuvunakool, S., Kramár, J., Hadsell, R., de Freitas, N., et al. Reinforcement and imitation learning for diverse visuomotor skills. *arXiv preprint arXiv:1802.09564*, 2018.

Ziebart, B. D. *Modeling purposeful adaptive behavior with the principle of maximum causal entropy*. Carnegie Mellon University, 2010.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. Maximum entropy inverse reinforcement learning. In *AAAI*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.