PLAN YOUR TARGET AND LEARN YOUR SKILLS: STATE-ONLY IMITATION LEARNING VIA DECOUPLED POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

State-only imitation learning (SOIL) enables agents to learn from massive demonstrations without explicit action or reward information. However, previous methods attempt to learn the implicit state-to-action mapping policy directly from state-only data, which results in ambiguity and inefficiency. In this paper, we overcome this issue by introducing hyper-policy as sets of policies that share the same state transition to characterize the optimality in SOIL. Accordingly, we propose Decoupled Policy Optimization (DPO) via explicitly decoupling the state-to-action mapping policy as a state transition predictor and an inverse dynamics model. Intuitively, we teach the agent to plan the target to go and then learn its own skills to reach. In-depth analyzed experiments on simulated environment and a real-world driving dataset demonstrate the effectiveness of DPO and its potential of bridging the gap between reality and simulations of reinforcement learning.

1 INTRODUCTION

Imitation learning offers a way to train an intelligent agent from demonstrations by mimicking the expert's behaviors without constructing hand-crafted reward functions (Hussein et al., 2017; Liu et al., 2021). The corresponding methods normally require the expert demonstrations include information of both states and actions. Unfortunately, the action information is not always accessible from many real-world demonstration resources, e.g., online video recordings of car driving or sports. Thus a natural desire to take advantage of these massive and valuable resources motivates the study of state-only imitation learning (SOIL), also known as learning from observations (LfO) (Torabi et al., 2019b). Analogy to human beings, SOIL is a more intuitive way to approach imitation by only matching the expert's state sequences without having explicit knowledge of the exact actions.

A wide range of algorithms have been proposed to solve SOIL by matching the state sequence of the expert (Torabi et al., 2018; 2019a). However, the action agnostic setting in SOIL makes it challenging to determine the optimal action because of the partial observability of the expert demonstrations that multiple policies could be chosen to match the same expert state sequence. Thus learning a state-to-action policy is implicit, leading to a less efficient modeling of the explicit information from demonstrations, and in result could cause suboptimality.

To this end, in this paper, we introduce the concept of *hyper-policy* denoting a *family* of policies that share the same state-state visitation. Based on that, instead of recovering the expert *policy*, we characterize the optimality in SOIL by finding the expert *hyper-policy*. The proposed method is called decoupled policy optimization (DPO), which separates the policy into two modules: an expert state transition predictor that finds the optimal *hyper-policy*, followed by an inverse dynamics model that builds the executable *policy* to deliver actions. Intuitively, the expert state transition predictor predicts the target, while the inverse dynamics model enables the agent to learn its own skills to reach the target. DPO takes the advantage of such a decoupled structure by explicitly modeling two kinds of data: (1) the expert state transition that is directly accessible in the demonstration; (2) the action to be performed which should be obtained by interacting with the environment.

To ensure the benefit of DPO, these two modules should work coherently to provide accurate foresight for targets and corresponding skills. To achieve this, we regularize the state transition predictor to prevent the model from predicting non-neighboring states via multi-step and cycle training style.

Further, to improve the learning efficiency by encouraging the agent to reach the expert states, we augment reward and apply policy gradient to DPO with additional generative adversarial objective.

In experiments, we conduct in-depth analysis for our method to show the advantage of the decoupled structure and the higher efficiency. We also evaluate DPO on a real-world driving dataset with state-only demonstrations, and the result shows that DPO can learn driving behaviors closer to human drivers when compared with baseline methods.

2 PRELIMINARIES

Markov Decision Process. Consider a γ-discounted infinite horizon Markov decision process (MDP) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0, r, \gamma \rangle$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is environment dynamics distribution, $\rho_0 : \mathcal{S} \to [0, 1]$ is the initial state distribution, and $\gamma \in [0, 1]$ is the discount factor. The agent makes decisions through a policy $\pi(a|s) : \mathcal{S} \times \mathcal{A} \to [0, 1]$ and receives rewards $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. For analyzing the effect of action ambiguity, we assume the dynamics $\mathcal{T}(s'|s, a)$ has redundant actions, i.e., its transition probabilities can be written as linear combination of other actions'. Formally, this refers to the existence of a state $s_m \in \mathcal{S}$, an action $a_n \in \mathcal{A}$ and a distribution p defined on $\mathcal{A} \setminus \{a_n\}$ such that $\int_{\mathcal{A}\setminus\{a_n\}} p(a)\mathcal{T}(s'|s_m, a)\,\mathrm{d}a = \mathcal{T}(s'|s_m, a_n)$.

Occupancy Measure. The concept of occupancy measure (OM) (Ho & Ermon, 2016) is proposed to characterize the statistical properties of a certain policy interacting with an MDP. Specifically, the state OM is defined as the time-discounted cumulative stationary density over the states under a given policy π: $\rho_\pi(s) = \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi)$. Following such a definition we can define different OM:

a) State-action OM: $\rho_\pi(s, a) = \pi(a|s)\rho_\pi(s)$

b) State transition OM: $\rho_\pi(s, s') = \int_{\mathcal{A}} \rho_\pi(s, a)\mathcal{T}(s'|s, a)\,\mathrm{d}a$

c) Joint OM: $\rho_\pi(s, a, s') = \rho_\pi(s, a)\mathcal{T}(s'|s, a)$

Imitation Learning from State-Only Demonstrations. Imitation learning (IL) (Hussein et al., 2017) studies the task of learning from demonstrations (LfD), which aims to learn a policy from expert demonstrations without getting access to the reward signals. The expert demonstrations typically consist of expert state-action pairs. General IL objective minimizes the state-action OM discrepancy:

$$\pi^* = \arg\min_\pi \mathbb{E}_{s\sim\rho_\pi^s} \left[\ell\left(\pi_E(\cdot|s), \pi(\cdot|s)\right)\right] \Rightarrow \arg\min_\pi \ell\left(\rho_{\pi_E}(s, a), \rho_\pi(s, a)\right) \ , \tag{1}$$

where ℓ denotes some distance metric. For example, GAIL (Ho & Ermon, 2016) chooses to minimize the JS divergence $\mathrm{D_{JS}}(\rho_{\pi_E}(s, a)\|\rho_\pi(s, a))$, and AIRL (Fu et al., 2018) utilizes the KL divergence $\mathrm{D_{KL}}(\rho_{\pi_E}(s, a)\|\rho_\pi(s, a))$ instead, which corresponds to a maximum entropy solution with the recovered reward (Liu et al., 2021). However, for the scenario studied in this paper, the action information is absent in state-only demonstrations. Such challenges prevent applying typical IL solutions. An popular solution (Torabi et al., 2019a) for this problem is to instead optimize the discrepancy of the state transition OM with the state-to-action policy $\pi(a|s)$:

$$\pi^* = \arg\min_\pi [\ell\left(\rho_{\pi_E}(s, s'), \rho_\pi(s, s')\right)] \ . \tag{2}$$

3 METHODOLOGY

3.1 RETHINKING THE OPTIMALITY IN SOIL

In standard IL tasks, when the expert actions are accessible in demonstrations, perfectly imitating the expert policy corresponds to matching the state-action OM due to the one-to-one correspondence between π and $\rho_\pi(s, a)$ (Ho & Ermon, 2016; Syed et al., 2008). However, such correspondence is not applicable for the state transition OM matching in SOIL.

Proposition 1. *Suppose Π is the policy space and $\mathcal{P}$ is a valid set of state transition OMs such that $\mathcal{P} = \{\rho : \rho \geq 0$ and $\exists \pi \in \Pi$, s.t. $\rho(s, s') = \rho_0(s) \int_a \pi(a|s)\mathcal{T}(s'|s, a)\,\mathrm{d}a + \int_{s'',a} \pi(a|s)\mathcal{T}(s'|s, a)\rho(s'', s)\,\mathrm{d}s''\,\mathrm{d}a\}$, then a policy $\pi \in \Pi$ corresponds to one state transition OM $\rho_\pi \in \mathcal{P}$. However, under the action-redundant assumption about the dynamics $\mathcal{T}$, a state transition OM $\rho \in \mathcal{P}$ can correspond to more than one policy in Π.*

The proof can be found in Appendix B. As a result, if we choose to optimize a state-to-action mapping policy, then the optimal solution to Eq. (2) is ambiguous. The ambiguity also comes from the fact that Eq. (2) does not correspond to a maximum policy entropy solution as in normal IL tasks (see Appendix C for details). Therefore, a state-to-action mapping function may be too implicit for matching the state sequence, which could cause training instability and lead to sub-optimal policies. In that case, we must find a one-to-one corresponding solution to solve SOIL explicitly and efficiently. Before continuing, we introduce the definition of *hyper-policy*.

Definition 1. *A hyper-policy $\Omega \in \Lambda$ is a maximal set of policies sharing the same state transition occupancy such that for any $\pi_1, \pi_2 \in \Omega$, we have $\rho_{\pi_1}(s, s') = \rho_{\pi_2}(s, s')$.*

Then by definition, there is a one-to-one correspondence between Ω and $\rho_\Omega(s, s')$. Similar to the normal state-to-action mapping policy, a hyper-policy Ω can be regarded as a state-to-state mapping function $h_\Omega(s'|s)$ which predicts the state transition such that for any $\pi \in \Omega$:

$$h_\Omega(s'|s) = \frac{\rho_\Omega(s, s')}{\int_{\tilde{s}} \rho_\Omega(s, \tilde{s})\, \mathrm{d}\tilde{s}} = \int_a \pi(a|s)\mathcal{T}(s'|s, a)\, \mathrm{d}a\ . \tag{3}$$

Proposition 2. *Suppose the state transition predictor h_Ω is defined as in Eq. (3) and $\Gamma = \{h_\Omega : \Omega \in \Lambda\}$ is a valid set of the state transition predictor, $\mathcal{P}$ is a valid set of the state-transition OM defined as in Proposition 1, then a state transition predictor $h_\Omega \in \Gamma$ corresponds to one state transition OM, where $\pi \in \Omega$; and a state transition OM $\rho \in \mathcal{P}$ only corresponds to one hyper-policy state transition predictor such that $h_\rho = \rho(s, s') / \int_{\tilde{s}} \rho(s, \tilde{s})\, \mathrm{d}\tilde{s}$.*

The proof can follow the Theorem 2 of Syed et al. (Syed et al., 2008), and for completness, we contain the proof in Appendix B. Therefore, we find a one-to-one correspondence between the optimization term $\rho(s, s')$ and a practical target $h_\Omega(s'|s)$, which indicates that under state-only demonstrations we only need to recover the state transition prediction of the hyper-policy Ω_E:

$$\arg\min_{\Omega} [\ell\left(\rho_{\Omega_E}(s, s'), \rho_\Omega(s, s')\right)] \Rightarrow \arg\min_{h_\Omega} \mathbb{E}_{s\sim\Omega}[\ell\left(h_{\Omega_E}(s'|s), h_\Omega(s'|s)\right)]\ . \tag{4}$$

However, SOIL still requires to learn a policy to interact with the MDP environment to match the state transition OM of the expert. This is achievable since we do not have to recover the expert policy π_E exactly but can learn any policy $\pi \in \Omega_E$ according to Eq. (4).

3.2 Policy Decoupling

To construct an unambiguous objective for SOIL, we define hyper-policy and solve the problem by finding the state transition predictor of the expert hyper-policy. Intuitively, this tells the agent the *target* that the expert will reach without informing any feasible *skill* that require the agent to learn itself. Therefore, to recover a $\pi \in \Omega_E$, we can construct an inverse dynamics such that

$$\pi = \underbrace{\mathcal{T}_\pi^{-1}}_{\text{Inverse dynamics}} \Big(\underbrace{\mathcal{T}(\pi_E)}_{\text{Expert state transition predictor}} \Big)\ . \tag{5}$$

Formally, the expert policy can be decoupled as

$$\pi_E(a|s) = \int_{s'} \mathcal{T}(s'|s, a)\pi_E(a|s)\, \mathrm{d}s' = \int_{s'} \frac{\rho_{\pi_E}(s, s')I_{\pi_E}(a|s, s')}{\rho_{\pi_E}(s)}\, \mathrm{d}s' = \int_{s'} h_{\pi_E}(s'|s)I_{\pi_E}(a|s, s')\, \mathrm{d}s'\ . \tag{6}$$

Notice that both the state transition predictor h and the inverse dynamics model I is policy dependent. Nevertheless, recall that the optimality in SOIL only requires us to recover $\pi \in \Omega_E$, we do not have to learn about I_{π_E} but just one feasible skill $I(a|s, s')$. Then a policy can be recovered by

$$\pi = \mathbb{E}_{\underbrace{s' \sim h_{\Omega_E}(s'|s)}_{\text{target}}} \Big[\underbrace{I(a|s, s')}_{\text{skill}} \Big]\ . \tag{7}$$

Here the inverse dynamics model I offers an arbitrary *skill* to reach the expected *target* state provided by the state transition predictor h. In fact, it does not depend on the hyper-policy Ω_E but a sampling policy $\pi_\mathcal{B}$ to construct $I = I_{\pi_\mathcal{B}}$. We only need a mild requirement for $\pi_\mathcal{B}$ that it covers the support of $\rho_{\Omega_E}(s, s')$ so that the learned I can provide a possible action to achieve the target state. In both experiments and theoretical analysis we show that this requirement alleviates the dependence on the inverse dynamics.

Furthermore, if the environment and the expert policy are both deterministic (which is usually the case in lots of real-world scenarios such as robotics), the state transition is a single-point distribution (or known as the Dirac delta function), and we can simply model h as a deterministic function. By decoupling the policy, which is a state-to-action mapping function, as a state-to-state mapping function (the transition predictor) and a state-pair-to-action mapping function (the inverse dynamics model), we can mimic the expert policy from state-only demonstrations by optimizing these two modules. The whole architecture is illustrated in Fig. 1.

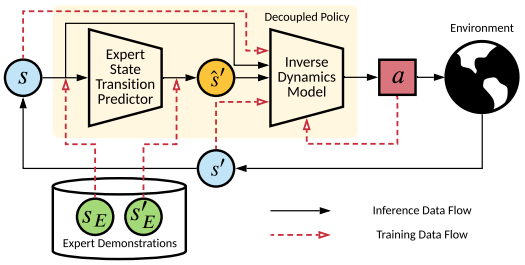


Figure 1: The architecture of Decoupled Policy Optimization (DPO), which consists of an expert state transition predictor (to plan where to go) followed by an inverse dynamics model (to decide how to reach).

State Transition Predictor. In practice, we construct a parameterized expert state transition predictor h_ψ which predicts the subsequent state of the expert taking the input as a current state $\hat{s}' = h_\psi(s)$.

The state transition predictor models the explicit information of the expert, and it can be learned from the demonstration data only. Thence, we implement Eq. (4) as a KL divergence minimization:

$$\min_{\psi} \mathbb{E}_{(s,s')\sim\Omega_E}[\mathrm{D}_{\mathrm{KL}}(h_{\Omega_E}(s'|s)\|h_\psi(s'|s))] \,, \tag{8}$$

which can be optimized in a supervised manner. Specifically, we sample state transitions (s, s') from the expert demonstrations $\mathcal{D}$ and optimize the L2 loss:

$$\mathcal{L}^h_\psi = \mathbb{E}_{(s,s')\sim\mathcal{D}}\left[\|s' - h_\psi(s)\|^2\right] \,. \tag{9}$$

Inverse Dynamics Model. Knowing where to go is not enough since the agent has to interact with the environment to reach the target. This can be achieved via an inverse dynamics model, which predicts the action given two consecutive states. Formally, let the ϕ-parameterized inverse dynamics model I_ϕ take input the state pair and predict the feasible action to achieve the state transition: $\hat{a} = I_\phi(s, s')$. Intuitively, we want the inverse dynamics to learn from possible transitions sampled by the agent. Recall that we only need the support of learned $I(a|s, s')$ covers the support of the expert state transition OM, from which we can infer at least one possible action. Hence, we can optimize the KL divergence between the inverse dynamics of a sampling policy $\pi_\mathcal{B}$ and I_ϕ:

$$\min_{\phi} \mathbb{E}_{(s,s')\sim\pi_\mathcal{B}}[\mathrm{D}_{\mathrm{KL}}(I_{\pi_\mathcal{B}}(a|s, s')\|I_\phi(a|s, s'))] \,, \tag{10}$$

and we can choose to optimize L2 loss in a supervised manner by sampling from the replay buffer $\mathcal{B}$:

$$\mathcal{L}^I_\phi = \mathbb{E}_{(s,a,s')\sim\mathcal{B}}\left[\|a - I_\phi(s, s')\|^2\right] \,. \tag{11}$$

In our implementation, both the state predictor and the inverse dynamics can be constructed as Gaussian distributions similar to a normal stochastic policy, thus encouraging exploration.

3.3 Tackling Compounding Error Challenges

In our formulation, we have decoupled the state-to-action mapping policy as a state-to-state mapping function and a state-pair-to-action mapping function. Unfortunately, the compounding error problem exists such that the agent cannot reach where it plans due to the fitting errors of these two parts.

Theorem 1 (Error Bound of DPO)**.** *Consider a deterministic environment whose dynamics transition function $\mathcal{T}(s, a)$ is deterministic and L-Lipschitz. Assume the ground-truth state transition $h_{\Omega_E}(s)$ is deterministic, and for each policy $\pi \in \Pi$, its inverse dynamics I_π is also deterministic and C-Lipschitz. Then for any state s, the distance between the desired state s'_E and reaching state s' sampled by the decoupled policy is bounded by*

$$\|s' - s'_E\| \leq LC\|h_{\Omega_E}(s) - h_\psi(s)\| + L\|I_{\pi_\mathcal{B}}(s, \hat{s}') - I_\phi(s, \hat{s}')\| \,, \tag{12}$$

where $\pi_\mathcal{B}$ is a sampling policy that covers the state transition support of the expert hyper-policy and $\hat{s}' = h_\psi(s)$ is the predicted next state.

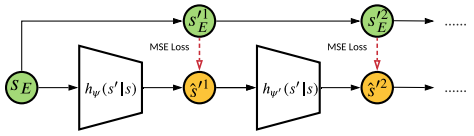


Figure 2: Multi-step optimization. Given an expert state s_E, h_ψ predicts the next possible state $\hat{s}'^1$, which is further fed to a target network $h_{\psi'}$ to predict the following sequence. The total loss computes the MSE loss along the state sequence.

The proof can be found in Appendix B, where we also induce a similar error bound for rollout with a state-to-action policy as BCO (Torabi et al., 2018) to show the advantage of the decoupled structure. From Theorem 1 we know that the compounding error can be enlarged due to each part's fitting error, where the first term corresponds to the error of predicted states and the second term indicates whether the agent can reach where it plans to. To alleviate the error, we further propose regularization on these two modules.

3.3.1 Regularization on Target Planning

One major problem is that the state transition predictor may suggest non-neighboring states instead of predicting one-step reachable states. To overcome this, we draw inspiration from Asadi et al. (Asadi et al., 2019) and Edwards et al. (Edwards et al., 2020), and regularize state transition predictor to prevent the model from predicting non-neighboring states via multi-step and cycle training style.

Multi-Step Optimization. We first explain the details of the multi-step optimization objective. This idea is motivated by Asadi et al. (Asadi et al., 2019), which optimizes a multi-step outcome by executing a sequence of actions in the dynamics model. Here we optimize the state sequence instead. As shown in Fig. 2, given an expert state s_E, h_ψ predicts the next possible state $\hat{s}'$ that the expert will reach; the predicted state is then fed into the predictor to output the predicted two-step state $\hat{s}''$. As such, the multi-step training loss is the L2 loss computed along the k-step outcome sequence:

$$\mathcal{L}_\psi^{h,\text{ms}} = \mathop{\mathbb{E}}_{(s,\{s_E'^i\}_{i=1}^k)\sim\mathcal{D}} \left[\|s_E'^1 - h_\psi(s)\|^2 + \sum_{i=2}^{k} \|s_E'^i - h_{\psi'}(s'^{i-1})\|^2 \right]. \tag{13}$$

Intuitively, such a regularization makes the state prediction $\hat{s}'$ close to the expert state distribution in order to make accurate long step predictions. It is worth noting that the gradient of the cascading state transition predictors should be dropped since we already have the true labels in the dataset, and therefore we should not require the cascade parts to be optimized using the predicted input. We use a target network $h_{\psi'}$ in practice.

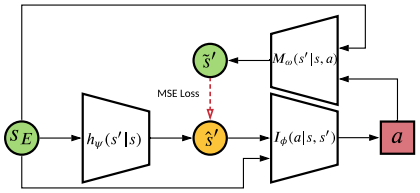


Figure 3: Cycle training style. Given an expert state s_E, $I_\phi(s, s')$ takes input the predicted state $\hat{s}'$ and s_E to get the execution action a, then an additional forward dynamics model M_ω is used to simulated one step rollout using (s_E, a) and get a forward next state $\tilde{s}'$. The total loss computes the MSE loss between the two predicted states.

Cycle Training Style. Another way to regularize the transition predictor's output to a neighboring state is to keep an additional function to ensure the cycle consistency, which is also an important technique in (Edwards et al., 2020). In particular, as illustrated in Fig. 3, given an expert state s_E, we take the predicted state $\hat{s}'$ and s_E into the inverse dynamics and get the action a, then we train an additional forward dynamics model M_ω to simulate one step rollout that takes the input (s_E, a) and gets a forward next state $\tilde{s}'$:

$$\begin{aligned} \mathcal{L}_\omega^M &= \mathbb{E}_{(s,a,s')\sim\mathcal{B}} \left[\|s' - M_\omega(s,a)\|^2 \right] \\ \mathcal{L}_\psi^{h,\text{cycle}} &= \mathbb{E}_{(s,s')\sim\mathcal{D}} \left[\|s' - h_\psi(s)\|^2 + \|h(s) - M_\omega(s, I_\phi(s,s'))\|^2 \right]. \end{aligned} \tag{14}$$

In other words, the cycle training scheme provides a regularization on h_ψ to make predictions consistent with the forward dynamics model.

3.3.2 Efficient Skills Learning via Decoupled Policy Gradient

In previous sections, we have mentioned that learning to reach a specific place requires the data-collecting policy to cover the support of the expert hyper-policy. This is easy to achieve on simple low-dimensional tasks, but may not be satisfied in high-dimensional continuous environments. To this end, we encourage the agent to approach those state transitions from the expert's hyper-policy Ω_E by minimizing the JS divergence of the state transition occupancy $\text{D}_{\text{JS}}\left(\rho_{\pi_E}(s,s'), \rho_\pi(s,s')\right)$. This can

be done by producing informative rewards via GAN-like methods (Ho & Ermon, 2016; Torabi et al., 2019a), and updating the decoupled policy with policy gradients (PG).

In detail, we construct a parameterized discriminator $D_\omega(s, s')$ to compute the reward $r(s, a) \triangleq r(s, s')$ as $\log D_\omega(s, s')$ and the decoupled policy served as the generator. In addition, since we decouple the policy as two parameterized modules, i.e., a state transition predictor and an inverse dynamics model, then by chain rule, the PG for the decoupled policy can be accomplished by

$$\begin{aligned} \nabla \mathcal{L}^\pi_{\phi,\psi} &= \mathbb{E}_\pi \left[Q(s, a) \nabla_{\phi,\psi} \log \pi_{\phi,\psi}(a|s) \right] \\ &= \mathbb{E}_\pi \left[Q(s, a) \int_{s'} \left(\nabla_\psi \log h_\psi(s'|s) + \nabla_\phi \log I_\phi(a|s, s') \right) \mathrm{d}s' \right] , \end{aligned} \tag{15}$$

where Q is the state-action value function estimated using the normal Bellman equation and proposed surrogate reward function; the first term is the gradient for updating the state transition predictor; and the second term is for the inverse dynamics model. Thus, the optimization for both the state transition predictor and the inverse dynamics can augment the supervised learning objectives with any PG-based learning algorithms (e.g., TRPO, PPO, SAC). In practice, the integration can be resolved via reparameterization tricks, and it will be easier for deterministic state transition that can be directly optimized end-to-end. As the training proceeds, we expect the agent sample more transition data around Ω_E, and thus the support of the sampling policy progressively covers the support of $\rho_{\Omega_E}(s, s')$. In experiments we will show that this benefit much for the performance when the task is complex.

3.4 Overall Algorithm

By combining the idea of generative adversarial training, we obtain our final algorithm, composed with three essential parts: the state transition predictor h used for predicting the possible future states sampled by the expert; the inverse dynamics model I used for inferring the possible actions conditioned on two adjacent states; and the discriminator D used for offering intermediate reward signals for training the decoupled policy $\pi = I(h)$. The overall objective of DPO is

$$\mathcal{L}^{\pi,h,I}_{\phi,\psi} = \lambda_G \mathcal{L}^\pi_{\phi,\psi} + \lambda_h \mathcal{L}^h_\psi + \lambda_I \mathcal{L}^I_\phi , \tag{16}$$

where λ_G, λ_h and λ_I are hyperparameters for trading off the training among each loss. In practice, we try a small set of variants for these parameters as shown in Appendix D.1.3, and we directly optimize $\mathcal{L}^\pi_{\phi,\psi}$ instead of iterative training the two modules independently. The detailed algorithm is summarized in Appendix A. Besides, it is worth noting that both the inverse dynamics model and the state transition predictor can be pre-trained, where we optimize $\mathcal{L}^h_\psi$ using the state-only demonstration and optimize $\mathcal{L}^I_\phi$ using samples collected by a randomized agent.

4 Related Work

Table 1: Comparison between different methods.

Method	Inverse Dynamics	State Predictor	Decoupled Policy	Task
BCO (Torabi et al., 2018)	✓	✗	✗	SOIL
GAIfO (Torabi et al., 2019a)	✗	✗	✗	SOIL
IDDM (Yang et al., 2019)	✗	✗	✗	SOIL
OPOLO (Zhu et al., 2020)	✓	✗	✗	SOIL
PID-GAIL (Huang et al., 2020)	✗	✗	✓	IL
QSS (Edwards et al., 2020)	✓	✓	✓	RL
SAIL (Liu et al., 2020)	✓	✓	✗	IL
DPO (Ours)	✓	✓	✓	SOIL

SOIL endows the agent with the ability to learn from expert states. Although lacking the expert decision information, most of the previous works still optimize a state-to-action mapping policy to match the expert state transition distribution. For example, Torabi et al. (2018) trained an inverse model to label the action information and applying behavioral cloning, while Torabi et al. (2019a) generalized GAIL to match the state transition distribution. Yang et al. (2019) analyzed the inverse dynamics mismatch in SOIL and introduced a mutual information term to narrow it, however, in our paper, we show that the mismatch is not the key to imitate the expert state sequence. Huang et al. (2020) applied SOIL on autonomous driving tasks by decoupling the policy into a neural decision module and a non-differentiable execution module in a hierarchical way.

Our work decouples the state-to-action policy into two modules. However, both the inverse dynamics model and the state transition predictor have been widely used by many previous works on RL and IL tasks. For instance, Torabi et al. (2018) and Guo et al. (2019) trained an inverse dynamics model to label the state-only demonstrations with inferred actions. Nair et al. (2017) proposed to match a human-specified image sequence of ropes manipulating with an inverse dynamics model. Pathak et al. (2018) also focused on image-based imitation and utilized a multi-step inverse dynamics model which

Table 2: Eventual performance against different methods on 6 easy-to-hard continuous control benchmarks. The means and the standard deviations are evaluated over more than 10 random seeds.

	InvertedPendulum	InvertedDoublePendulum	Hopper	Walker2d	HalfCheetah	Ant
Random	25.28 ± 5.53	78.28 ± 10.73	13.09 ± 0.10	7.07 ± 0.13	74.48 ± 12.39	713.59 ± 203.92
BCO	**1000.0 ± 0.00**	416.92 ± 141.56	1516.91 ± 524.86	270.45 ± 33.22	6.56 ± 151.49	456.45 ± 179.76
GAIfO	**1000.0 ± 0.00**	**8589.46 ± 1391.82**	3068.10 ± 24.90	3864.03 ± 326.64	8918.66 ± 1031.41	4879.13 ± 897.46
GAIfO-DP	**947.27 ± 110.24**	**8674.24 ± 1318.04**	3030.70 ± 139.85	4008.14 ± 200.79	8710.29 ± 853.13	**5502.25 ± 214.34**
DPO (w/o PG)	**1000.00 ± 0.00**	3933.33 ± 3414.51	713.17 ± 369.05	310.53 ± 68.27	-442.55 ± 120.23	-383.12 ± 198.18
DPO (w PG)	**1000.00 ± 0.00**	**8587.96 ± 1394.29**	**3163.74 ± 64.46**	**4395.21 ± 216.96**	**10522.08 ± 394.44**	5413.03 ± 161.97
Expert (SAC)	1000.00 ± 0.00	9358.87 ± 0.10	3402.94 ± 446.48	5639.32 ± 29.97	13711.64 ± 111.47	5404.55 ± 1520.49

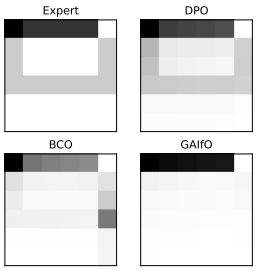
is regularized by cycle consistency. However, their method can be classified as supervised learning and requires a pre-collected exploration dataset rather than only small numbers of demonstrations considered in our paper. Kimura et al. (2018) utilized a state transition predictor to fit the state transition probability in the expert data, which is further used to compute a predefined reward function. Liu et al. (2020) constructed a policy prior using the inverse dynamics and the state transition predictor, but the policy prior was only used for regularizing the policy network. However, as shown in this paper, the policy can be exactly decoupled as these two parts, which can be uniformly optimized through policy gradient without keeping an extra policy. Edwards et al. (2020) estimated $Q(s, s')$ for RL tasks which employs a similar policy form as Eq. (6) and updates the state transition predictor through a deterministic policy gradient similar to DDPG (Lillicrap et al., 2016). To sort out the difference between these methods and ours, we summarize the key factors in Tab. 1.

5 Experiments

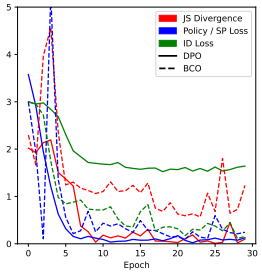
We conduct four sets of experiments to investigate the following research questions:

RQ1 Is decoupled learning structure superior than state-to-action structure on SOIL tasks?
RQ2 Does DPO achieve higher efficiency or better performance than baselines on SOIL tasks?
RQ3 Can agent reach where it plans with less compounding error?
RQ4 How can DPO be applied on real-world data?

To answer RQ1, we conduct toy experiments with a simple 2D grid world environment and compare both qualitative and quantitative imitation results. Regarding RQ2, we empirically evaluate DPO on easy-to-hard continuous control benchmarking tasks. And for RQ3, we evaluate the difference between the predicted states that the agent plans to reach and the actually reached consecutive states in the environment for the proposed regularization. Finally, we explain a case on imitating real-world traffic surveillance recordings in a simulated environment to investigate RQ4, which shows the potential of using real-world data for human behavior simulation. Due to the space limit, we leave experiment details, additional results and ablation studies in Appendix D.



(a) Rollout density.



(b) Loss curves.

Figure 4: Toy example.

5.1 Understanding the Decoupled Structure

In this paper we design decoupled policy optimization (DPO) to perform SOIL tasks, and in previous sections we propose that the key technical contribution of DPO is the decoupled structure of policy that models the explicit state transition information and the latent action information from demonstrations, which solves the ambiguity and enhances the learning efficiency. Therefore, in this set of experiments, we aim to demonstrate how DPO is superior than state-to-action policy methods (RQ1). We first generate expert demonstrations in a 2D 6×6 grid world environment, in which the agent starts at the upper left corner and aims to reach the upper right corner. In each grid the agent has $k \times 4$ actions, which means that the agent has k possible actions to reach the neighboring block and in our experiment we choose $k = 5$ to enlarge the action space.

The density of the expert trajectories and the trajectories sampled by different methods are shown in Fig. 4(a). We show that both BCO and GAIfO have troubles in directly learning the implicit action from state-only behaviors. Notably, GAIfO only imitates the major trajectory and omit the other choice and BCO also stucks in the middle right. By contrast, DPO recovers the expert demonstrations

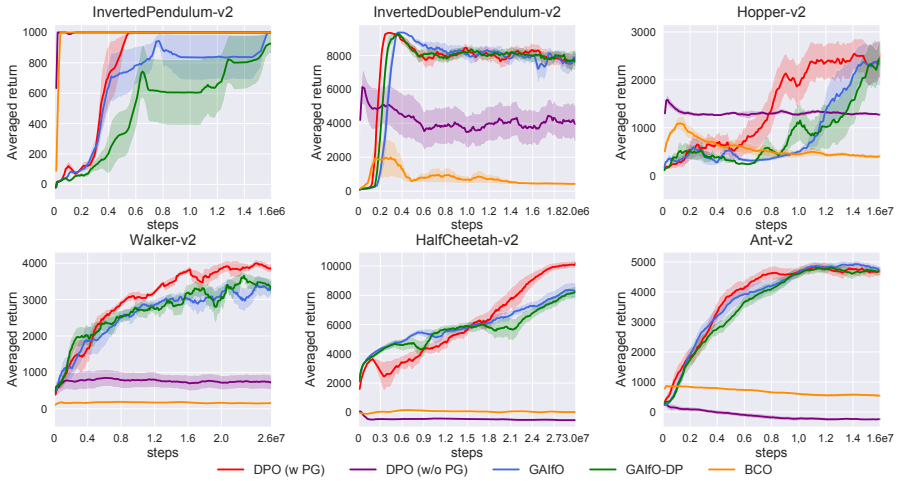


Figure 5: Learning curves on 6 easy-to-hard continuous control benchmarks, where the solid line and the shade represent the mean and the standard deviation of the averaged return over more than 5 random seeds. We pre-train BCO and DPO for 50k steps and show it in figures.

much better, benefiting from the decoupled structure that first determining the target and then taking the action to achieve it. To further illustrate the learning efficiency advantage of DPO, we illustrate the JS divergence curves of DPO and BCO during training in Fig. 4(b). Besides, we show the policy loss for BCO, the state predictor (SP) loss for DPO, and the inverse dynamics (ID) loss for both methods. Except that the JS divergence of DPO decreases more quickly than BCO, it is also observable that DPO relies less on the inverse dynamics than BCO, since the inverse dynamics loss of DPO converges to a higher level. We further provide a in-depth theoretic and experimental analysis of the dependence on inverse dynamics with BCO and DPO in Appendix B and Appendix D.

5.2 Comparative Evaluations

We compare the qualitative results of DPO against other baseline methods on easy-to-hard continuous control benchmarking environments (RQ2), including InvertedPendulum, InvertedDoublePendulum, Hopper, Walker2d, HalfCheetah and Ant. In each environment, besides GAIfO and BCO, we also evaluate GAIfO with decoupled policy (denoted as GAIfO-DP). For DPO we compare the reward augmented version of DPO (denoted as DPO w PG)[1] with the supervised learning version of DPO, *i.e.*, $\lambda_G = 0$ (denoted as DPO w/o PG). For fairness, we re-implement all the algorithms and adopt Soft Actor-Critic (SAC) (Haarnoja et al., 2018) as the RL learning algorithm for GAIfO and DPO. For all environments, we first train an SAC agent to collect 4 state-only expert trajectories and then train agents with such data. All algorithms are evaluated by a deterministic policy. The eventual results are summarized in Tab. 2, and the learning curves are shown in Fig. 5. It is worth noting that for DPO, we choose the best performance among the experiments that use multi-step or cycle regularization, and we put the full experiment results in Appendix D.

One can easily observe that on simple environments, BCO is able to achieve a good performance, and GAIfO also does well on harder tasks. Even so, DPO can still gain the best or comparable performance against its counterparts. Particularly, without augmented reward, DPO is able to beat BCO with the higher sample efficiency on simple tasks like Pendulums. By contrast, on higher-dimensional tasks such as Walker2d, HalfCheetah and Ant, it is difficult to construct accurate inverse dynamics that covers the support of the expert hyper-policy from scratch. However, thanks to the decoupled structure, DPO combines generative adversarial policy gradients with the supervision, and finally recovers a good policy from the expert hyper-policy. This is particularly evident on HalfCheetah where DPO behaves poorly at the beginning but improves fast as the training proceeds. Besides, as illustrated in Fig. 5, DPO owns better sample efficiency in most of the environments, but the improvements are limited on the hardest tasks. We think that this may be due to larger state spaces (111 dimensions for Ant) that makes it difficult to recover the state predictor or an inverse dynamics model. In all experiments, GAIfO-DP achieves similar results as GAIfO, indicating that the network structure does not count much for the performance.

[1]Without ambiguity we simply denote DPO for this version of algorithm in the following sections.

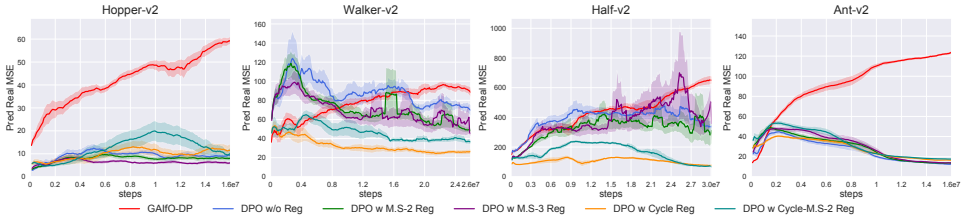


Figure 6: Compounding error of the predicted consecutive states and the real states the agent reaches when rollout in the environments.

5.3 Compounding Error Reduction

In this section, we aim to study whether the agent can reach the target as it plans (RQ3) and see if the supervision signal is meaningful. Therefore, we analyze the distance of the reaching states and the predicted consecutive states, and draw the mean square error (MSE) along the training procedure in Fig. 6. We compare our regularization including multi-step optimization (denoted as M.S.-k, where k is the number of rollout steps) and cycle training style (denoted as Cycle). Note that DPO needs at least 1-step rollout for training the state transition predictor. From the figure, we see the supervision does regularize the state prediction to be meaningful compared with GAIfO-DP, although the agent still has gaps to get to where it plans to. Combining regularization can always achieve lower compounding error, and the cycle training is effective in most of the environments. In Appendix D.6, we further illustrate the correlation between the final performance and the distance.

5.4 Learn to Drive from Real-World Traffic Data

The rapid development of autonomous driving has brought a lot of demand for simulating and training an RL agent in the simulator, which requires realistic interactions with various social vehicles (Zhou et al., 2020). However, driver's detailed actions are not easily to obtain yet we adopt SOIL from a traffic surveillance recording dataset (NGSIM I-80 (Halkias & Colyar, 2006)) that contains kinds of recorded human driving trajectories. We wish to further examine the potential of DPO for decreasing the gap between the real world and simulation (RQ4). We utilize the simulator provided by Henaff et al. (Henaff et al., 2019) as our simulation platform and learn to imitate real-world driving behaviors. We compare DPO against GAIfO and BCO, and choose *Success Rate*, *Mean Distance* and *KL Divergence* as evaluation metrics. Specifically, *Success Rate* is the percentage of driving across the entire area without crashing into other vehicles or driving off the road, *Mean Distance* is the distance traveled before the episode ends, and *KL Divergence* measures the position distribution distance between the expert and the agent.

Table 3: Performance on NGSIM I-80 driving task over 5 random seeds.

Method	Success Rate (%)	Mean Distance (m)	KL Divergence
BCO	27.4 ± 1.1	129.8 ± 2.0	24.4 ± 2.2
GAIfO	77.5 ± 0.8	188.3 ± 1.1	11.5 ± 3.9
DPO	$\mathbf{80.3 \pm 0.5}$	$\mathbf{192.7 \pm 0.6}$	$\mathbf{9.5 \pm 1.8}$
Expert	100	210.0	0

As shown in Tab. 3, DPO outperforms baseline methods in all three metrics with higher stability. The decoupled policy allows the state predictor to focus on matching the distribution of expert trajectories, thus achieving smaller deviations from the expert position distribution. Furthermore, since the policy gradient can be computed with non-differentiable inverse dynamics, we can generate stable action sequences (Huang et al., 2017; 2020) by replacing the inverse dynamics model with classical controllers, which can be generalized to realistic applications.

6 Conclusion

In this paper, we revisit the optimality the ambiguity problem in state-only imitation learning, and accordingly propose Decoupled Policy Optimization (DPO), which splits the state-to-action mapping policy into a state-to-state mapping state transition predictor and a state-pair-to-action mapping inverse dynamics model. Furthermore, we employ regularization and generative adversarial methods to mitigate the compounding error caused by the decoupled modules. The flexibility of the decoupled architecture allows a wide range of interesting future works, such as involving high-dimensional visual inputs, learning specific skills with shared state transition and multi-task target learning with shared pre-trained skills.

REFERENCES

Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004)*, 2004.

Kavosh Asadi, Dipendra Misra, Seungchan Kim, and Michel L Littman. Combating the compounding-error problem with a multi-step model. *arXiv preprint arXiv:1905.13320*, 2019.

Ashley D. Edwards, Himanshu Sahni, Rosanne Liu, Jane Hung, Ankit Jain, Rui Wang, Adrien Ecoffet, Thomas Miconi, Charles Isbell, and Jason Yosinski. Estimating q(s,s') with deep deterministic dynamics gradients. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, 2020.

Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International Conference on Machine Learning*, pp. 49–58, 2016.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adverserial inverse reinforcement learning. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

Xiaoxiao Guo, Shiyu Chang, Mo Yu, Gerald Tesauro, and Murray Campbell. Hybrid reinforcement learning with expert state sequences. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pp. 3739–3746, 2019.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, pp. 1856–1865, 2018.

John Halkias and James Colyar. Next generation simulation fact sheet. *US Department of Transportation: Federal Highway Administration*, 2006.

Mikael Henaff, Alfredo Canziani, and Yann LeCun. Model-predictive policy learning with uncertainty regularization for driving in dense traffic. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems 29*, pp. 4565–4573, 2016.

Junning Huang, Sirui Xie, Jiankai Sun, Qiurui Ma, Chunxiao Liu, Dahua Lin, and Bolei Zhou. Learning a decision module by imitating driver's control behaviors. In *Proceedings of the Conference on Robot Learning (CoRL) 2020*, 2020.

Sandy H. Huang, Nicolas Papernot, Ian J. Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

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

Daiki Kimura, Subhajit Chaudhury, Ryuki Tachibana, and Sakyasingha Dasgupta. Internal model from observations for reward shaping. *arXiv preprint arXiv:1806.01267*, 2018.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016,*, 2016.

Fangchen Liu, Zhan Ling, Tongzhou Mu, and Hao Su. State alignment-based imitation learning. In *8th International Conference on Learning Representations, ICLR 2020*, 2020.

Minghuan Liu, Tairan He, Minkai Xu, and Weinan Zhang. Energy-based imitation learning. In *20th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2021*, 2021.

Leland McInnes and John Healy. UMAP: uniform manifold approximation and projection for dimension reduction. *CoRR*, abs/1802.03426, 2018. URL `http://arxiv.org/abs/1802.03426`.

Ashvin Nair, Dian Chen, Pulkit Agrawal, Phillip Isola, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017*, pp. 2146–2153, 2017.

Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal, Dian Chen, Yide Shentu, Evan Shelhamer, Jitendra Malik, Alexei A Efros, and Trevor Darrell. Zero-shot visual imitation. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 2050–2053, 2018.

Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, pp. 832–837, 1956.

Umar Syed, Michael H. Bowling, and Robert E. Schapire. Apprenticeship learning using linear programming. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008)*, pp. 1032–1039, 2008.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018*, pp. 4950–4957, 2018.

Faraz Torabi, Garrett Warnell, and Peter Stone. Adversarial imitation learning from state-only demonstrations. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19*, pp. 2229–2231, 2019a.

Faraz Torabi, Garrett Warnell, and Peter Stone. Recent advances in imitation learning from observation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019*, pp. 6325–6331, 2019b.

Chao Yang, Xiaojian Ma, Wenbing Huang, Fuchun Sun, Huaping Liu, Junzhou Huang, and Chuang Gan. Imitation learning from observations by minimizing inverse dynamics disagreement. In *Advances in Neural Information Processing Systems 32*, pp. 239–249, 2019.

Ming Zhou, Jun Luo, Julian Villela, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, et al. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. In *Conference on Robot Learning*, 2020.

Zhuangdi Zhu, Kaixiang Lin, Bo Dai, and Jiayu Zhou. Off-policy imitation learning from observations. In *Advances in Neural Information Processing Systems 33*, 2020.

Appendices

A ALGORITHM

Algorithm 1 Decoupled Policy Optimization

Input: State-only expert demonstration data $\mathcal{D} = \{(s_i)\}_{i=1}^{N}$, empty replay buffer $\mathcal{B}$, randomly initialized discriminator model D_ϕ, state transition predictor h_ψ and parameterized inverse dynamics model I_ϕ;
for $k = 0, 1, 2, \cdots$ **do** ▷ Pre-training stage
  Collect trajectories $\{(s, a, s', r, \text{done})\}$ using a random initialized policy $\pi = I_\phi(h_\psi)$ and store in $\mathcal{B}$
  Sample $(s, a, s') \sim \mathcal{B}$ and update ϕ by $\mathcal{L}_\phi(I)$
  Sample $(s, s') \sim \mathcal{D}$ and update ψ by $\mathcal{L}_\psi^h$
end for
for $k = 0, 1, 2, \cdots$ **do** ▷ Online training stage
  Collect trajectories $\{(s, a, s', r, \text{done})\}$ using current policy $\pi = I_\phi(h_\psi)$ and store in $\mathcal{B}$
  Sample $(s, a, s') \sim \mathcal{B}, (s, s') \sim \mathcal{D}$
  Update the discriminator D_ω with the loss:

$$\mathcal{L}_\omega^D = -\mathbb{E}_{(s,s')\sim\mathcal{B}}[\log D_\omega(s, s')] - \mathbb{E}_{(s,s')\sim\mathcal{D}}[\log(1 - D_\omega(s, s'))] , \tag{17}$$

  Update ϕ, ψ by $\mathcal{L}_{\phi,\psi}^{h,I}$
end for

B PROOFS

In our proofs we will work in finite state and action spaces $\mathcal{S}$ and $\mathcal{A}$ to avoid technical machinery out of the scope of this paper.

Proposition 1. *Suppose Π is the policy space and $\mathcal{P}$ is a valid set of state transition OMs such that $\mathcal{P} = \{\rho : \rho \geq 0$ and $\exists \pi \in \Pi$, s.t. $\rho(s, s') = \rho_0(s) \int_a \pi(a|s)\mathcal{T}(s'|s, a)\,\mathrm{d}a + \int_{s'',a} \pi(a|s)\mathcal{T}(s'|s, a)\rho(s'', s)\,\mathrm{d}s''\,\mathrm{d}a\}$, then a policy $\pi \in \Pi$ corresponds to one state transition OM $\rho_\pi \in \mathcal{P}$. However, under the action-redundant assumption about the dynamics $\mathcal{T}$, a state transition OM $\rho \in \mathcal{P}$ can correspond to more than one policy in Π.*

Proof. We first provide the proof for the one-to-one correspondence between marginal distribution $\sum_a \pi(a|s)\mathcal{T}(s'|s, a)$ and state transition OM $\rho(s, s') \in \mathcal{P}$.

For a given policy π, by definition of state transition OM, we have

$$\begin{aligned} \rho_\pi(s, s') &= \sum_a \mathcal{T}(s'|s, a)\rho_\pi(s, a) \\ &= \sum_a \pi(a|s)\mathcal{T}(s'|s, a) \sum_{t=0}^{\infty} \gamma^t P(s_t = s|\pi) . \end{aligned} \tag{18}$$

For all t greater than or equal to 1, we have

$$P(s_t = s|\pi) = \sum_{s''} P(s_{t-1} = s'', s_t = s|\pi) . \tag{19}$$

Take Eq. (19) into Eq. (18), we have

$$\begin{aligned}\rho_\pi(s,s') &= P(s_0=s,s_1=s') + \sum_a \pi(a|s)\mathcal{T}(s'|s,a)\sum_{t=1}^{\infty}\gamma^t \sum_{s''} P(s_{t-1}=s'', s_t=s|\pi) \\ &= P(s_0=s,s_1=s') + \gamma\sum_a \pi(a|s)\mathcal{T}(s'|s,a)\sum_{s''}\sum_{t=0}^{\infty}\gamma^t P(s_t=s'', s_{t+1}=s|\pi) \\ &= P(s_0=s,s_1=s') + \gamma\sum_a \pi(a|s)\mathcal{T}(s'|s,a)\sum_{s''}\rho_\pi(s'',s) \\ &= \rho_0(s)\sum_a \pi(a|s)\mathcal{T}(s'|s,a) + \gamma\sum_{s'',a}\pi(a|s)\mathcal{T}(s'|s,a)\rho_\pi(s'',s)\end{aligned} \tag{20}$$

Consider the following equation of variable ρ:

$$\rho(s,s') = \rho_0(s)\sum_a \pi(a|s)\mathcal{T}(s'|s,a) + \gamma\sum_{s'',a}\pi(a|s)\mathcal{T}(s'|s,a)\rho(s'',s)\ . \tag{21}$$

According to Eq. (20), ρ_π is a solution of Eq. (21). Now we proceed to prove ρ_π as the unique solution of Eq. (21).

Define the matrix

$$A_{(ss',s''s)} \triangleq \begin{cases} 1-\gamma\sum_a \pi(a|s)\mathcal{T}(s'|s,a) & \text{if } (s,s')=(s'',s) \\ -\gamma\sum_a \pi(a|s)\mathcal{T}(s'|s,a) & \text{otherwise .}\end{cases}$$

Note that A is a two-dimensional matrix indexed by state transition pairs. Also define the vector

$$b_{s,s'} \triangleq \rho_0(s)\sum_a \pi(a|s)\mathcal{T}(s'|s,a)\ .$$

We can rewrite Eq. (21) equivalently as

$$A\rho = b\ . \tag{22}$$

Since $\sum_{s',a}\pi(a|s)\mathcal{T}(s'|s,a)=1$ and $\gamma<1$, for all (s'',s), we have

$$\begin{aligned}&\sum_{s,s'}\gamma\sum_a \pi(a|s)\mathcal{T}(s'|s,a) = \gamma < 1 \\ \Rightarrow\ & 1-\gamma\sum_a \pi(a|s'')\mathcal{T}(s|s'',a) > \sum_{(s,s')\neq(s'',s)}\gamma\sum_a \pi(a|s)\mathcal{T}(s'|s,a) \\ \Rightarrow\ & \left|A_{(s''s,s''s)}\right| \geq \sum_{(s,s')\neq(s'',s)}\left|A_{(ss',s''s)}\right|\ .\end{aligned}$$

Therefore, we have proven A as column-wise strictly diagonally dominant, which implies that A is non-singular, so Eq. (21) has at most one solution. Since for all ρ in $\mathcal{P}$, it must satisfy the constraint Eq. (21), which means that for any marginal distribution $\sum_a \pi(a|s)\mathcal{T}(s'|s,a)$, there is only one corresponding ρ in $\mathcal{P}$.

Now, we proceed to prove that for every ρ in $\mathcal{P}$, there is only one corresponding marginal distribution $\sum_a \pi(a|s)\mathcal{T}(s'|s,a)$ such that $\rho_\pi=\rho$. By definition of $\mathcal{P}$, ρ is the solution of Eq. (21) for some policy π. By rewriting Eq. (21), the marginal distribution can be written in the form of a function expression of ρ as

$$\sum_a \pi(a|s)\mathcal{T}(s'|s,a) = \frac{\rho(s,s')}{\rho_0(s)+\gamma\sum_{s''}\rho(s'',s)}\ . \tag{23}$$

This means every $\rho \in \mathcal{P}$ only corresponds to one marginal distribution $\sum_a \pi(a|s)\mathcal{T}(s'|s,a)$. As we discussed before, ρ is the state transition OM of π, i.e., $\rho = \rho_\pi$.

By establishing the one-to-one correspondence between the marginal distribution $\sum_a \pi(a|s)\mathcal{T}(s'|s,a)$ and state transition OM $\rho \in \mathcal{P}$, we can alternatively study the correspondence between the marginal distribution and policy. Obviously, one policy can only corresponds to one marginal distribution. We now prove that if the dynamics $\mathcal{T}$ has redundant actions, one marginal distribution can correspond to more than one policy in π.

We prove the statement by counterexample construction. If the dynamics $\mathcal{T}$ has redundant actions, there exist $s_m \in \mathcal{S}$, $a_n \in \mathcal{A}$ and distribution p defined on $\mathcal{A} \setminus \{a_n\}$ such that $\sum_{a \in \mathcal{A}\setminus\{a_n\}} p(a)\mathcal{T}(s'|s_m,a) = \mathcal{T}(s'|s_m,a_n)$. Consider two policy π_0 and π_1 such that

$$\begin{cases} \pi_0(a|s) = \pi_1(a|s) & \text{if } s \neq s_m \\ \pi_0(a_n|s_m) = 1 & \\ \pi_0(a|s_m) = 0 & \text{if } a \neq a_n \\ \pi_1(a_n|s_m) = 0 & \\ \pi_1(a|s_m) = p(a) & \text{if } a \neq a_n \ . \end{cases} \tag{24}$$

From Eq. (24), we know that π_0 and π_1 are two different policies. However, they share the same marginal distribution $\sum_a \pi(a|s)\mathcal{T}(s'|s,a)$. To justify this, we first consider the case when s equals to s_m, where we have

$$\begin{aligned} \sum_a \pi_0(a|s_m)\mathcal{T}(s'|s_i,a) &= \pi_0(a_n|s_m)\mathcal{T}(s'|s_m,a_n) + \sum_{a\in\mathcal{A}\setminus\{a_n\}} \pi_0(a|s_m)\mathcal{T}(s'|s_m,a) \\ &= \mathcal{T}(s'|s_m,a_n) \\ &= \sum_{a\in\mathcal{A}\setminus\{a_n\}} \pi_1(a|s_m)\mathcal{T}(s'|s_m,a) \\ &= \sum_{a\in\mathcal{A}\setminus\{a_n\}} \pi_1(a|s_m)\mathcal{T}(s'|s_m,a) + \pi_1(a_n|s_m)\mathcal{T}(s'|s_m,a_n) \\ &= \sum_a \pi_1(a|s_m)\mathcal{T}(s'|s_m,a) \ . \end{aligned} \tag{25}$$

When s does not equal to s_m, the equality holds trivially, since the action selection probability of π_0 and π_1 defined on these states are exactly the same. Thus, one marginal distribution can correspond to more than one policy in π when there are redundant actions.

□

Proposition 2. *Suppose the state transition predictor h_Ω is defined as in Eq. (3) and $\Gamma = \{h_\Omega : \Omega \in \Lambda\}$ is a valid set of the state transition predictors, $\mathcal{P}$ is a valid set of the state-transition OMs defined as in Proposition 1, then a state transition predictor $h_\Omega \in \Gamma$ corresponds to one state transition OM $\rho_\Omega \in \mathcal{P}$; and a state transition OM $\rho \in \mathcal{P}$ only corresponds to one hyper-policy state transition predictor such that $h_\rho = \rho(s,s')/\int_{s'} \rho(s,s')\,\mathrm{d}s'$.*

Proof. During the proof of Proposition 1, we have an intermediate result that there is one-to-one correspondence between the marginal distribution $\sum_a \pi(a|s)\mathcal{T}(s'|s,a)$ and state transition OM $\rho \in \mathcal{P}$. Since the definition of state transition predictor is exactly $h_\Omega(s'|s) = \sum_a \pi(a|s)\mathcal{T}(s'|s,a)$ ($\forall \pi \in \Omega$), the one-to-one correspondence naturally holds between state transition predictor $h(s'|s)$ and state transition OM $\rho \in \mathcal{P}$.

□

Theorem 1 (Error Bound of DPO)**.** *Consider a deterministic environment whose transition function $\mathcal{T}(s,a)$ is deterministic and L-Lipschitz. Assume the ground-truth state transition $h_{\Omega_E}(s)$ is deterministic, and for each policy $\pi \in \Pi$, its inverse dynamics I_π is also deterministic and C-Lipschitz.*

Then for any state s, the distance between the desired state s'_E and reaching state s' sampled by the decoupled policy is bounded by:

$$\|s' - s'_E\| \leq LC\|h_{\Omega_E}(s) - h_\psi(s)\| + L\|I_{\tilde{\pi}}(s, \hat{s}') - I_\phi(s, \hat{s}')\| \,, \tag{26}$$

where $\tilde{\pi}$ is a sampling policy that covers the state transition support of the expert hyper-policy and $\hat{s}' = h_\psi(s)$ is the predicted consecutive state.

Proof. Given a state s, the expert takes a step in a deterministic environment and get s'. We assume that the expert Ω_E can use any feasible policy $\tilde{\pi}$ that covers the support of Ω_E to reach s:

$$s'_E = \mathcal{T}(s, I_{\tilde{\pi}}(s, h_\Omega(s))) \tag{27}$$

Similarly, using decoupled policy, the agent predict $\hat{s}' = h_\psi(s)$ and infer an executing action by an inverse dynamics model $a = I_\phi(s, s')$, which is learned from the sampling policy $\tilde{\pi}$. Denote the reaching state of the agent as s':

$$s' = \mathcal{T}(s, I_\phi(s, h_\psi(s))) \tag{28}$$

Therefore, the distance between s' and s'_E is:

$$\|s' - s'_E\| = \|\mathcal{T}(s, I_{\tilde{\pi}}(s, h_\Omega(s))) - \mathcal{T}(s, I_\phi(s, h_\psi(s)))\|$$

Lets consider the deterministic transition on s is a function of a such that $s' = \mathcal{T}^s(a)$, then we continue the deviation:

$$\begin{aligned}
\|s' - s'_E\| &\leq \|\mathcal{T}^s(I_{\tilde{\pi}}(s, h_\Omega(s))) - \mathcal{T}^s(I_\phi(s, h_\psi(s)))\| \\
&\leq L\|I_{\tilde{\pi}}(s, h_\Omega(s))) - I_\phi(s, h_\psi(s))\| \\
&\leq L\|I_{\tilde{\pi}}(s, h_\Omega(s))) - I_\phi(s, h_\psi(s))\| \\
&\leq L\|I_{\tilde{\pi}}(s, h_\Omega(s))) - I_{\tilde{\pi}}(s, h_\psi(s))) + I_{\tilde{\pi}}(s, h_\psi(s))) - I_\phi(s, h_\psi(s))\|
\end{aligned}$$

Similarly we also take the inverse transition on s is a function of s' such that $a = I^s(s')$, then we have that:

$$\begin{aligned}
\|s' - s'_E\| &\leq L\|I^s_{\tilde{\pi}}(h_\Omega(s))) - I^s_{\tilde{\pi}}(h_\psi(s))) \\
&\quad + I^s_{\tilde{\pi}}(h_\psi(s))) - I^s_\phi(h_\psi(s))\| \\
&\leq L\|I^s_{\tilde{\pi}}(h_\Omega(s))) - I^s_{\tilde{\pi}}(h_\psi(s)))\| + L\|I^s_{\tilde{\pi}}(h_\psi(s))) - I^s_\phi(h_\psi(s))\| \\
&\leq LC\|h_\Omega(s)) - h_\psi(s))\| + L\|I^s_{\tilde{\pi}}(\hat{s}') - I^s_\phi(\hat{s}')\| \,.
\end{aligned} \tag{29}$$

□

Theorem 2 (Error Bound of BCO)**.** *Consider a deterministic environment whose transition function $\mathcal{T}(s, a)$ is deterministic and L-Lipschitz, and a parameterized policy $\pi_\psi(a|s)$ that learns from the label provided by a parameterized inverse dynamics model I_ϕ. Then for any state s, the distance between the desired state s'_E and reaching state s' sampled by a state-to-action policy as BCO (Torabi et al., 2018) is bounded by:*

$$\begin{aligned}
\|s' - s'_E\| \leq L \left\| \pi_\psi(a|s) - \int_{s'^*} p_{\pi_E}(s'^*|s) I_\phi(a|s, s'^*) \, \mathrm{d}s'^* \right\| \\
+ L \left\| \int_{s'^*} p_{\pi_E}(s'^*|s) I_{\tilde{\pi}}(a|s, s'^*)) - p_{\pi_E}(s'^*|s) I_\phi(a|s, s'^*) \, \mathrm{d}s'^* \right\| \,,
\end{aligned} \tag{30}$$

where $\tilde{\pi} \in \omega_E$ is a policy instance of the expert hyper-policy ω_E such that $\mathcal{T}(s, \tilde{\pi}(s)) = s'_E$.

Proof.

$$
\begin{aligned}
\|s' - s'_E\| &= \|\mathcal{T}(s, \pi_\psi(s)) - \mathcal{T}(s, \tilde{\pi}(s))\| \\
&= \|\mathcal{T}^s(\pi_\psi(s)) - \mathcal{T}^s(\tilde{\pi}(s))\| \\
&\leq L\|\tilde{\pi}(a|s) - \pi_\psi(a|s)\| \\
&= L \Bigg\| \pi_\psi(a|s) - \int_{s'^*} p_{\pi_E}(s'^*|s) I_\phi(a|s, s'^*)\, \mathrm{d}s'^* \\
&\quad + \int_{s'^*} p_{\pi_E}(s'^*|s) I_\phi(a|s, s'^*)\, \mathrm{d}s'^* - \int_{s'^*} p_{\pi_E}(s'^*|s) I_{\tilde{\pi}}(a|s, s'^*))\, \mathrm{d}s'^* \Bigg\| \\
&\leq L \left\| \pi_\psi(a|s) - \int_{s'^*} p_{\pi_E}(s'^*|s) I_\phi(a|s, s'^*)\, \mathrm{d}s'^* \right\| \\
&\quad + L \left\| \int_{s'^*} p_{\pi_E}(s'^*|s) I_{\tilde{\pi}}(a|s, s'^*)) - p_{\pi_E}(s'^*|s) I_\phi(a|s, s'^*)\, \mathrm{d}s'^* \right\|
\end{aligned} \tag{31}
$$

□

An intuitive explanation for the bound is that BCO (Torabi et al., 2018) first seeks to recover a policy that shares the same hyper-policy with π_E via learning an inverse dynamics model and then try to conduct behavior cloning. Therefore the errors comes from the reconstruction error of $\tilde{\pi}$ using I_ϕ (the second term) and the fitting error of behavior cloning (the first term).

By comparing Theorem 1 and Theorem 2, it is observed that for reaching each state, BCO requires a good inverse dynamics model over the state space to construct $\tilde{\pi}$ and then conduct imitation learning to $\tilde{\pi}$, while DPO only requires to learn a good inverse dynamics model on the predicted state and directly construct $\tilde{\pi}$ without the second behavior cloning step. This intuition meets our evaluation results in experiment Section 5.1.

C State Transition Occupancy Measure Matching

In the literature of inverse reinforcement learning (Syed et al., 2008; Abbeel & Ng, 2004; Finn et al., 2016), the ambiguity comes from the multiple answer for matching the feature of the expert demonstrations. A feasible solution to this problem is the maximum entropy principle that models the expert data with probability models. In a recent work (Liu et al., 2021), the authors show that state-action OM matching corresponds to maximum entropy reinforcement learning. Specifically, consider modeling the state-action OM with the Boltzmann distribution as $\rho_\pi(s, a) \propto \exp r(s, a)$, then we have that:

$$
\begin{aligned}
\mathrm{D}_{\mathrm{KL}}(\rho_\pi(s,a) \| \rho_{\pi_E}(s,a)) &= \sum_{s,a} \rho_\pi(s,a) \log \frac{\rho_\pi(s,a)}{\rho_{\pi_E}(s,a)} \\
&= \sum_{s,a} \rho_\pi(s,a) \left(-r(s,a) + \log \rho_\pi(s,a)\right) + \text{const} \\
&= \mathbb{E}_\pi \left[-r(s,a)\right] + \sum_{s,a} \rho_\pi(s,a) \log \rho_\pi(s,a) + \text{const} \\
&= \mathbb{E}_\pi \left[-r(s,a)\right] + \sum_{s,a} \rho_\pi(s,a) \log \left(\rho_\pi(s)\pi(a|s)\right) + \text{const} \\
&= \mathbb{E}_\pi \left[-r(s,a)\right] - H(\pi(a|s)) - H(\rho_\pi(s)) + \text{const} \\
&\leq \mathbb{E}_\pi \left[-r(s,a)\right] - H(\pi(a|s)) + \text{const}\,,
\end{aligned} \tag{32}
$$

Therefore, maximizing the entropy of the state-action OM accounts for maximizing the entropy of the policy such that conducting maximum entropy reinforcement learning with a recovered reward corresponds to the upper bound of the state-action OM matching problem. Similarly, if we model the

state transition OM with the Boltzmann distribution as $\rho_\pi(s, s') \propto \exp r(s, s')$, then:

$$\begin{aligned}
\mathrm{D}_{\mathrm{KL}}(\rho_\pi(s, s') \| \rho_{\pi_E}(s, s')) &= \sum_{s,s'} \rho_\pi(s, s') \log \frac{\rho_\pi(s, s')}{\rho_{\pi_E}(s, s')} \\
&= \sum_{s,s'} \rho_\pi(s, s') \left(-r(s, s') + \log \rho_\pi(s, s')\right) + \text{const} \\
&= \mathbb{E}_\pi \left[-r(s, s')\right] + \sum_{s,s'} \rho_\pi(s, s') \log \rho_\pi(s, s') + \text{const} \\
&= \mathbb{E}_\pi \left[-r(s, s')\right] + \sum_{s,s'} \rho_\pi(s, s') \log \rho_\pi(s, s') + \text{const} \\
&= \mathbb{E}_\pi \left[-r(s, s')\right] - H(\rho_\pi(s, s')) + \text{const} \,.
\end{aligned} \tag{33}$$

However, maximum the entropy of the state-transition OM $\rho_\pi(s, s') = \int_a \pi(a|s)\rho_\pi(s)T(s'|s, a)\,\mathrm{d}a$ does not account for maximizing the entropy of the policy, and therefore can not alleviate the ambiguity.

D Experiments

D.1 Experiment Settings

D.1.1 Real-World Traffic Dataset

NGSIM I-80 dataset includes three videos with a total length of 45 minutes recorded in a fixed area, from which 5596 driving trajectories of different vehicles can be obtained. We choose 85% of these trajectories as the training set and the remaining 15% as the test set. In our experiment, the state space includes the position and velocity vectors of the ego vehicle and six neighbor vehicles and the actions are acceleration and the change in steering angle.

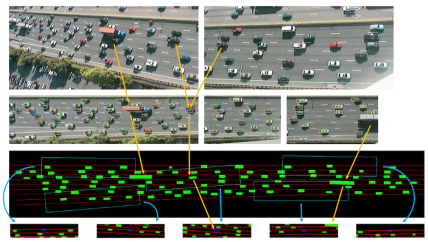


Figure 7: Visualization of NGSIM I-80 data set and its mapping on the simulator. This figure is borrowed from (Henaff et al., 2019).

D.1.2 Implementation Details

For all experiments, we implement the decoupled policy network, value network as two-layer MLPs with 256 hidden units and the discriminator as 128 hidden units. For fairness, we re-implement all the algorithms based on a Pytorch code framework[2] and adopt Soft Actor-Critic (SAC) (Haarnoja et al., 2018) as the RL learning algorithm for GAIfO and DPO.

For Mujoco benchmarks, we train an SAC agent to collect expert data, and take it for training the imitation learning agents without any normalization. At training time we remove the terminal state and episode will end until 1000 steps. At testing time the terminal state are set for fair comparison.

For NGSIM driving experiment, the original state contains the information of other cars, which is hard to predict. Therefore, we ignore it when predicting the state transition and the action of inverse dynamics. During training, we randomly pick one car to be controlled by the policy at the beginning of every episode, and we replay the other cars by data. The episode ends when cars

[2]https://github.com/KamyarGh/rl_swiss

collide or successfully get through the road. To reduce the sampling time in the driving simulator, we implemented parallel sampling using Python *multiprocessing* library. In practice, we ran 25 simulators to collect samples at the same time.

D.1.3 HYPERPARAMETERS

We list the key hyperparameters of the best performance of DPO on each task in Tab. 4. For each task, we first fine-tune GAIfO to find good hyperparameters for generative adversarial training, depending on which we further fine-tune state predictor coefficient λ_h and inverse dynamics coefficient λ_I from a initial hyperparameter $\lambda_h = 1.0$ and $\lambda_I = 0.5$. We find λ_h affects the performance most, along with the multi-step number k and the cycle loss. Note that DPO needs at least 1-step rollout for training the state transition predictor. In our experiment, we do not fine-tune the number of pre-training steps, and the final performances are almost the same with / without pre-training in most of environments. However, in few tasks, it can even deteriorate the training.

Table 4: Hyperparameters of DPO.

Environments	Invert.	InvDouble.	Hop.	Walk.	Half.	Ant.	NGSIM.
Trajectory maximum length	1000						1500
Optimizer	AdamOptimizer						
Discount factor γ	0.99						
Replay buffer size	2e5						2e6
Batch size	256						1024
State predictor coefficient λ_h	1.0				0.35	1.2	1.0
Tuning range of λ_h	[1.0]				[0.3,0.35,0.45,0.5,1.0]	[0.9,1.0,1.1,1.2,1.3]	[1.0]
Inverse dynamics coefficient λ_I	0.5				0.25	0.5	
Tuning range of λ_I	[0.5]				[0.25,0.5]	[0.5]	
Generative adversarial coefficient λ_G	1.0						
Generative adversarial reward form	$\log D$		$-\log(1-D)$	$\log D$			
Multi-step k	1		3	1	2		1
Cycle loss	✗			✓		✗	
Pre-train step	0		50000				0
Q learning rate	3e-4						
π learning rate	3e-4						
D learning rate	3e-4						
Gradient penalty weight	4.0				0.5		4.0
Reward scale	2.0						

D.2 QUALITATIVE ANALYSIS ON THE LEARNED POLICY

In this section, we provide qualitative experiments on investigating how the learned policy behaves. Based on the setting as in Section 5.1, we first analyse how the state transition predictor behaves. Specifically, we compare the learned prediction with the expert state transition, shown in Fig. 8, which indicates that the prediction by state transition predictor exactly match the state transition in the demonstration and achieve to the duty of 'plan the target'.

In addition, we further discuss the output action probability of the inverse dynamics, shown in Fig. 9. In the figure, we compare the action distribution among 20 possible actions at 3 different states with the expert ground truth. We conclude that the inverse dynamics mismatch is not the key for imitating expert the state sequence since the agent can select different actions from the expert as long as they lead to the same transition. Therefore, the inverse dynamics module play his role for 'learn the skill' of the agent's own.

D.3 ANALYSIS ON NO REDUNDANT ACTION

To better understand what the effect of the added action ambiguity achieves, we also include an experiment on the grid world environment in Section 5.1 when there are no redundant actions to show if empirically the baseline algorithms suffer from environments with action ambiguity. As shown in Fig. 10, BCO and DPO share similar asymptotic performance (KLD), but DPO has a significantly faster convergence rate. On the contrary, GAIfO also fails to find the second path.

D.4 DISTRIBUTIONAL EVALUATION METRIC

Apart from the accumulated reward reported in Tab. 2, the performance of imitation learning methods should also be evaluated by distributional similarities to expert data. For example, in SOIL tasks we try

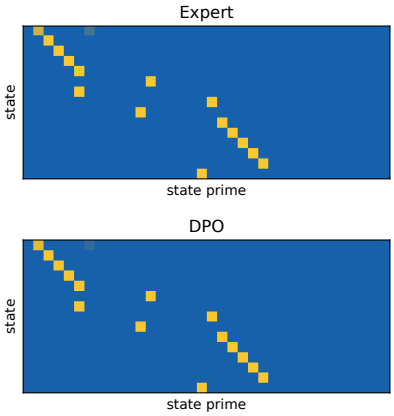


Figure 8: Each grid in the 'Expert' graph represents the transition probability from s (state) to s' (state prime) in demonstration data. Since some states do not appear in the demonstration, the transition probabilities from such states are undefined, and we exclude them from the graph. The 'DPO' graph shows the state prime output by state transition predictor from each state accordingly.

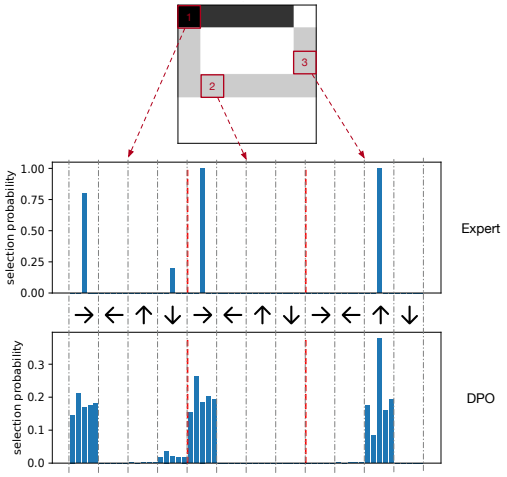


Figure 9: The learned policy (action selection probability) among 20 possible actions at 3 states (which is marked on the top sub-graph). The distributions in different states are split by red lines, and the resulting transition is labeled between two sub-graphs. This shows that the inverse dynamics mismatch is not the key for imitating expert the state sequence since the agent can select different actions from the expert as long as they lead to the same transition.

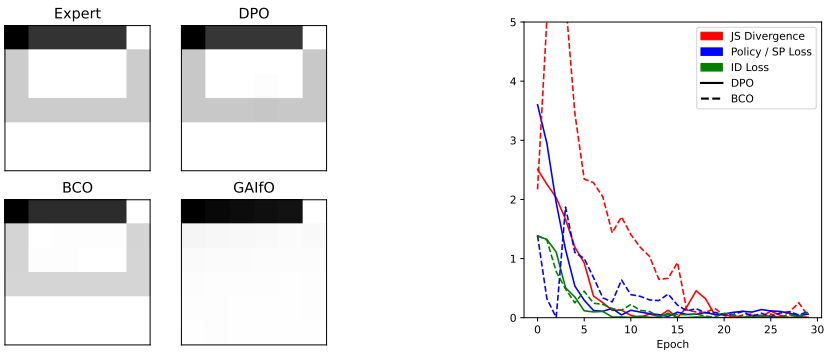


Figure 10: The rollout density and loss curves when $k = 1$. BCO and DPO have similar asymptotic performance (KLD), but DPO has a significantly faster convergence rate. On the contrary, GAIfO still fails to find the second path.

to evaluate the KL divergence between policy and expert state transitions $\mathrm{D}_{\mathrm{KL}}(\rho_{\pi_E}(s, s')\|\rho_{\pi}(s, s'))$ for different methods. Since it is hard to compute the distributional distance in high-dimensional continuous control environments, we reduce the dimension of the input data to 2 dimensions. Specifically, we adopt UMAP (McInnes & Healy, 2018), which maintains a mapping function that can be used for transforming new data collections. In our case, we first fit a UMAP model on the expert demonstration and then use it to transform (s, s') pairs collected by different algorithms. We first estimate the distribution via Kernel Density Estimation (KDE) (Rosenblatt, 1956) with Gaussian kernel to compute the Kullback-Leibler (KL) divergence, and show the qualitative results in Tab. 5. Furthermore, we visualize a 2 dimensional distributional density example of these trajectories on Halfcheetah in Fig. 11. Higher frequency positions in collected data are colored darker in the plane, and higher the value with respect to its marginal distributions. And it is noticeably that DPO does not reach a higher return but recover the better expert state transition occupancy measure.

Table 5: KL divergence between policy-sampled and the expert state transitions distribution.

	Hopper	Walker2d	HalfCheetah	Ant
BCO	**1.32 ± 0.04**	1.63 ± 0.26	5.76 ± 0.31	3.76 ± 0.42
GAIfO	1.77 ± 0.05	1.32 ± 0.21	2.47 ± 0.79	**0.40 ± 0.04**
DPO	1.76 ± 0.05	**1.13 ± 0.09**	**1.68 ± 0.16**	0.48 ± 0.06

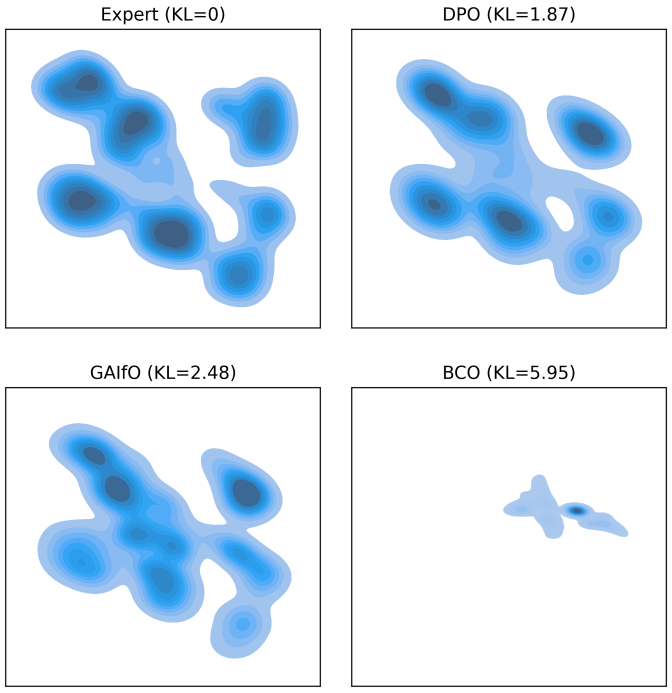


Figure 11: Visualization of sampled state transition distributions on HalfCheetah environment using UMAP reduction.

D.5 Ablation Study On Hyperparameters

In this section we investigate the effect on different values of hyperparameter λ_h. As illustrated in Fig. 12, the final performance is robust upon a range of λ_h. However, we find it affects the sample efficiency and the optimal hyperparameter among different tasks differs.

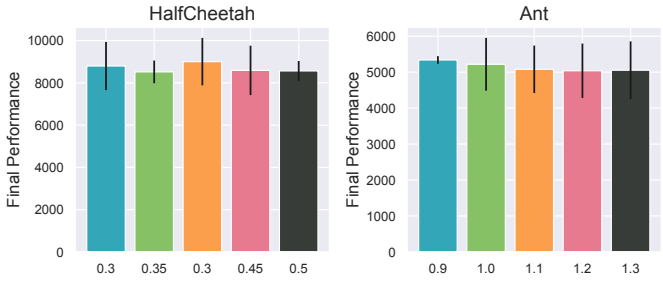


Figure 12: Hyperparameter study on λ_h.

D.6 Empirical Correlation between Compounding Error and Reward

The motivation of DPO indicates that if the agent can exactly predict where the expert will go and then learn a skill to reach that place, it can solve SOIL efficiently. In previous sections we propose to evaluate the distance of the reaching states and the predicted consecutive states to quantify the compounding error. Interestingly, in our experiments, we do find that the compounding error has a great impact on the efficacy of DPO. Therefore, we analyze the empirical correlation between the prediction-real distance and the reward. Specifically, we sample several epochs from experiments with different hyperparameters on each tasks and draw the connection of its prediction-real distance and its reward. As shown in Fig. 13, lower distance always achieves higher performance, indicating the rationality of the intuition and the key ingredient for utilizing DPO.

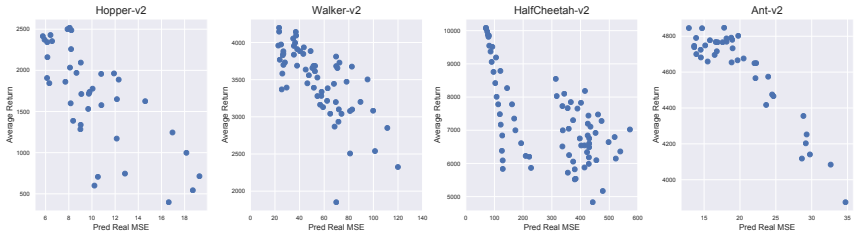


Figure 13: The empirical correlation between the prediction-real distance and the reward. Typically, less prediction-real distance achieves better performance

D.7 Complete Evaluation Results

In this section we show complete evaluation training curves of DPO with different regularization in Fig. 14. Typically, experiments with less prediction-real distance can achieve better performance. It is worth noting that, DPO can generally achieve better efficiency than the baselines in most of the environments. However, with fine-tuning the regularization, we are able to dig the potential of DPO.

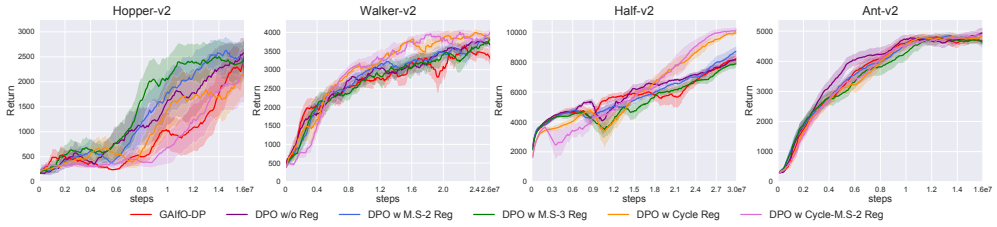


Figure 14: Complete learning curves of DPO with different regularization.

D.8 Additional Comparison and Ablation On Inverse Dynamics Regularization

In this section we emphasize on the regularization on the inverse dynamics. In this paper, specifically, we propose to utilize the policy gradient with generative adversarial methods to encourage the agent

Table 6: Additional ablation studies on the inverse dynamics regularization.

	InvertedPendulum	InvertedDoublePendulum	Hopper	Walker2d	HalfCheetah	Ant
BCO	**1000.0 ± 0.00**	416.92 ± 141.56	1516.91 ± 524.86	270.45 ± 33.22	6.56 ± 151.49	456.45 ± 179.76
GAIfO	**1000.0 ± 0.00**	**8589.46 ± 1391.82**	3068.10 ± 24.90	3864.03 ± 326.64	8918.66 ± 1031.41	4879.13 ± 897.46
DPO (w/o PG)	**1000.00 ± 0.00**	3933.33 ± 3414.51	713.17 ± 369.05	310.53 ± 68.27	-442.55 ± 120.23	-383.12 ± 198.18
DPO (w PG)	**1000.00 ± 0.00**	**8587.96 ± 1394.29**	**3163.74 ± 64.46**	**4395.21 ± 216.96**	**10522.08 ± 394.44**	5413.03 ± 161.97
DPO (w/o SP)	801.14 ± 444.66	5977.65 ± 4655.55	2587.88 ± 1237.44	3902.7 ± 171.12	8816.94 ± 998.3	5043.12 ± 552.29
DPO (w/o PG, w CL)	**1000.0 ± 0.0**	120.73 ± 23.17	1235.19 ± 938.26	20.63 ± 37.73	-585.45 ± 73.58	7.33 ± 22.56

to match the state transition of the expert. This is proved to improve the performance and the sample efficiency as shown in Section 5.2. However, in other point of view, our work can also be seen as adding additional supervision signals for a generative adversarial imitation learning method, due to the flexibility of the decoupled policy structure. Therefore, one can regard utilizing policy gradient (PG) in DPO as encouraging the imitator to follow the expert demonstrations better than a naive inverse model. Under this view, we aim to conduct ablation studies on the importance of the PG.

To this end, we compare PG with a related work (Pathak et al., 2018), which studies the problem of state-only imitation learning by matching the demonstrated image sequence. In the training stage, Pathak et al. (2018) allow the agent to learn a goal-conditioned skill (GSP) policy (i.e, an recurrent inverse dynamics model) to predict a plausible action; and in the evaluation stage, the agent is provided with demonstrated images, and it executes to achieve the every intermediate goal states one-by-one. Compared with DPO, Pathak et al. (2018) does not predict the target, instead the policy takes both the intermediate goal states s_g from the demonstration along with the action history a_h into planning the next action to reach the goal states. To achieve that, Pathak et al. (2018) also requires a binary classifier to identify whether the agent achieves the goals so that it can switch to the next goal state. Furthermore, Pathak et al. (2018) proposes a foward-consistency loss, which is used to regularize the inverse dynamics model to predict a plausible action; on the contrary, in our work, we want to regularize the state transition predictor to be consistent with the environmental forward dynamics, and therefore we do not apply the cycle loss to update the inverse dynamics model but utilize the PG regularization. Intuitively, in Pathak et al. (2018), the consistency is computed as:

$$s, s_g, a_h \xrightarrow{\text{GSP Policy}} a \xrightarrow{\text{Forward Model}} \underbrace{\tilde{s'} \quad s'}_{\text{Consistency Loss}} \xleftarrow{\text{execute (s,a)}} \text{Real Env}$$

However, our cycle consistency is more like the one in Edwards et al. (2020):

$$s_E \xrightarrow{\text{State Predictor}} \underbrace{\hat{s'} \xrightarrow{\text{Inverse Dynamics}} a \xrightarrow{\text{Forward Model}} \tilde{s'}}_{\text{Consistency Loss}}$$

Considering that using cycle consistency of Pathak et al. (2018) or discriminator rewards of ours in training of inverse dynamics are both encouraging state-matching, we include a comparison experiment for replace the policy gradients as the cycle consistency loss in Pathak et al. (2018), as denoted as DPO (w/o PG, w CL) shown in Tab. 6. From the results, we observe that the consistency regularization on inverse dynamics can decrease the compounding error in some environments (from the performance gain of DPO (w/o PG, w CL) over DPO (w/o PG)), but is far less effective than the PG regularization. This is rather obvious on the harder tasks with higher-dimensional state spaces.

To further illustrate if both supervision signals count, we also test the performance of DPO without the loss of expert state prediction, denoted as DPO (w/o SP) in Tab. 6. Obviously, without predicting the expert states, DPO (w/o SP) behaves even worse than GAIfO on some tasks since the inputs for the inverse dynamics no longer have semantic meanings and therefore the supervised loss for the inverse dynamics might hurt the performance. This ablation shows that explicitly predicting the expert's next states actually matters and is meaningful in achieving higher performance.