Adversarial Option-Aware Hierarchical Imitation Learning

Mingxuan Jing [1] **Wenbing Huang** [1] **Fuchun Sun** [†] [1 2] **Xiaojian Ma** [3] **Tao Kong** [4] **Chuang Gan** [5] **Lei Li** [4]

Abstract

It has been a challenge to learning skills for an agent from long-horizon unannotated demonstrations. Existing approaches like Hierarchical Imitation Learning(HIL) are prone to compounding errors or suboptimal solutions. In this paper, we propose Option-GAIL, a novel method to learn skills at long horizon. The key idea of Option-GAIL is modeling the task hierarchy by options and train the policy via generative adversarial optimization. In particular, we propose an Expectation-Maximization(EM)-style algorithm: an E-step that samples the options of expert conditioned on the current learned policy, and an M-step that updates the low- and high-level policies of agent simultaneously to minimize the newly proposed option-occupancy measurement between the expert and the agent. We theoretically prove the convergence of the proposed algorithm. Experiments show that Option-GAIL outperforms other counterparts consistently across a variety of tasks.

1. Introduction

Hierarchical Imitation Learning (HIL) has exhibited promises on acquiring long-term skills directly from demonstrations (Le et al., 2018; Yu et al., 2018; Byrne & Russon, 1998; Sharma et al., 2019; Hamidi et al., 2015). It contends the nature of sub-task decomposition in long-horizontal tasks, enjoying richer expressivity on characterizing complex decision-making than canonical Imitation Learning (IL). In general, HIL designs micro-policies for accomplishing the specific control for each sub-task, and employs a macro-policy for scheduling the switching of micro-policies. Such a two-level decision-making process is usually formulated by an Option model (Sutton et al., 1999) or goal-based framework (Le et al., 2018). Although some works (Fei et al., 2020) have assumed the help of sub-task segmentation annotations, this paper mainly focuses on learning from unsegmented demonstrations to allow more practicability.

Plenty of HIL methods have been proposed, including the Behavior-Cloning(BC) based and the Inverse Reinforcement Learning(IRL) based approaches. For the first avenue, Zhang & Paschalidis (2020); Le et al. (2018) customize BC to hierarchical modeling (Daniel et al., 2016), which, unfortunately, is prone to compounding errors (Ross et al., 2011) in case that the demonstrations are limited in practice. On the contrary, IRL is potential to avoid compounding errors by agent self-explorations. However, addressing IRL upon the Option model, by no means, is non-trivial considering the temporal coupling of the high-level and low-level policies. The works by Sharma et al. (2018); Lee & Seo (2020) relax this challenge by training the two-level policies separately. Nevertheless, this two-stage method potentially leads to sub-optimal solutions in the absence of training collaboration between the two stages.

To overcome the aforementioned issues, this work proposes a novel Hierarchical IRL framework—Option-GAIL, which is theoretically elegant, free of compounding errors, and end-to-end trainable. Basically, it is built upon GAIL (Ho & Ermon, 2016) with two nutritive enhancements: 1) replacing Markov Decision Process (MDP) with the Option model; 2) augmenting the occupancy measurement matching with options. Note that GAIL is a popular IL method that casts policy learning upon MDP into an occupancy measurement matching problem. Therefore, it is natural to replace the MDP with the Option model for hierarchical modeling. Besides, GAIL mimicking a policy from an expert is equivalent to matching its occupancy measurement. This equivalence holds when the policy is one-to-one corresponding to its induced occupancy measurement, which, yet, is not valid in our hierarchical context. As we will depict in the paper, the policy of HIL is related to options, and its one-to-one correspondence only exists with regard to option-extended occupancy measurement other than traditional occupancy measurement without options. Hence, it is indispensable to involve options into the matching goal in our second enhancement. Notably, the option switching is inherently guaranteed in our model, without extra regulators such as

[1]Department of Computer Science and Technology, Tsinghua University, Beijing, China (Mingxuan Jing <jingmingxuan@outlook.com>; Wenbing Huang <hwenbing@126.com>; Fuchun Sun <fcsun@tsinghua.edu.cn>) [2]THU-Bosch JCML center [3]University of California, Los Angeles, USA [4]Bytedance AI Lab, Beijing, China [5]MIT-IBM Watson AI Lab, USA. Correspondence to: [†]Fuchun Sun <fcsun@mail.tsinghua.edu.cn>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

mutual information used in (Sharma et al., 2018) to encourage the correlation between action-state pairs and options.

It is non-straightforward to train our Option-GAIL, specifically when the expert options are unavailable. We thereby propose an EM-like learning algorithm to remedy the training difficulty. Specifically, in the E-step, we apply a Viterbi method (Viterbi, 1967) to infer the expert options conditional on the agent policy and expert actions/states. For the M-step, with the expert options provided, we optimize both the high- and low-level policies to minimize the discrepancy between the option-occupancy measurements of expert and agent. To be more specific, the M-step is implemented by a min-max game: maximizing a discriminator to estimate the discrepancy, and updating the policy to minimize the discriminative cost via a hierarchical RL method (Zhang & Whiteson, 2019). Notably, the convergence of the proposed EM algorithm is theoretically guaranteed if certain mild conditions hold.

In summary, our main contributions include:

- We propose Option-GAIL, a theoretically-grounded framework that integrates hierarchical modeling with option-occupancy measurement matching. It is proved that Option-GAIL ensures the discrepancy minimization between the policies of demonstrator and imitator.
- We propose an EM-like algorithm to train the parameters of Option-GAIL end-to-end. This method alternates option inference and policy updating and is proved to converge eventually.
- We evaluate our proposed method on several robotic locomotion and manipulation tasks against state-of-the-art HIL/IL baselines. The results demonstrate that our approach attains both dramatically faster convergence and better final performance over the counterparts. A complete set of ablation studies also verify the validity of each component we proposed.

2. Related Works

There have already been plenty of works researching on HIL, which, in general, can be categorized into two classes: Hierarchical Behavior Cloning (H-BC) and Hierarchical Inverse Reinforcement Learning (H-IRL).

Hierarchical BC. In these avenues, HIL is an extension of Behavior Cloning (BC) (Pomerleau, 1988; Atkeson & Schaal, 1997). As a result, the missing sub-task information needs to be provided or inferred to ensure the learnability of hierarchical policies. For example, Le et al. (2018) require predefined sub-goals when learning the goal-based policies; Kipf et al. (2019) try to alleviate the requirement of sub-task information by formulating the policy learning as an unsupervised trajectory soft-segmentation problem; Daniel et al. (2016) and Zhang & Paschalidis (2020) employ Baum-Welch algorithm (Baum et al., 1972) upon the option framework to infer the latent option from demonstrations, and directly optimize both the high- and low-level policies. Despite its easy implementation, behavior cloning is vulnerable to compounding errors (Ross et al., 2011) in case of the limited demonstrations, while our method enjoys better sample efficiency thanks to the IRL formulation.

Hierarchical IRL. Contrasted to the BC-based approaches, Hierarchical IRL avoids compounding errors by taking advantage of the agent's self-exploration and reward reconstruction. Following the Generative Adversarial Imitation Learning (GAIL) (Ho & Ermon, 2016), the works by (Sharma et al., 2018; Lee & Seo, 2020) realize HIL by introducing a regularizer into the original IRL problem and maximizing the directed information between generated trajectories and options. However, the high- and low-level policies are trained separately in these approaches: the high-level policy is learned with behavior cloning and remains fixed during the GAIL-based low-level policy learning. As we reported in the experiments, such a two-staged paradigm could lead to potentially poor convergence compared to our end-to-end training fashion. Henderson et al. (2018) propose an end-to-end HIL method without pre-training. However, it adopts a Mixture-of-Expert (MoE) strategy rather than the canonical option framework. Therefore, an option is exclusively determined by the corresponding state, ignoring its relation to the option of the previous step (Figure 1). On the contrary, our method conducts option inference and policy optimization that are strictly amenable to the option dynamics (Sutton et al., 1999), thus delineating the hierarchy of sub-tasks in a more holistic manner.

3. Preliminaries

This section briefly introduces preliminary knowledge and notation used in our work.

Markov Decision Process: A Markov Decision Process (MDP) is a 6-element-tuple $\left(\mathbb{S}, \mathbb{A}, P^a_{s,s'}, R^a_s, \mu_0, \gamma\right)$, where $(\mathbb{S}, \mathbb{A})$ denote the continuous/discrete state-action space; $P^a_{s,s'} = P(s_{t+1} = s'|s_t = s, a_t = a)$ is the transition probability of next state $s' \in \mathbb{S}$ given current state $s \in \mathbb{S}$ and action $a \in \mathbb{A}$, determined by the environment; $R^a_s = \mathbb{E}[r_t|s_t = s, a_t = a]$ returns the expected reward from the task when taking action a on state s; $\mu_0(s)$ denotes the initial state distribution and $\gamma \in [0, 1)$ is a discounting factor. The effectiveness of a policy $\pi(a|s)$ is evaluated by its expected infinite-horizon reward: $\eta(\pi) = \mathbb{E}_{s_0 \sim \mu_0, a_t \sim \pi(s_t), s_{t+1} \sim P^{a_t}_{s_t,s_{t+1}}} \left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$.

The Option Framework: Options $\mathbb{O} = \{1, \cdots, K\}$ are introduced for modeling the policy-switching procedure on long-term tasks (Sutton et al., 1999). An option model is defined as a tuple $\mathcal{O} = \left(\mathbb{S}, \mathbb{A}, \{\mathbb{I}_o, \pi_o, \beta_o\}_{o\in\mathbb{O}}, \pi_{\mathcal{O}}(o|s), P^a_{s,s'}, R^a_s, \mu_0, \gamma\right)$, where, $\mathbb{S}, \mathbb{A}, P^a_{s,s'}, R^a_s, \mu_0, \gamma$ are defined as the same as MDP; $\mathbb{I}_o \subseteq \mathbb{S}$ denotes an initial set of states, from which an option can be activated; $\beta_o(s) = P(b = 1|s)$ is a terminate function which decides whether current option should be terminated or not on a given state s; $\pi_o(a|s)$ is the intra-option policy that determines an action on a given state within an option o; a new option is activated in the call-and-return style by an inter-option policy $\pi_{\mathcal{O}}(o|s)$ once the last option terminates.

Generative Adversarial Imitation Learning (GAIL): For simplicity, we denote $(x_0, \cdots, x_n)$ as $x_{0:n}$ for short throughout this paper. Given expert demonstrations on a specified task as $\mathbb{D}_{demo} = \left\{\tau_E = (s_{0:T}, a_{0:T})\right\}$, imitation learning aims at finding a policy π that can best reproduce the expert's behavior, without the access of the real reward. Ho & Ermon (2016); Ghasemipour et al. (2020) cast the original maximum-entropy inverse reinforcement learning problem into an occupancy measurement (Syed et al., 2008) matching problem:

$$\arg\min_{\pi} D_f\left(\rho_\pi(s,a)\|\rho_{\pi_E}(s,a)\right), \tag{1}$$

where, D_f computes f-Divergence, $\rho_\pi(s,a)$ and $\rho_{\pi_E}(s,a)$ are the occupancy measurements of agent and expert respectively. By introducing a generative adversarial structure (Goodfellow et al., 2014), GAIL minimizes the discrepancy via alternatively optimizing the policy and the estimated discrepancy. To be specific, a discriminator $D_\theta(s,a) : \mathbb{S}\times\mathbb{A} \mapsto (0,1)$ parameterized by θ in GAIL is maximized and then the policy is updated to minimize the overall discrepancy along each trajectory of exploration. Such optimization process can be cast into: $\max_\theta \min_\pi \mathbb{E}_\pi\left[\log D_\theta(s,a)\right] + \mathbb{E}_{\pi_E}\left[\log\left(1 - D_\theta(s,a)\right)\right]$.

4. Our Framework: Option-GAIL

In this section, we provide the details of the proposed Option-GAIL. Our goal is towards imitation learning from demonstrations of long-horizon tasks consisting of small sub-tasks. We first introduce necessary assumptions to facilitate our analyses, and follow it up by characterizing the motivations and formulation of Option-GAIL. We then provide the EM-like algorithm towards the solution and conclude this section with theoretical analyses on the convergence of our algorithm.

4.1. Assumptions and Definitions

We have introduced the full option framework for modeling switching procedure on hierarchical tasks in Section 3. However, it is inconvenient to directly learn the policy of this framework due to the difficulty of determining the initial set $\mathbb{I}_o$ and break condition β_o. Here, we introduce the following assumption given which the original option framework will be equivalently converted into the one-step option that is easier to be dealt with.

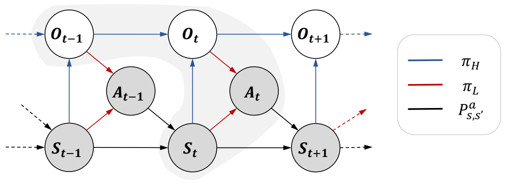

Figure 1. The probabilistic graph of the one-step option model. The shade masks the group of nodes that induce our option-occupancy measurement.

Assumption 1 (Valid Option Switching) *We assume that, 1) each state is switchable: $I_o = \mathbb{S}, \forall o \in \mathbb{O}$; 2) each switching is effective: $P(o_t = o_{t-1}|\beta_{o_{t-1}}(s_{t-1}) = 1) = 0$.*

Assumption 1 asserts that each state is switchable for each option, and once the switching happens, it switches to a different option with probability 1. Such assumption usually holds in practice without the sacrifice of model expressiveness. Now, we define the following one-step model.

Definition 1 (One-step Option) *We define $\mathcal{O}_{\text{one-step}} = \left(\mathbb{S}, \mathbb{A}, \mathbb{O}^+, \pi_H, \pi_L, P^a_{s,s'}, R^a_s, \tilde{\mu}_0, \gamma\right)$ where $\mathbb{O}^+ = \mathbb{O} \cup \{\#\}$ consists of all options plus a special initial option class satisfying $o_{-1} \equiv \#, \beta_\#(s) \equiv 1$. Besides, $\tilde{\mu}_0(s,o) \doteq \mu_0(s)\mathbb{1}_{o=\#}$, where $\mathbb{1}_{x=y}$ is the indicator function, and it is equal to 1 iff $x = y$, otherwise 0. The high- and low-level policies are defined as:*

$$\begin{aligned}\pi_H(o|s,o') &\doteq \beta_{o'}(s)\pi_{\mathcal{O}}(o|s) + \left(1-\beta_{o'}(s)\right)\mathbb{1}_{o=o'},\\ \pi_L(a|s,o) &\doteq \pi_o(a|s).\end{aligned} \tag{2}$$

It can be derived that $\mathcal{O}_{\text{one-step}} = \mathcal{O}$ under Assumption 1[1]. This equivalence is beneficial as we can characterize the switching behavior by only looking at π_H and π_L without the need to justify the exact beginning/breaking condition of each option. We denote $\tilde{\pi} \doteq (\pi_H, \pi_L)$ and $\tilde{\Pi} = \{\tilde{\pi}\}$ as the set of stationary policies. We will employ the one-step option in the remainder of our paper. Note that in a current paper (Li et al., 2021) the rigorous notions and formulations have been provided for further discussing the relationship between the full option model and the one-step option model.

We also provide the definition of the option-occupancy mea-

[1]The proofs of all theorems are provided in Appendix.

surement by borrowing the notion of the occupancy measurement $\rho(s, a)$ from MDP (Syed et al., 2008).

Definition 2 (Option-Occupancy Measurement) *We define the option-occupancy measurement as* $\rho_{\tilde{\pi}}(s, a, o, o') \doteq \mathbb{E}_{\tilde{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t \mathbb{1}_{(s_t=s, a_t=a, o_t=o, o_{t-1}=o')}\right]$.

The measurement $\rho_{\tilde{\pi}}(s, a, o, o')$ can be explained as the distribution of the state-action-option tuple that generated by the policy π_H and π_L on a given $\tilde{\mu}_0$ and $P^a_{s,s'}$. According to the Bellman Flow constraint, one can easily obtain that $\rho_{\tilde{\pi}}(s, a, o, o')$ belongs to a feasible set of affine constraint $\mathbb{D} = \{\rho(s, a, o, o') \geq 0; \sum_{a,o} \rho(s, a, o, o') = \tilde{\mu}_0(s, o') + \gamma \sum_{s', a', o''} P^{a'}_{s', s} \rho(s', a', o', o'')\}$.

4.2. Problem Formulation

Now we focus on the imitation task on long-term demonstrations. Note that GAIL is no longer suitable for this scenario since it is hard to capture the hierarchy of sub-tasks by MDP. Upon GAIL, the natural idea is to model the long-term tasks via the one-step Option instead of MDP, and the policy is learned by minimizing the discrepancy of the occupancy measurement between expert and agent, namely,

$$\min_{\tilde{\pi}} D_f\left(\rho_{\tilde{\pi}}(s, a) \| \rho_{\tilde{\pi}_E}(s, a)\right). \tag{3}$$

While this idea is straightforward and never explored before, we claim that it will cause solution ambiguity—we cannot make sure that $\tilde{\pi} = \tilde{\pi}_E$, even we can get the optimal solution $\rho_{\tilde{\pi}}(s, a) = \rho_{\tilde{\pi}_E}(s, a)$ in Equation 3.

Intuitively, for the hierarchical tasks, the action we make depends not only on the current state we observe but also on the option we have selected. By Definition 1, the hierarchical policy is relevant to the information of current state, current action, last-time option and current option, which motivates us to leverage the option-occupancy measurement instead of conventional occupancy measurement to depict the discrepancy between expert and agent. Actually, we have a one-to-one correspondence between $\tilde{\Pi}$ and $\mathbb{D}$.

Theorem 1 (Bijection) *For each $\rho \in \mathbb{D}$, it is the option-occupancy measurement of the following policy:*

$$\pi_H = \frac{\sum_a \rho(s, a, o, o')}{\sum_{a,o} \rho(s, a, o, o')}, \pi_L = \frac{\sum_{o'} \rho(s, a, o, o')}{\sum_{a,o'} \rho(s, a, o, o')}, \tag{4}$$

and $\tilde{\pi} = (\pi_H, \pi_L)$ is the only policy whose option-occupancy measure is ρ.

With Theorem 1, optimizing the option policy is equivalent to optimizing its induced option-occupancy measurement, since $\rho_{\tilde{\pi}}(s, a, o, o') = \rho_{\tilde{\pi}_E}(s, a, o, o') \Leftrightarrow \tilde{\pi} = \tilde{\pi}_E$. Our hierarchical imitation learning problem becomes:

$$\min_{\tilde{\pi}} D_f\left(\rho_{\tilde{\pi}}(s, a, o, o') \| \rho_{\tilde{\pi}_E}(s, a, o, o')\right) \tag{5}$$

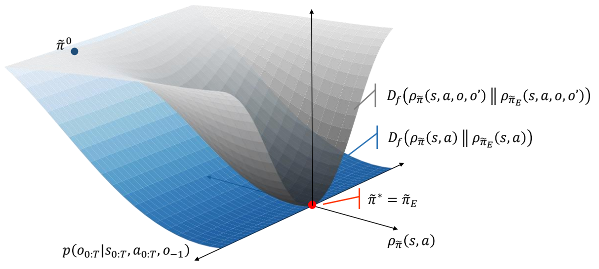

Figure 2. Illustration of the different convergence properties of the optimisation problems defined in Equation 3 and Equation 5. The options are not explicitly constrained by Equation 3

.

Note that the optimization problem defined in Equation 5 implies that in Equation 3, but not vice versa: (1) since $\rho_{\tilde{\pi}}(s, a) = \sum_{o,o'} \rho_{\tilde{\pi}}(s, a, o, o')$, we can easily obtain $\rho_{\tilde{\pi}}(s, a, o, o') = \rho_{\tilde{\pi}_E}(s, a, o, o') \Rightarrow \rho_{\tilde{\pi}}(s, a) = \rho_{\tilde{\pi}_E}(s, a)$; (2) as $\mathbb{E}_{q(o,o')}\left[D_f\left(\rho_{\tilde{\pi}}(s, a, o, o') \| \rho_{\tilde{\pi}_E}(s, a, o, o')\right)\right] \geq \mathbb{E}_{q(o,o')}\left[D_f\left(\rho_{\tilde{\pi}}(s, a) \| \rho_{\tilde{\pi}_E}(s, a)\right)\right]$ for any option distribution $q(o, o')$, addressing the problem defined in Equation 5 is addressing an upper bound of that defined in Equation 3. Indeed, we will show in § 4.6 that decreasing the goal in Equation 5 will definitely decrease that of Equation 3 given certain conditions. Figure 2 depicts the relationship between Equation 5 and Equation 3.

Yet, it is nontrivial to tackle Equation 5 particularly owing to the unobserved options in expert demonstrations. Here, we propose an EM-like algorithm to address it, which will be detailed in § 4.3 and § 4.4.

4.3. Option-Occupancy Measurement Matching

In this section, we assume the expert options are given and train the hierarchical policy $\tilde{\pi}$ to minimize the goal defined in Equation 5. We denote the option-extended expert demonstrations as $\tilde{\mathbb{D}}_{\text{demo}} = \{\tilde{\tau} = (s_{0:T}, a_{0:T}, o_{-1:T})\}$.

Inspired from GAIL (Ho & Ermon, 2016), rather than calculating the exact value of the option-occupancy measurement, we propose to estimate the discrepancy by virtue of adversarial learning. We define a parametric discriminator as $D_\theta(s, a, o, o') : \mathbb{S} \times \mathbb{A} \times \mathbb{O} \times \mathbb{O}^+ \mapsto (0, 1)$. If specifying the f-divergence as Jensen–Shannon divergence, Equation 5 can be converted to a min-max game:

$$\begin{aligned} \min_{\tilde{\pi}} \max_{\theta} \mathbb{E}_{\tilde{\mathbb{D}}_{\text{demo}}}\left[\log\left(1 - D_\theta(s, a, o, o')\right)\right] \\ + \mathbb{E}_{\tilde{\mathbb{D}}_{\tilde{\pi}}}\left[\log\left(D_\theta(s, a, o, o')\right)\right]. \end{aligned} \tag{6}$$

The inner loop is to train $D_\theta(s, a, o, o')$ with the expert demonstration $\tilde{\mathbb{D}}_{\text{demo}}$ and samples $\tilde{\mathbb{D}}_{\tilde{\pi}}$ generated by self-exploration. Regarding the outer loop, a hierarchical reinforcement learning (HRL) method should be used to minimize the discrepancy:

$$\tilde{\pi}^\star = \arg\min_{\tilde{\pi}} \mathbb{E}_{\tilde{\pi}} \left[c(s, a, o, o') \right] - \lambda \mathbb{H}(\tilde{\pi}), \tag{7}$$

where $c(s, a, o, o') = \log D_\theta(s, a, o, o')$ and the causal entropy $\mathbb{H}(\tilde{\pi}) = \mathbb{E}_{\tilde{\pi}} [-\log \pi_H - \log \pi_L]$ as a policy regularizer with $\lambda \in [0, 1]$. Notice that the cost function is related to options, which is slightly different from many HRL problems with option agnostic cost/reward (Bacon et al., 2017; Zhang & Whiteson, 2019). To deal with this case, we reformulate the option-reward and optimize Equation 7 using similar idea as Zhang & Whiteson (2019).

DAC characterizes the option model as two-level MDPs. Here we borrow its theoretical result by re-interpreting the reward used by each MDP: For the high-level MDP, we define state $s_t^H \doteq (s_t, o_{t-1})$, action $a_t^H \doteq o_t$, and reward $R^H((s, o'), o) \doteq -\sum_a \pi_L(a_t|s_t, o_t) c(s_t, a_t, o_t, o_{t-1})$. For the low-level MDP, we denote $s_t^L \doteq (s_t, o_t)$, $a_t^L \doteq a_t$, and $R^L((s, o), a) = -\sum_{o'} c(s, a, o, o') p_{\pi_H}(o'|s, o)$ with the posterior propability $p_{\pi_H}(o'|s, o) = \pi_H(o|s, o') \frac{p(o'|s)}{p(o|s)}$. Other terms including the initial state distributions μ_0^H and μ_0^L, the state-transition dynamics P^H and P^L are defined similar to Zhang & Whiteson (2019). The HRL task in Equation 7 can be separated into two non-hierarchical ones with augmented MDPs: $\mathbb{M}^H = (S^H, A^H, P^H, R^H, \mu_0^H, \gamma)$, $\mathbb{M}^L = (S^L, A^L, P^L, R^L, \mu_0^L, \gamma)$, whose action decisions depend on π_H and π_L, separately. Such two non-hierarchical problems can be solved alternatively by utilizing typical reinforcement learning methods like PPO (Schulman et al., 2017) with immediate imitation reward $r_t = -c(s_t, a_t, o_t, o_{t-1})$ when in practice.

By alternating the inner loop and the outer loop, we finally derive $\tilde{\pi}^\star$ that addresses Equation 5.

4.4. Expert Option Inference

So far, we have supposed that the expert options are provided, which, however, is usually not true in practice. It thus demands a kind of method to infer the options from observed data (states and actions). Basically, given a policy, the options are supposed to be the ones that maximize the likelihood of the observed state-actions, according to the principle of Maximum-Likelihood-Estimation (MLE).

We approximate the expert policy with $\tilde{\pi}$ learned by the method above. With states and actions observed, the option model will degenerate to a Hidden-Markov-Model (HMM), therefore, the Viterbi method (Viterbi, 1967) is applicable for expert option inference. We call this method as Option-Viterbi for its specification to the option model.

Given current learned π_H, π_L and $\tau = (s_{0:T}, a_{0:T}) \in \mathbb{D}_{\text{demo}}$, Option-Viterbi generates the most probable values of $o_{-1:T}$ that induces the maximization of the whole trajectory:

$$\begin{aligned} &\arg\max_{o_{-1:T}} P(s_{0:T}, a_{0:T}, o_{-1:T}) \\ \propto &\arg\max_{o_{0:T}} \log P(a_{0:T}, o_{0:T}|s_{0:T}, o_{-1} = \#) \\ = &\arg\max_{o_T} \hat{\alpha}_T(o_T). \end{aligned} \tag{8}$$

Here a maximum foreword message $\hat{\alpha}_t(o_t)$ is introduced for indicating the maximum probability value of the partial trajectory till time t given o_t, and it can be calculated recursively as Equation 9:

$$\begin{aligned} \hat{\alpha}_t(o_t) &= \max_{o_{0:t-1}} \log P(a_{0:t}, o_t|s_{0:t}, o_{0:t-1}, o_{-1} = \#) \\ &= \max_{o_{t-1}} \hat{\alpha}_{t-1}(o_{t-1}) + \log \pi_H(o_t|s_t, o_{t-1}) \\ &\quad + \log \pi_L(a_t|s_t, o_t), \end{aligned} \tag{9}$$

$$\hat{\alpha}_0(o_0) = \log \pi_L(a_0|s_0, o_0) + \log \pi_H(o_0|s_0, \#). \tag{10}$$

Clearly, Option-Viterbi has a computation complexity of $O(T \times K^2)$ for a T-step trajectory with K options. By back-tracing o_{t-1} that induces the maximization on $\hat{\alpha}(o_t)$ at each time step, the option-extended expert trajectories $\tilde{\mathbb{D}}_{\text{demo}}$ can finally be acquired.

Although our above option inference process implies that the expert demonstrations are assumed to be generated through a hierarchical policy as the same as the agent, our method is still applicable to the non-hierarchical expert, given the fact that a non-hierarchical expert can be imitated by a hierarchical agent (with fewer options activated).

4.5. Summary of Option-GAIL

We briefly give an overview of our proposed Option-GAIL in Algorithm 1. With expert-provided demonstrations $\mathbb{D}_{\text{demo}} = \{\tau_E = (s_{0:T}, a_{0:T})\}$ and initial policy $\tilde{\pi}$, our method alternates the policy optimization and option inference for sufficient iterations.

Algorithm 1 Option-GAIL

```
Data: Expert trajectories $\mathbb{D}_{\text{demo}} = \left\{ \tau_E = \left\{ (s_{0:T}, a_{0:T}) \right\} \right\}$
Input: Initial policy $\tilde{\pi}^0 = (\pi_H^0, \pi_L^0)$
Output: Learned Policy $\tilde{\pi}^\star$
for $n = 0 \cdots N$ do
    E-step: Infer expert options with $\tilde{\pi}^n$ by (9) and get $\tilde{\mathbb{D}}_{\text{demo}}$; sample trajectories with $\tilde{\pi}^n$ and get $\mathbb{D}_{\tilde{\pi}}$.
    M-step: Update $\tilde{\pi}^{n+1}$ by (6).
end
```

4.6. Convergence Analysis

Algorithm 1 has depicted how our method is computed, but it is still unknown if it will converge in the end. To enable the analysis, we first define a surrogate distribution of the options as $q(o_{-1:T})$. We denote Q_n as the expected objective in Equation 5 at iteration n by Algorithm 1, namely, $Q_n = \mathbb{E}_{q(o_{-1:T})}\left[D_f\left(\rho_{\tilde{\pi}}(s,a,o,o')\|\rho_{\tilde{\pi}_E}(s,a,o,o')\right)\right]$. We immediately have the following convergence theorem for Algorithm 1.

Theorem 2 (Convergence) *Algorithm 1 only decreases the objective: $Q_{n+1} \leq Q_n$, if these two conditions hold: (1) $q(o_{-1:T})$ is our option sampling strategy in E-step and is equal to the posterior of the options given current policy $\tilde{\pi}^n$,* i.e. *$q(o_{-1:T}) = P_{\tilde{\pi}^n}(o_{-1:T}|s_{0:T}, a_{0:T})$; (2) M-step definitely decreases the objective in Equation 5 at each iteration.*

The proof is devised by generalizing traditional EM algorithm to the f-divergence minimization problem. We provide the details in Appendix. Since $Q_n \geq 0$, Theorem 2 confirms that Algorithm 1 will converge eventually. The first condition in Theorem 2 guarantees that the objective of Equation 3 is equal to Q_n after E-step, thus Algorithm 1 also helps minimize Equation 3. Besides, if the global minimum is achieved, we have $Q_n = 0 \Rightarrow \rho_{\tilde{\pi}}(s,a,o,o') = \rho_{\tilde{\pi}_E}(s,a,o,o') \Rightarrow \tilde{\pi} = \tilde{\pi_E}$.

In our implementation, as mentioned in § 4.4, we adopt the maximized sampling process instead of the posterior sampling that is required by Theorem 2, as, in this way, we find that it still maintains the convergence in our experiments while reducing the computation complexity with only sampling one option series per trajectory.

5. Experiments

We evaluate our model on four widely used robot learning benchmarks with locomotion and manipulation tasks. We first compare our model against other counterparts that do not employ hierarchical structure or self-explorations. Then we provide an ablated study to understand the impact of each component in our model.

5.1. Environments

Four environments are used for evaluations:

Hopper-v2 and **Walker2d-v2:** The Hopper-v2 and the Walker2d-v2 are two standardized continuous-time locomotion environments implemented in the OpenAI Gym (Brockman et al., 2016) with the MuJoCo (Todorov et al., 2012) physics simulator. On these two tasks, the robot should move toward the desired direction by moving its leg periodically. We use expert demonstration containing 1,000 time-steps for learning on Hopper-v2 environment and 5,000 time-steps for learning on the Walker2d-v2 environment. Both of the expert demonstrations are generated by a policy trained by PPO (Schulman et al., 2017).

AntPush-v0: The AntPush-v0 is a navigation/locomotion task proposed in Nachum et al. (2018), where an Ant robot is required to navigate into a room that is blocked by a movable obstacle, as shown in Figure 10. Specifically, the robot should first navigate to and push away the obstacle and then go into the blocked room. For better comparing the learning performance, we slightly extend the original binary reward as $r_t = -\Delta\|\text{pos}_t - \text{tar}\|_2^2 + 100 \times \mathbb{1}_{\text{pos}_t=\text{tar}}$ where pos is the position of the robot and tar is the location of the blocked room. We use expert demonstrations containing 50,000 time-steps for learning in this environment, where the policy is trained with DAC (Zhang & Whiteson, 2019) regarding our modified reward.

CloseMicrowave2: The Closemicrowave2 is a more challenging robot operation environment in RLBench (James et al., 2020). A 6-DoF robot arm with a gripper is required to reach and close the door of a microwave by controlling the increments on joint positions of the robot arm, as shown in Figure 10. The reward is defined as $r_t = -\Delta\theta_t + 100 \times \mathbb{1}_{\theta_t=0}$, where θ denotes the rotation angle of the door, We use expert demonstrations containing 1,000 time-steps for learning in this environment generated by a handcrafted feedback controller.

5.2. Comparative Evaluations

To illustrate the effectiveness of the proposed method, we contrast several popular imitation learning baselines, including: 1) **supervised Behavior Cloning (BC)** (Pomerleau, 1988) that do not contain hierarchical structure or self-exploration; 2) **GAIL** (Ho & Ermon, 2016) that uses self-exploration without hierarchical structure; 3) **Hierarchical Behavioral Cloning (H-BC)** (Zhang & Paschalidis, 2020), which builds upon the Option structure, but optimizing both the high- and low-level policies by directly maximizing the probability of the observed expert trajectories without any self-exploration. This baseline can also be regarded as optimizing Equation 9 with forward-backward messages; 4) a variant of GAIL, denoted as **GAIL-HRL** that updates hierarchical policies according to Equation 3, where the immediate imitation reward $r_t = -\log D_{\theta'}(s_t, a_t)$ is used instead. GAIL-HRL can be regarded as a simplified version of our Option-GAIL without using options in occupancy measurement matching, and it is designed for justifying the necessity of involving options during the whole occupancy measurement matching procedure.

We employ the same demonstration trajectories, network structures, and option numbers on each environment for fair comparisons. Specifically, we allow 4 available op-

Table 1. Comparative results. All results are measured by the average **maximum average reward-sum** among different trails.

	Hopper-v2	Walker2d-v2	AntPush-v0	CloseMicrowave2
Demos $(s, a) \times T$	$(\mathbb{R}^{11}, \mathbb{R}^{3}) \times 1\text{k}$	$(\mathbb{R}^{17}, \mathbb{R}^{6}) \times 5\text{k}$	$(\mathbb{R}^{107}, \mathbb{R}^{8}) \times 50\text{k}$	$(\mathbb{R}^{101}, \mathbb{R}^{8}) \times 1\text{k}$
Demo Reward	3656.17±0.0	5005.80±36.18	116.60±14.07	149.68 ± 16.29
BC	275.93±31.09	589.17±90.81	4.60±2.72	26.03±2.33
GAIL	535.29±7.19	2787.87±2234.46	54.82±4.81	39.14±12.88
H-BC	970.91±72.69	3093.56±107.11	89.23±1.37	89.54±15.44
GAIL-HRL	3697.40±1.14	3687.63±982.99	20.53±6.89	56.95±25.74
Ours	**3700.42±1.70**	**4836.85±100.09**	**95.00±2.70**	**100.74±21.33**

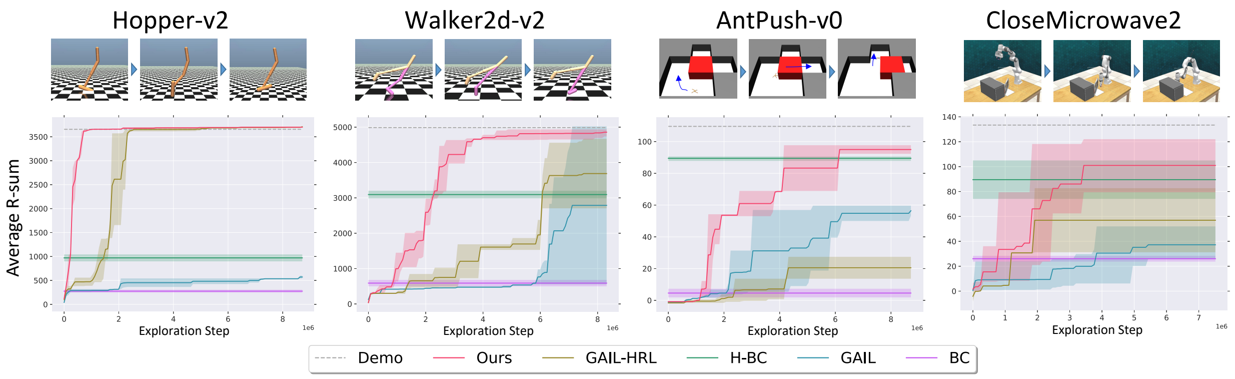

Figure 3. comparison of learning performance on four environments. The environments and the task designs are illustrated on the top of the figure with more details provided in § 5.1. We compare the **maximum average reward-sums** vs. exploration steps on different environments. The solid line indicates the average performance among several trials under different random seeds, while the shade indicates the range of the maximum average reward-sums over different trials.

tion classes for all environments, a Multi-Layer Perception(MLP) with hidden size (64, 64) to implement the policies of both levels on Hopper-v2, Walker2d-v2, AntPush-v0, and (128, 128) on Closemicrowave2; the discriminator is realized by an MLP with hidden size (256, 256) on all environments. For methods that do not use self-exploration, we train the policy for 100 epochs and then evaluate it by average reward-sums; for methods that rely on self-exploration, we update the policy and record the average reward-sum every 4096 environmental steps, and record maximum average reward-sums over the whole training period. The evaluation results are displayed in Figure 10.

Obviously, our method gains faster convergence speed and better eventual performance than all baselines in general, as illustrated in Figure 10. For example, on CloseMicrowave2, which is clearly a long-horizon task, our Option-GAIL converges to a remarkably larger reward than all others. Besides, by introducing the option-based hierarchical structure, Option-GAIL, H-BC and GAIL-HRL are superior to the counterparts that use single-level policy, namely, BC and GAIL. When demonstrations are limited, for instance, on Hopper-v2, Walker2d-v2, and Closemicrowave2, GAIL-like

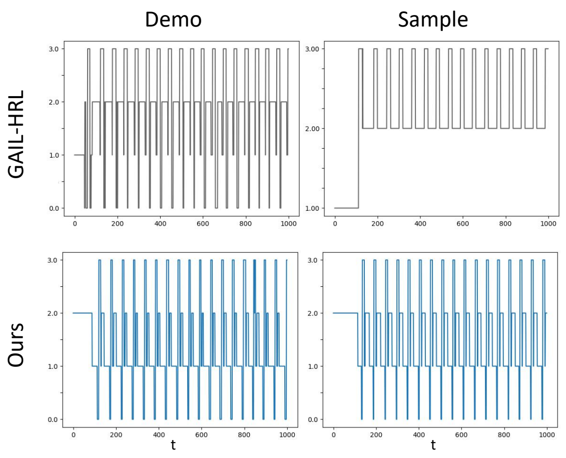

Figure 4. Visualization of the options activated at each step, learned respectively by our proposed method (blue) and GAIL-HRL (gray) on Walker2d-v2. 'Demo' denotes the options inferred from the expert, and 'Sample' denotes the options used by agent when doing self-explorations. The effectiveness of our proposed method on regularizing the option switching is obvious by comparing the consistent switching tendencies between Demo and Sample.

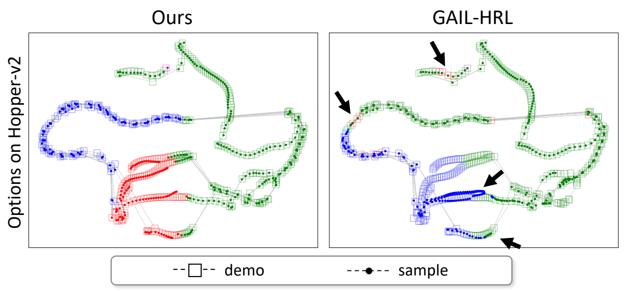

Figure 5. Visualization of the state-action and options on Hopper-v2 environment by t-SNE (Maaten & Hinton, 2008). The dots and squares in the figure indicate the state-action pairs generated by agent and expert on each time step, respectively, and the gray lines connect time-adjacent points. The color indicates the option activated at each time-step. Arrows point to the miss-matched option switching behaviors between agent and expert demonstration by GAIL-HRL.

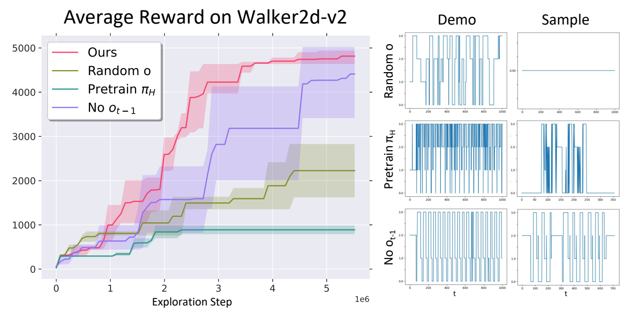

Figure 6. Ablations on individual components of our proposed method. The curve on the left side compares the **maximum average reward-sum** within a given range of exploration steps. The options used by agent and inferred from expert are presented one the right side.

methods including Option-GAIL and GAIL-HRL outperform their BC-like counterpart H-BC, probably thanks to the reduction of compounding errors by self-explorations. On AntPush-v0, the advantage of our method over H-BC is reduced. We conjecture this is because H-BC is data-hungry, and it can become better when sufficient demonstrations are fed. Comparing with GAIL, with the help of hierarchical modeling, our method can successfully imitate the behavior of the expert with fewer self-explorations.

To examine if our proposed method actually regularizes the options switching between expert and agent, Figure 9 illustrates the options including that are inferred from expert and that generated by agent on Walker2d-v2 with more examples deferred to Appendix. It is observed in our method that the expert's options are consistent with the agent's, while for GAIL-HRL, a dramatic conflict exists between expert and agent. This result confirms the benefit of our method since we explicitly minimize the discrepancy of the option-extended occupancy measurement between expert and agent in Equation 5.

5.3. Ablations and Analysis

In this section, we perform an in-depth analysis on each component of our model.

The necessity of using option-extended occupancy: We geometrically assess the difference between our Option-GAIL and GAIL-HRL on Walker2d-v2. To do so, we visualize the trajectories of the expert (squares) and the agent (dots) as well as their options (color) at each time step. Different colors indicate the different options used by the low-policy, and the expert's options are inferred with the agent's policy. The visualizations in Figure 5 read that the agent's trajectory by Option-GAIL almost overlaps with the demonstration, whereas the one by GAIL-HRL sometimes tracks a wrong direction with respect to the expert. Moreover, in Option-GAIL, the options of the expert and agent always switch collaboratively with each other, while GAIL-HRL occasionally derives inconsistent options (highlighted by an arrow) between the expert and agent. This phenomenon once again verifies the necessity of performing the option-extended occupancy measurement matching in Equation 5. If otherwise conducting the occupancy measurement matching in Equation 3, it will give rise to ambiguities on the learned policies and eventually affect the learning performance.

Ablations on individual components: To verify the rationality of the model design, we test different variants of our method in Figure 6.

1) Random o. In this case, we generate the expert's options by random without option inference. As observed, the random options mislead policy learning and thus yield worse performance. We further display the option switching in the right sub-figure in Figure 6. The high-level policy tends to be conservative on switching since the sampling options always keep unchanged, making our model less expressive to fit the expert's hierarchical policy.

2) No o_{t-1}. we simply omit the o_{t-1} in Equation 6 and implement the discriminator with only (s_t, a_t, o_t). Clearly, without the option information from the last step, the high-level policy is not fully regularized and cannot capture the option dynamics, thereby delivering a performance drop compared to the full-option version.

3) Pretrain π_H. We pre-train the high-level policy using H-BC for 50 epochs and then fix it during the subsequent learning procedure, which can be regarded as a version of the two-stage method by Sharma et al. (2018). Such ablation explains it is indeed demanded to simultaneously learn the high- and low-level policies end-to-end by examining the results in Figure 6. In the optimization perspective, the two-level policies are coupled with each other and will be improved better under an alternative training fashion.

In summary, all the results here support the optimal choice of our design.

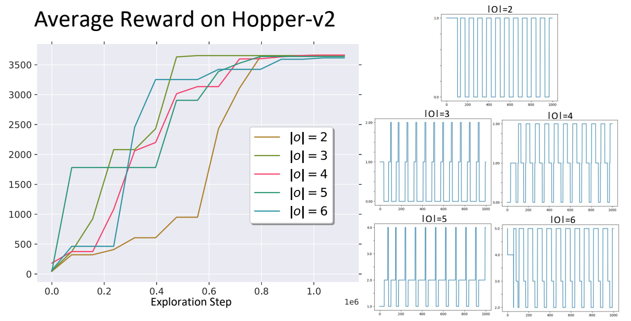

Figure 7. Ablations on the total number of available option classes. Left: how different value of $|o|$ influences the **maximum average reward-sum** within a given range of exploration steps on Hopper-v2 environment. Right: actual option transition for different $|o|$.

Ablations on the number of available option classes: Throughout our above experiments, we set the number of option classes $|o| = 4$ and find it works generally. In fact, changing $|o|$ does not essentially change our performance if $|o|$ is sufficiently large, as our algorithm will automatically inactivate redundant options (similar sub-task switching will be clustered into the same option class). For illustrating such robustness, we evaluate our method with $|o| \in \{2 \cdots 6\}$ on the Hopper-v2 environment. As observed from Figure7, when $|o| \geq 3$, all variants share similar convergence behavior and option transition: the number of activated options is observed as 3 for all cases. When $|o| \geq 2$, the convergence becomes worse, probably due to its less expressivity.

6. Discussion and Conclusion

In this paper, we have presented Option-GAIL, a theoretically-grounded framework that integrates hierarchical modeling with option occupancy measurement matching. We then provide an EM-like algorithm for training the policies of Option-GAIL with a provable guaranteed training convergence. Comparing to existing HIL methods, Option-GAIL can regularize both the high- and low-level policies to better fit the hierarchical behavior of the expert. Experiments on robotic locomotion and manipulation benchmarks demonstrate the effectiveness of the proposed Option-GAIL over other counterparts; Option-GAIL works well particularly for the tasks consisting of evidently separable sub-tasks. Our method is generally powerful and can be enhanced by absorbing external knowledge. For example, it could be a future work direction to consider a long-horizon plan as the prior knowledge when inferring options.

Acknowledgement

This research was primary funded by National Key R&D Program of China (Grant No.20191241501). It was also partially funded by THU-Bosch JCML center (Grant No.20193000122). Meanwhile, this work is jointly sponsored by CAAI MindSpore Open Fund, China Postdoctoral Science Foundation (Grant No.2020M670337), and the National Natural Science Foundation of China (Grant No. 62006137). We would like to thank Chao Yang and Yu Luo for their generous help and insightful advice.

References

Atkeson, C. G. and Schaal, S. Robot learning from demonstration. In *Proc. Int. Conf. Mach. Learn.*, 1997.

Bacon, P.-L., Harb, J., and Precup, D. The option-critic architecture. In *Proc. AAAI Conf. Artificial Intell.*, 2017.

Baum, L. E. et al. An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities*, 3(1):1–8, 1972.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. OpenAI Gym. *arXiv preprint arXiv:1606.01540*, 2016.

Byrne, R. W. and Russon, A. E. Learning by imitation: A hierarchical approach. *Behavioral and brain sciences*, 21 (5):667–684, 1998.

Daniel, C., Van Hoof, H., Peters, J., and Neumann, G. Probabilistic inference for determining options in reinforcement learning. *Machine Learning*, 104(2-3):337–357, 2016.

Fei, C., Wang, B., Zhuang, Y., Zhang, Z., Hao, J., Zhang, H., Ji, X., and Liu, W. Triple-GAIL: A multi-modal imitation learning framework with generative adversarial nets. *Proc. Int. Joint Conf. Artificial Intell.*, 2020.

Ghasemipour, S. K. S., Zemel, R., and Gu, S. A divergence minimization perspective on imitation learning methods. In *Proc. Conf. Robot. Learn.*, 2020.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. *Proc. Advances in Neural Inf. Process. Syst.*, 2014.

Hamidi, M., Tadepalli, P., Goetschalckx, R., and Fern, A. Active imitation learning of hierarchical policies. In *Proc. Int. Joint Conf. Artificial Intell.*, 2015.

Henderson, P., Chang, W.-D., Bacon, P.-L., Meger, D., Pineau, J., and Precup, D. OptionGAN: Learning joint reward-policy options using generative adversarial inverse reinforcement learning. In *Proc. AAAI Conf. Artificial Intell.*, 2018.

Ho, J. and Ermon, S. Generative adversarial imitation learning. In *Proc. Advances in Neural Inf. Process. Syst.*, 2016.

James, S., Ma, Z., Arrojo, D. R., and Davison, A. J. RLBench: The robot learning benchmark & learning environment. *IEEE Robot. Auto. Letters*, 5(2):3019–3026, 2020.

Kipf, T., Li, Y., Dai, H., Zambaldi, V., Sanchez-Gonzalez, A., Grefenstette, E., Kohli, P., and Battaglia, P. CompILE: Compositional imitation learning and execution. In *Proc. Int. Conf. Mach. Learn.*, 2019.

Le, H., Jiang, N., Agarwal, A., Dudik, M., Yue, Y., and Daumé, H. Hierarchical imitation and reinforcement learning. In *Proc. Int. Conf. Mach. Learn.*, 2018.

Lee, S.-H. and Seo, S.-W. Learning compound tasks without task-specific knowledge via imitation and self-supervised learning. In *Proc. Int. Conf. Mach. Learn.*, 2020.

Li, C., Song, D., and Tao, D. The skill-action architecture: Learning abstract action embeddings for reinforcement learning. *submissions of ICLR*, 2021.

Maaten, L. v. d. and Hinton, G. Visualizing data using t-sne. *J. Mach. Learn. Res.*, 9(Nov):2579–2605, 2008.

Nachum, O., Gu, S. S., Lee, H., and Levine, S. Data-efficient hierarchical reinforcement learning. In *Proc. Advances in Neural Inf. Process. Syst.*, 2018.

Pomerleau, D. A. ALVINN: An autonomous land vehicle in a neural network. In *Proc. Advances in Neural Inf. Process. Syst.*, 1988.

Ross, S., Gordon, G., and Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proc. Int. Conf. Artificial Intell. & Stat.*, 2011.

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

Sharma, M., Sharma, A., Rhinehart, N., and Kitani, K. M. Directed-Info GAIL: Learning hierarchical policies from unsegmented demonstrations using directed information. In *Proc. Int. Conf. Learn. Representations*, 2018.

Sharma, P., Pathak, D., and Gupta, A. Third-person visual imitation learning via decoupled hierarchical controller. In *Proc. Advances in Neural Inf. Process. Syst.*, 2019.

Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2): 181–211, 1999.

Syed, U., Bowling, M., and Schapire, R. E. Apprenticeship learning using linear programming. In *Proc. Int. Conf. Mach. Learn.*, 2008.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots & Systems*, 2012.

Viterbi, A. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269, 1967.

Yu, T., Abbeel, P., Levine, S., and Finn, C. One-shot hierarchical imitation learning of compound visuomotor tasks. *arXiv preprint arXiv:1810.11043*, 2018.

Zhang, S. and Whiteson, S. DAC: The double actor-critic architecture for learning options. In *Proc. Advances in Neural Inf. Process. Syst.*, 2019.

Zhang, Z. and Paschalidis, I. Provable hierarchical imitation learning via em. *arXiv preprint arXiv:2010.03133*, 2020.