Option-Aware Adversarial Inverse Reinforcement Learning for Robotic Control

Jiayu Chen[1], Tian Lan[2], and Vaneet Aggarwal[1]

Abstract— **Hierarchical Imitation Learning (HIL) has been proposed to recover highly-complex behaviors in long-horizon tasks from expert demonstrations by modeling the task hierarchy with the option framework. Existing methods either overlook the causal relationship between the subtask and its corresponding policy or cannot learn the policy in an end-to-end fashion, which leads to suboptimality. In this work, we develop a novel HIL algorithm based on Adversarial Inverse Reinforcement Learning and adapt it with the Expectation-Maximization algorithm in order to directly recover a hierarchical policy from the unannotated demonstrations. Further, we introduce a directed information term to the objective function to enhance the causality and propose a Variational Autoencoder framework for learning with our objectives in an end-to-end fashion. Theoretical justifications and evaluations on challenging robotic control tasks are provided to show the superiority of our algorithm. The codes are available at https://github.com/LucasCJYSDL/HierAIRL.**

I. INTRODUCTION

Reinforcement Learning (RL) has achieved impressive performance in a variety of scenarios, such as games [1], [2], [3] and robotic control [4], [5]. However, most of its applications rely on carefully-crafted, task-specific reward signals to drive exploration and learning, limiting its use in real-life scenarios. In this case, Imitation Learning (IL) methods have been developed to acquire a policy for a certain task based on the corresponding expert demonstrations (e.g., trajectories of state-action pairs) rather than reinforcement signals. However, complex long-horizon tasks can often be broken down and processed as a series of subtasks which can serve as basic skills for completing various compound tasks. In this case, learning a single monolithic policy with IL to represent a structured activity can be challenging. Therefore, Hierarchical Imitation Learning (HIL) has been proposed to recover a two-level policy for a long-horizon task from the demonstrations. Specifically, HIL trains low-level policies (i.e., skills) for accomplishing the specific control for each sub-task, and a high-level policy for scheduling the switching of the skills. Such a hierarchical policy, which is usually formulated with the option framework [6], makes full use of the sub-structure between the parts within the activity and has the potential for better performance.

The most state-of-the-art (SOTA) works on HIL [7], [8] are developed based on Generative Adversarial Imitation Learning (GAIL) [9] which is a widely-adopted IL algorithm. In [7], they additionally introduce a directed information [10] term to the GAIL objective function. In this way, their method can enhance the causal relationship between the skill choice and the corresponding state-action sequence to form low-level policies for each subtask, while encouraging the hierarchical policy to generate trajectories similar to the expert ones in distribution through the GAIL objectives. However, they update the high-level and low-level policy on two separate stages. Specifically, the high-level policy is learned with behavioral cloning, which is a supervised IL algorithm and vulnerable to compounding errors [11], and remains fixed during the low-level policy learning with GAIL. Given that the two-level policies are coupled with each other, such a two-staged paradigm potentially leads to sub-optimal solutions. On the other hand, in [8], they propose to learn a hierarchical policy by option-occupancy measurement matching, that is, imitating the joint distribution of the options, states and actions of the expert demonstrations rather than only matching the state-action distributions like GAIL. However, they overlook the causal relationship between the subtask structure and the policy hierarchy, so the recovered policy may degenerate into a poorly-performed monolithic policy for the whole task, especially when the option annotations of the expert demonstrations are not provided.

In this work, we propose a novel HIL algorithm – Hierarchical Adversarial Inverse Reinforcement Learning (H-AIRL), which integrates the SOTA IL algorithm Adversarial Inverse Reinforcement Learning (AIRL) [12] with the option framework through an objective based on the directed information. Compared with GAIL, AIRL is able to recover the expert reward function along with the expert policy and has more robust and stable performance for challenging robotic tasks [12], [13], [14]. From the algorithm perspective, our contributions are as follows: **(1)** We propose a practical lower bound of the directed information between the option choice and the corresponding state-action sequences, which can be modeled as a variational posterior and updated in a Variational Autoencoder (VAE) [15] framework. This design enables our algorithm to update the high-level and low-level policy at the same time in an end-to-end fashion, which is an improvement compared with [7]. **(2)** We redefine the AIRL objectives on the extended state and action space, in order to directly recover a hierarchical policy from the demonstrations. **(3)** We provide an Expectation-Maximization (EM) [16] adaption of our algorithm so that it can be applied to the expert demonstrations without the sub-task annotations (i.e., unsegmented expert data) which are easier to obtain in

[1]J. Chen and V. Aggarwal are with the School of Industrial Engineering, Purdue University, West Lafayette, IN 47907, USA chen3686@purdue.edu, vaneet@purdue.edu. V. Aggarwal is also with the CS Department, KAUST, Thuwal, Saudi Arabia.

[2]T. Lan is with the Department of Electrical and Computer Engineering, George Washington University, Washington D.C., 20052, USA tlan@gwu.edu

practice. Further, we provide solid theoretical justification of the three folds mentioned above, and comparisons of our algorithm with SOTA HIL and IL baselines on multiple Mujoco [17] continuous control tasks where our algorithm significantly outperforms the others.

II. RELATED WORK

Imitation Learning. Imitation learning methods [18] seek to learn to perform a task from expert demonstrations, where the learner is given only samples of trajectories from the expert and is not provided any reinforcement signals, such as the environmental rewards which are usually hard to acquire in real-life scenarios. There are two main branches for this algorithm setting: behavioral cloning (BC) [19], which learns a policy as a supervised learning problem over state-action pairs from expert trajectories, and inverse reinforcement learning (IRL) [20], which first infers a reward function under which the expert is uniquely optimal and then recovers the expert policy based on it. Behavioral cloning only tends to succeed with large amounts of data, due to the compounding error caused by covariate shift [21]. Inverse reinforcement learning, while avoiding the compounding error, is extremely expensive to solve and scale, since it requires reinforcement learning to get the corresponding optimal policy in each iteration of updating the reward function. GAIL [9] and AIRL [12] have been proposed to scale IRL for complex high-dimensional control tasks. They realize IRL through an adversarial learning framework, where they alternatively update a policy and discriminator network. The discriminator serves as the reward function and learns to differentiate between the expert demonstrations and state-action pairs from the learned policy. While, the policy is trained to generate trajectories that are difficult to be distinguished from expert data by the discriminator. Mathematical details are provided in Section III-A. AIRL explicitly recovers the reward function and provides more robust and stable performance among challenging tasks [12], [13], [14], which is chosen as our base algorithm for extension.

Hierarchical Imitation Learning. Given the nature of subtask decomposition in long-horizon tasks, hierarchical imitation learning can achieve better performance than imitation learning by forming micro-policies for accomplishing the specific control for each subtask first and then learning a macro-policy for scheduling among the micro-policies. The micro-policies (a.k.a., skills) in RL can be modeled with the option framework proposed in [6], which extends the usual notion of actions to include options — the closed-loop policies for taking actions over a period of time. We provide further details about the option framework in Section III-B. Through integrating IL with the options, the hierarchical versions of the IL methods mentioned above have been developed, including hierarchical behavioral cloning (HBC) and hierarchical inverse reinforcement learning (HIRL). In HBC, they train a policy for each subtask through supervised learning with the corresponding state-action pairs, due to which the subtask annotations need to be provided or inferred. In particular, the methods proposed in [22], [23] require segmented data with the subtask information. While, in [24], [25], they infer the subtask information as the hidden variables in a Hidden Markov Model [26] and solve the HBC as an MLE problem with the Expectation–Maximization (EM) algorithm [16]. Despite its theoretical completeness, HBC is also vulnerable to compounding errors in case of limited demonstrations. On the other hand, the HIRL methods proposed in [7], [8] have extended GAIL with the option framework to recover the hierarchical policy (i.e., the high-level and low-level policies mentioned above) from unsegmented expert data. Specifically, in [7], they introduce a regularizer into the original GAIL objective function to maximize the directed information between generated trajectories and the subtask/option annotations. However, the high-level and low-level policies are trained in two separate stages in their approach, which will inevitably lead to convergence with a poor local optimum. As for the approach proposed in [8] which claims to outperform [7] and HBC, it replaces the occupancy measurement in GAIL, which measures the distribution of the state-action pairs, with option-occupancy measurement to encourage the hierarchical policy to generate state-action-option tuples with similar distribution to the expert demonstrations. However, they do not adopt the directed information objective to enhance the causal relationship between the option choice and the corresponding state-action sequence. In this paper, we propose a new HIL algorithm based on AIRL, which takes advantage of the directed information objective and updates the high-level and low-level policies in an end-to-end fashion. Moreover, we provide theoretical justification of our algorithm, and demonstrate its superiority on challenging robotic control tasks.

III. BACKGROUND

In this section, we introduce the background knowledge of our work, including AIRL and the One-step Option Framework. These are defined with the Markov Decision Process (MDP), denoted by $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mu, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition function, $\mu : \mathcal{S} \to [0, 1]$ is the distribution of the initial state, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in (0, 1]$ is the discount factor.

A. Adversarial Inverse Reinforcement Learning

Inverse reinforcement learning (IRL) [20] aims to infer an expert's reward function from demonstrations, based on which the policy of the expert can be recovered. As a representative, Maximum Entropy IRL [27] solves it as a maximum likelihood estimation (MLE) problem shown as Equation (1). $\tau_E \triangleq (S_0, A_0, \cdots, S_{T-1}, A_{T-1}, S_T)$ denotes the expert trajectory, i.e., a sequence of state-action pairs of horizon T. Z_ϑ is the partition function defined as $Z_\vartheta = \int \widehat{P}_\vartheta(\tau_E) d\tau_E$ (continuous $\mathcal{S}$ and $\mathcal{A}$) or $Z_\vartheta = \sum_{\tau_E} \widehat{P}_\vartheta(\tau_E)$ (discrete $\mathcal{S}$ and $\mathcal{A}$).

$$\begin{aligned} &\max_\vartheta \mathbb{E}_{\tau_E} \left[\log P_\vartheta(\tau_E)\right], \ P_\vartheta(\tau_E) = \widehat{P}_\vartheta(\tau_E)/Z_\vartheta \\ &\widehat{P}_\vartheta(\tau_E) = \mu(S_0) \prod_{t=0}^{T-1} \mathcal{P}(S_{t+1}|S_t, A_t) \exp(\mathcal{R}_\vartheta(S_t, A_t)) \end{aligned} \tag{1}$$

Since Z_ϑ is intractable for the large-scale state-action space, the authors of [12] propose Adversarial Inverse Reinforcement Learning (AIRL) to solve this MLE problem in a sample-based manner. They realize this through alternatively training a discriminator D_ϑ and policy network π in an adversarial setting. Specifically, the discriminator is trained by minimizing the cross-entropy loss between the expert demonstrations τ_E and generated samples τ by π:

$$\min_{\vartheta} -\sum_{t=0}^{T-1} \mathbb{E}_{\tau_E}[\log D_\vartheta(S_t, A_t)] - \mathbb{E}_{\tau}[\log(1 - D_\vartheta(S_t, A_t))] \tag{2}$$

where $D_\vartheta(S, A) = \exp(f_\vartheta(S, A))/[\exp(f_\vartheta(S, A)) + \pi(A|S)]$. Meanwhile, the policy π is trained with off-the-shelf RL algorithms using the reward function defined as $\log D_\vartheta(S, A) - \log(1 - D_\vartheta(S, A))$. Further, they justify that, at optimality, $f_\vartheta(S, A)$ can serve as the recovered reward function $\mathcal{R}_\vartheta(S, A)$ and π is the recovered expert policy which maximizes the entropy-regularized objective: $\mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{T-1} \mathcal{R}_\vartheta(S_t, A_t) - \log \pi(A_t|S_t)\right]$.

B. One-step Option Framework

As proposed in [6], an option $Z \in \mathcal{Z}$ can be described with three components: an initiation set $I_Z \subseteq \mathcal{S}$, an intra-option policy $\pi_Z(A|S) : \mathcal{S} \times \mathcal{A} \to [0, 1]$, and a termination function $\beta_Z(S) : \mathcal{S} \to [0, 1]$. An option Z is available in state S if and only if $S \in I_Z$. Once the option is taken, actions are selected according to π_Z until it terminates stochastically according to β_Z, i.e., the termination probability at the current state. A new option will be activated in this call-and-return style by a high-level policy $\pi_{\mathcal{Z}}(Z|S) : \mathcal{S} \times \mathcal{Z} \to [0, 1]$ once the last option terminates. In this way, $\pi_{\mathcal{Z}}(Z|S)$ and $\pi_Z(A|S)$ constitute a hierarchical policy for a certain task. However, it's inconvenient to deal with the initiation set I_Z and termination function β_Z while learning this hierarchical policy. Thus, in [28], [8], they adopt the one-step option framework. It's assumed that each option is available in each state, i.e., $I_Z = \mathcal{S}, \forall Z \in \mathcal{Z}$. Also, the high-level and low-level (i.e., intra-option) policy are redefined as π_θ and π_ϕ respectively:

$$\begin{aligned} \pi_\theta(Z|S, Z') &= \beta_{Z'}(S)\pi_{\mathcal{Z}}(Z|S) + [(1 - \beta_{Z'}(S))\delta_{Z=Z'}] \\ \pi_\phi(A|S, Z) &= \pi_Z(A|S) \end{aligned} \tag{3}$$

where Z' denotes the option in the last time step and $\delta_{Z=Z'}$ is the indicator function. We can see that if the previous option terminates (with probability $\beta_{Z'}(S)$), the agent will select a new option according to $\pi_{\mathcal{Z}}(Z|S)$; otherwise, it will stick to Z'. With the new definition and assumption, we can optimize the hierarchical policy π_θ and π_ϕ without the extra need to justify the exact beginning and breaking condition of each option. Nevertheless, $\pi_\theta(Z|S, Z')$ still includes two separate parts, i.e., $\beta_{Z'}(S)$ and $\pi_{\mathcal{Z}}(Z|S)$, and due to the indicator function, the update gradients of $\pi_{\mathcal{Z}}$ will be blocked/gated by the termination function $\beta_{Z'}(S)$. In this case, the authors of [29] propose to marginalize the termination function away, and instead implement $\pi_\theta(Z|S, Z')$ as an end-to-end neural network (NN) with the Multi-Head Attention (MHA) mechanism [30] which enables their algorithm to temporally extend options in the absence of the termination function. We provide more details on MHA and the structure design of π_θ and π_ϕ in Appendix I [1]. With the marginalized one-step option framework, we only need to train the two NN-based policy, i.e., π_θ and π_ϕ. In particular, we adopt the Hierarchical Reinforcement Learning algorithm, i.e., SA, proposed in [29] to learn π_θ and π_ϕ.

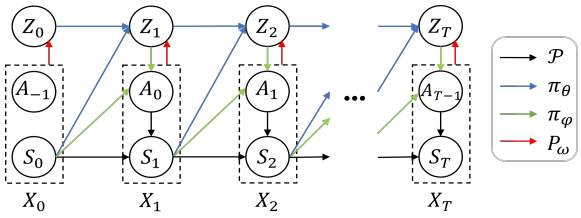


Fig. 1. Illustration of the probabilistic graphical model and its implementation with the one-step option model.

IV. PROPOSED APPROACH

A. Optimization with the Directed Information Objective

Our work focuses on learning a hierarchical policy from expert demonstrations through integrating the one-step option framework with AIRL. In this section, we define the directed information objective function for training the hierarchical policy, fit it with the one-step option model, and propose how to optimize it in an end-to-end fashion with an RNN-based VAE structure, which is part of our novelty and contribution.

As mentioned in Section III-B, the hierarchical policy agent will first decide on its option choice Z using the high-level policy π_θ and then select the primitive action based on the low-level policy π_ϕ corresponding to Z, when observing a new state. In this case, the policy learned should be conditioned on the option choice Z, and the option choice is specific to each timestep $t \in \{0, \cdots, T\}$, so we view the option choices $Z_{0:T}$ as the local latent contexts in a probabilistic graphical model shown as Figure 1. It can be observed from Figure 1 that the local latent context $Z_{0:T}$ has a directed causal relationship with the trajectory $X_{0:T} = (X_0, \cdots, X_T) = ((A_{-1}, S_0), \cdots, (A_{T-1}, S_T))$, where A_{-1} is the dummy variable. Inspired by information theory [10], [7], this kind of connection can be established by maximizing the directed information (a.k.a., casual information) flow from the trajectory to the latent factors of variation within the trajectory, i.e., $I(X_{0:T} \to Z_{0:T})$, which is defined as:

$$\begin{aligned} I(X_{0:T} \to Z_{0:T}) &= \sum_{t=1}^{T} \left[H(Z_t|Z^{t-1}) - H(Z_t|X^t, Z^{t-1})\right] \\ &= \sum_{t=1}^{T} [H(Z_t|Z^{t-1}) + \sum_{X^t, Z^t} P(X^t, Z^t) \log P(Z_t|X^t, Z^{t-1})] \end{aligned} \tag{4}$$

[1] All the appendices are available in the extended version of our paper at https://github.com/LucasCJYSDL/HierAIRL/blob/main/ICRA.pdf

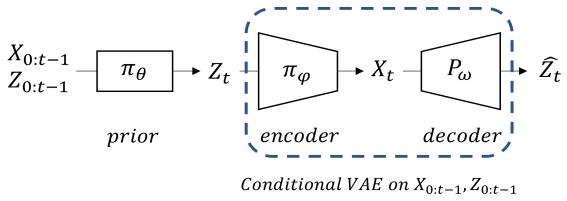


Fig. 2. The analogy of our learning framework with the VAE structure.

where the first equality follows the general definition of directed information. Note that we use X^t to represent $X_{0:t}$ for simplicity, and so on. In the above equations, $H(\cdot)$ denotes the entropy [31] and we assume the trajectory $X_{0:T}$ to be discrete in order to simplify the notations, but it can be either discrete or continuous in use. It is infeasible to directly optimize the above objective, since it is difficult to calculate the posterior distribution $P(Z_t|X_{0:t}, Z_{0:t-1})$ with only the hierarchical policy, i.e., π_θ and π_ϕ. In this case, we instead maximize its variational lower bound as follows: (Please refer to Appendix II for derivations.)

$$\begin{aligned} L^{DI} \triangleq \sum_{t=1}^{T} [H(Z_t|X^{t-1}, Z^{t-1}) + \\ \sum_{X^t, Z^t} P(X^t, Z^t) \log P_\omega(Z_t|X^t, Z^{t-1})] \end{aligned} \tag{5}$$

where P_ω is the variational estimation of $P(Z_t|X_{0:t}, Z_{0:t-1})$. While, $P(X_{0:t}, Z_{0:t})$ is calculated by: (Please refer to Appendix II for derivations.)

$$\begin{aligned} &\mu(S_0) \prod_{i=1}^{t} P(Z_i|X^{i-1}, Z^{i-1}) P(A_{i-1}|X^{i-1}, Z^i) \mathcal{P}^{S_i}_{S_{i-1}, A_{i-1}} \\ &= \mu(S_0) \prod_{i=1}^{t} \pi_\theta(Z_i|S_{i-1}, Z_{i-1}) \pi_\phi(A_{i-1}|S_{i-1}, Z_i) \mathcal{P}^{S_i}_{S_{i-1}, A_{i-1}} \end{aligned} \tag{6}$$

where μ is the initial state distribution and $\mathcal{P}^{S_i}_{S_{i-1}, A_{i-1}} = \mathcal{P}(S_i|S_{i-1}, A_{i-1})$ is the transition function. Note that the expectation term in L^{DI} can be estimated through Monte Carlo sampling [32] by interacting with the environment using the definition in Equation (6), even though μ and $\mathcal{P}$ are unknown to us. Moreover, the second equality in Equation (6) holds if we adopt the Markov assumption [33], i.e., the conditional probability distribution of a random variable depends only on its parent nodes in the probabilistic graphical model (e.g., Figure 1). As a result, we can fit in the one-step option model (i.e., Equation (3)) introduced in Section III-B, and use L^{DI} as the objective to update the hierarchical policy.

To sum up, we can train the hierarchical policy π_θ and π_ϕ by maximizing the directed information to enhance the directed causal relationship between the latent contexts $Z_{0:T}$ and trajectories $X_{0:T}$ shown as Figure 1. To realize this, we additionally introduce a variational posterior P_ω and update it together with π_θ and π_ϕ in a Variational Autoencoder (VAE) [15] framework. As shown in Figure 2, for the optimization of L^{DI} (Equation 5) which is a VAE-like objective function, at each timestep t, π_ϕ and P_ω form a conditional VAE between Z_t and X_t, which is conditioned on the history, i.e., $(X_{0:t-1}, Z_{0:t-1})$, with the prior distribution of Z_t provided by π_θ. Note that P_ω uses sequential data, i.e., $(X_{0:t}, Z_{0:t-1})$, as input and thus are implemented with RNN, of which the details are in Appendix III. With this framework, we can train the hierarchical policy in an end-to-end manner, instead of learning the high- and low-level policy separately in two training schemes like in [7]. Moreover, in [7], they overlook the first entropy term in the first line of Equation (4) directly and get a negative lower bound of the positive directed information, which is trivial to solve.

B. Hierarchical Adversarial Inverse Reinforcement Learning

Imitation Learning algorithms have been adopted to learn a policy from expert demonstrations rather than based on the reward signals which are usually difficult to acquire in real-life tasks. However, state-of-the-art IL algorithms like AIRL [12] can not be directly adopted to recover a hierarchical policy since they don't take the local latent codes $Z_{0:T}$ into consideration. Thus, we propose a novel hierarchical extension of AIRL, denoted as H-AIRL, as a solution, which is also part of our contribution. Further, we note that it is usually difficult to annotate the local latent codes $Z_{0:T}$ for an expert trajectory $X_{0:T}$, so we propose an Expectation-Maximization (EM) adaption of H-AIRL as well to learn the hierarchical policy based on only the unsegmented expert trajectories, i.e., $\{X_{0:T}\}$.

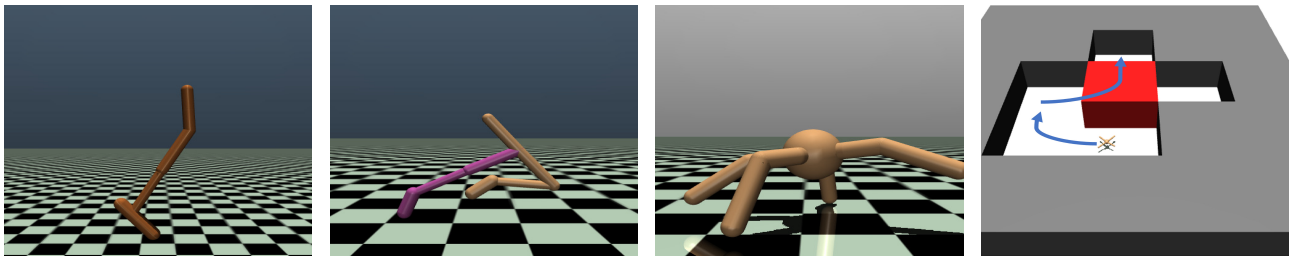
First, we give out the definition of the hierarchical policy. When observing a state S_t at timestep $t \in \{0, \cdots, T-1\}$, the agent needs first to decide on its option choice based on S_t and its previous option choice Z_t using the high-level policy $\pi_\theta(Z_{t+1}|S_t, Z_t)$, and then decide on the action with the corresponding low-level policy $\pi_\phi(A_t|S_t, Z_{t+1})$. Thus, the hierarchical policy can be acquired with the chain rule as:

$$\begin{aligned} \pi_\theta(Z_{t+1}|S_t, Z_t) \cdot \pi_\phi(A_t|S_t, Z_{t+1}) &= \pi_{\theta,\phi}(Z_{t+1}, A_t|S_t, Z_t) \\ &= \pi_{\theta,\phi}(\widetilde{A}_t|\widetilde{S}_t) \end{aligned} \tag{7}$$

where the first equality holds because of the one-step Markov assumption, $\widetilde{S}_t \triangleq (S_t, Z_t)$ and $\widetilde{A}_t \triangleq (Z_{t+1}, A_t)$ denote the extended state and action space respectively.

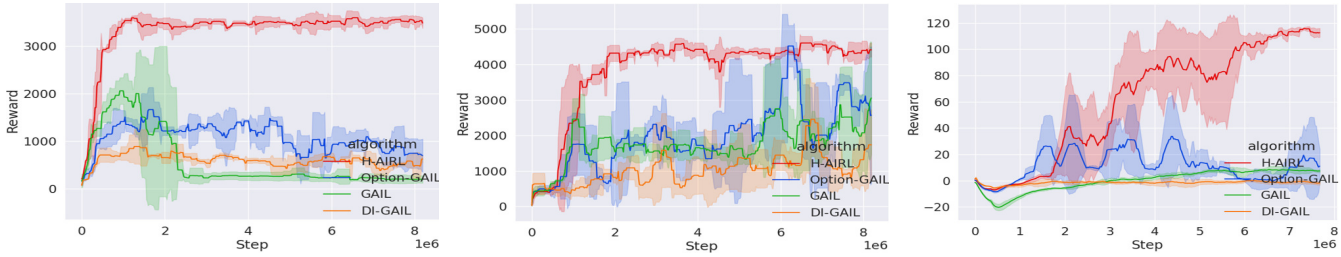
Next, by substituting (S_t, A_t) with $(\widetilde{S}_t, \widetilde{A}_t)$ and τ_E with the hierarchical trajectory $(X_{0:T}, Z_{0:T})$ in Equation (1), we can get an MLE problem shown as Equation (8), from which we can recover the hierarchical reward function and policy. The derivation of (8) is available in Appendix IV.

$$\begin{aligned} &\max_\vartheta \mathbb{E}_{(X_{0:T}, Z_{0:T}) \sim \pi_E(\cdot)} [\log P_\vartheta(X_{0:T}, Z_{0:T})] \\ &P_\vartheta(X_{0:T}, Z_{0:T}) \propto \widehat{P}_\vartheta(X_{0:T}, Z_{0:T}) \\ &= \mu(\widetilde{S}_0) \prod_{t=0}^{T-1} \mathcal{P}(\widetilde{S}_{t+1}|\widetilde{S}_t, \widetilde{A}_t) \exp(\mathcal{R}_\vartheta(\widetilde{S}_t, \widetilde{A}_t)) \\ &= \mu(S_0) \prod_{t=0}^{T-1} \mathcal{P}(S_{t+1}|S_t, A_t) \exp(\mathcal{R}_\vartheta(S_t, Z_t, Z_{t+1}, A_t)) \end{aligned} \tag{8}$$



(a) Hopper (b) Walker (c) Ant (d) AntPusher

Fig. 3. Evaluation tasks built with Mujoco.



(a) Hopper (b) Walker (c) AntPusher

Fig. 4. Comparisons with SOTA HIL and IL algorithms on Mujoco. Each training is repeated three times with different random seeds, and the mean and standard deviation are plotted as the solid lines and shaded areas, respectively. It can be observed that our algorithm, i.e., H-AIRL, performs the best, the GAIL-based algorithms suffer from unstableness, and the two-stage learning of the hierarchical policy in Directed-Info GAIL leads to poor performance.

As mentioned in Section III-A, the MLE problem can be efficiently solved with an adversarial learning framework, which is summarized as Equation (9). At optimality, we can recover the hierarchical reward function (i.e., f_ϑ) and policy (i.e., $\pi_{\theta,\phi}$) of the expert with these objectives, of which the justification is provided in Appendix V.

$$\begin{aligned}
&\min_{\vartheta} -\mathbb{E}_{(X_{0:T},Z_{0:T})\sim\pi_E(\cdot)}\left[\sum_{t=0}^{T-1}\log D_\vartheta(S_t,Z_t,Z_{t+1},A_t)\right]\\
&-\mathbb{E}_{(X_{0:T},Z_{0:T})\sim\pi_{\theta,\phi}(\cdot)}\left[\sum_{t=0}^{T-1}\log(1-D_\vartheta(S_t,Z_t,Z_{t+1},A_t))\right]\\
&\max_{\theta,\phi} L^{IL}=\mathbb{E}_{(X_{0:T},Z_{0:T})\sim\pi_{\theta,\phi}(\cdot)}\sum_{t=0}^{T-1}R^t_{IL}
\end{aligned} \tag{9}$$

In the above equation, the reward is defined as $R^t_{IL} = \log D^t_\vartheta - \log(1-D^t_\vartheta)$ and $D^t_\vartheta = D_\vartheta(S_t,Z_t,Z_{t+1},A_t) = \frac{\exp(f_\vartheta(S_t,Z_t,Z_{t+1},A_t))}{\exp(f_\vartheta(S_t,Z_t,Z_{t+1},A_t))+\pi_{\theta,\phi}(Z_{t+1},A_t|S_t,Z_t)}$.

Usually, we can only acquire the trajectories of state-action pairs, i.e., $X_{0:T}$, from the expert. In this case, we view the latent contexts $Z_{0:T}$ as hidden variables and adopt an EM-style adaption of our algorithm. In the expectation (E) step, we sample possible local latent codes with $Z_{0:T}\sim P_{\overline{\omega}}(\cdot|X_{0:T})$. $P_{\overline{\omega}}$ represents the trained posterior distribution network for $Z_{0:T}$, with the parameter $\overline{\omega}$, i.e., the old parameters before being updated in the M step. Then, in the maximization (M) step, we optimize the objectives shown in Equation (9) for iterations, by replacing the samples in the first term of Equation (9) with $(X_{0:T},Z_{0:T})$ collected in the E step. The theoretical justification of the effectiveness of this EM-like algorithm is provided in Appendix VI.

To sum up, there are in total four networks to learn in our system: the high-level policy π_θ, low-level policy π_ϕ, discriminator D_ϑ, and variational posterior P_ω. D_ϑ can be trained by minimizing the cross entropy loss shown in Equation (9). While, the update of the other three networks should follow:

$$\max_{\theta,\phi,\omega} L=\alpha_1 L^{DI}(\theta,\phi,\omega)+\alpha_2 L^{IL}(\theta,\phi) \tag{10}$$

where $\alpha_{1:2}>0$ are the weights for each objective term and fine-tuned as hyperparameters, L^{DI} and L^{IL} are defined in Equation (5) and (9), respectively. The pseudo code of our algorithm is available at Appendix VII.

V. EVALUATION AND MAIN RESULTS

In this section, we compare H-AIRL with SOTA HIL algorithms: Option-GAIL [8] and Directed-Info GAIL [7], to justify the superiority of our algorithm, and we provide comparisons with SOTA IL algorithms: GAIL [9] to show the importance of hierarchical policy learning for challenging long-horizon tasks. To keep it fair, we use the original implementations of these baseline algorithms provided by the authors. Moreover, we provide ablation study of our algorithm to evaluate the key components of our algorithm design. Specifically, we compare with (i) Option-AIRL: a version of our algorithm by only keeping the AIRL-related term in the objective to update the hierarchical policy, i.e., L^{IL} in Equation (10); (ii) H-GAIL: a variant by replacing the AIRL component of our algorithm with GAIL, of which the details are in Appendix VIII.

TABLE I
NUMERIC RESULTS OF THE ABLATION STUDY

	Expert	H-AIRL (Ours)	Option-AIRL	H-GAIL
Hopper	3139.85 ± 712.02	$\mathbf{3501.81 \pm 110.79}$	1841.19 ± 401.22	2574.06 ± 920.34
Walker	5317.56 ± 99.90	$\mathbf{4354.14 \pm 193.28}$	3951.41 ± 631.48	3812.99 ± 712.83
AntPusher	116.00 ± 3.99	$\mathbf{113.94 \pm 2.54}$	81.58 ± 39.75	80.23 ± 30.29

As shown in Figure 3, these algorithms are evaluated on three challenging continuous control robotic tasks built with Mujoco [17], i.e., Hopper, Walker and AntPusher. Hopper and Walker are locomotion tasks where the robot agents are required to move toward a certain direction by learning to coordinate their legs. Both of them have continuous state and action spaces. Specifically, Hopper has a 11-dim state space and 3-dim action space, and Walker has a 17-dim state space and 6-dim action space. While, the AntPusher needs not only to learn locomotion skills for the Ant agent (Figure 3(c)) but also to learn to navigate into a room that is blocked by a movable box (i.e., the red one in Figure 3(d)). In particular, the Ant agent needs to first navigate to the left side of the box and push it away, and then enter the blocked room to complete the task, which is much more challenging with a 107-dim continuous state space and 8-dim continuous action space. Note that all of the three tasks are long-horizon episodic tasks with a horizon of 1000 time steps. The expert demonstrations for imitation are generated from well-trained policies by PPO [34] with 2×10^7 exploration steps.

A. Main Results

First, we compare our algorithm, i.e., H-AIRL, with the SOTA HIL and IL baselines mentioned above on the three Mujoco tasks. As shown in Figure 4, we plot the change of the episodic rewards (i.e., the sum of the rewards at each time step within an episode) in the training process. Note that these episodic rewards are specifically designed by OpenAI Gym [35] to encourage the Mujoco agents to complete the corresponding tasks as fast as possible at the least control cost, which can be used as evaluation metrics for the agents' learning performance. We repeat each training for three times with different random seeds, plot the average value as the solid lines and the standard deviation as the shaded areas. It can be observed from Figure 4 that H-AIRL outperforms the baselines significantly in terms of both the convergence speed and final performance. Also, Option-GAIL has better performance than GAIL, which shows the advantages of hierarchical policy learning for challenging long-horizon tasks. While, the fluctuations during the training process of GAIL and Option-GAIL show the unstableness of the GAIL-based algorithms. Moreover, it can be observed that Directed-Info GAIL performs even worse than GAIL which does not take advantage of options in the learning process, showing that the separate learning of the high-level and low-level policy (i.e., learning the high-level policy first at the pretraining stage and then fixing it during the low-level policy training) will harm the agents' performance and lead to convergence to a poor local optimum.

Next, we provide comparisons of H-AIRL with Option-AIRL and H-GAIL as the ablation study. As shown in Equation (10), the objective function for updating the hierarchical policy includes two parts, i.e., the directed information term L^{DI} and AIRL term L^{IL}. In order to evaluate the importance of L^{DI}, we implement the baseline Option-AIRL, for which we only keep L^{IL}, i.e., the AIRL objectives on the extended state and action spaces, for updating the hierarchical policy. On the other hand, we replace the AIRL objective with the one defined with GAIL (Equation (24)), denoted as H-GAIL, to show the necessity to adopt AIRL as our base imitation learning algorithm. In Table I, we provide the numeric results of the performance of the expert demonstrations, our algorithm, and the ablation baselines on the three Mujoco benchmarks. To be specific, we repeat the training with each algorithm on each task for three times with different random seeds, and calculate the mean and standard deviation of the episodic rewards after they converge across different random seeds as the metric of their final performance. It can be observed that our algorithm outperforms the baselines in both the average performance and the stableness, showing the effectiveness of the key components of our algorithm design. On the other hand, even the ablation variants of our algorithm, i.e., Option-AIRL and H-GAIL, have better final performance than the baselines shown in Figure 4, which further shows the superiority of our algorithm.

VI. CONCLUSION

Hierarchical Imitation Learning has been proved to outperform canonical Imitation Learning when the demonstrated task is complex and has a subtask structure. In this paper, we propose a novel HIL algorithm by integrating the extended AIRL objectives with a directed information term, and provide a VAE-like framework for updating the target hierarchical policy in an end-to-end fashion. Further, we propose an EM adaption of our algorithm to fit it with unsegmented expert demonstrations to allow more practicability. We also have provided theoretical analysis and ablation study of each key component of our algorithm design for justification, and comparisons on Mujoco control tasks with SOTA HIL and IL baselines to show the superiority of our algorithm.

As for future works, applying H-AIRL to realistic robotic tasks based on human demonstrations can be a solid practical contribution. Integrating H-AIRL with Meta/Multi-task Learning techniques [36], [37], [38] for novel Multi-task HIL algorithms is also an interesting direction.

REFERENCES

[1] N. Brown and T. Sandholm, "Superhuman ai for multiplayer poker," *Science*, vol. 365, no. 6456, pp. 885–890, 2019.

[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. P. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nat.*, vol. 529, no. 7587, pp. 484–489, 2016. [Online]. Available: https://doi.org/10.1038/nature16961

[3] I. Hosu and T. Rebedea, "Playing atari games with deep reinforcement learning and human checkpoint replay," *CoRR*, vol. abs/1607.05077, 2016. [Online]. Available: http://arxiv.org/abs/1607.05077

[4] F. Ebert, C. Finn, S. Dasari, A. Xie, A. Lee, and S. Levine, "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control," *arXiv preprint arXiv:1812.00568*, 2018.

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

[6] R. S. Sutton, D. Precup, and S. P. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, vol. 112, pp. 181–211, 1999.

[7] M. Sharma, A. Sharma, N. Rhinehart, and K. M. Kitani, "Directed-info GAIL: learning hierarchical policies from unsegmented demonstrations using directed information," in *Proceedings of the 7th International Conference on Learning Representations*. OpenReview.net, 2019.

[8] M. Jing, W. Huang, F. Sun, X. Ma, T. Kong, C. Gan, and L. Li, "Adversarial option-aware hierarchical imitation learning," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, 2021, pp. 5097–5106.

[9] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in Neural Information Processing Systems 29*, 2016.

[10] J. Massey *et al.*, "Causality, feedback and directed information," in *Proc. Int. Symp. Inf. Theory Applic.(ISITA-90)*, 1990, pp. 303–305.

[11] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[12] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," *CoRR*, vol. abs/1710.11248, 2017.

[13] R. Jena, C. Liu, and K. Sycara, "Augmenting gail with bc for sample efficient imitation learning," in *Conference on Robot Learning*. PMLR, 2021, pp. 80–90.

[14] P. Wang, D. Liu, J. Chen, H. Li, and C.-Y. Chan, "Decision making for autonomous driving via augmented adversarial inverse reinforcement learning," in *IEEE International Conference on Robotics and Automation*. IEEE, 2021, pp. 1036–1042.

[15] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proceedings of the 2nd International Conference on Learning Representations*, 2014.

[16] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal processing magazine*, vol. 13, no. 6, pp. 47–60, 1996.

[17] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[18] B. D. Argall, S. Chernova, M. M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[19] D. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," *Neural Computation*, vol. 3, no. 1, pp. 88–97, 1991.

[20] A. Y. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *Proceedings of the 7th International Conference on Machine Learning*. Morgan Kaufmann, 2000, pp. 663–670.

[21] S. Ross, G. J. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, vol. 15. JMLR, 2011, pp. 627–635.

[22] H. M. Le, N. Jiang, A. Agarwal, M. Dudík, Y. Yue, and H. D. III, "Hierarchical imitation and reinforcement learning," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80. PMLR, 2018, pp. 2923–2932.

[23] T. Kipf, Y. Li, H. Dai, V. F. Zambaldi, A. Sanchez-Gonzalez, E. Grefenstette, P. Kohli, and P. W. Battaglia, "Compile: Compositional imitation learning and execution," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97. PMLR, 2019, pp. 3418–3428.

[24] C. Daniel, H. van Hoof, J. Peters, and G. Neumann, "Probabilistic inference for determining options in reinforcement learning," *Machine Learning*, vol. 104, no. 2-3, pp. 337–357, 2016.

[25] Z. Zhang and I. C. Paschalidis, "Provable hierarchical imitation learning via EM," in *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, vol. 130. PMLR, 2021, pp. 883–891.

[26] S. Fine, Y. Singer, and N. Tishby, "The hierarchical hidden markov model: Analysis and applications," *Machine learning*, vol. 32, no. 1, pp. 41–62, 1998.

[27] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*. AAAI Press, 2008, pp. 1433–1438.

[28] S. Zhang and S. Whiteson, "DAC: the double actor-critic architecture for learning options," in *Advances in Neural Information Processing Systems 32*, 2019, pp. 2010–2020.

[29] C. Li, D. Song, and D. Tao, "The skill-action architecture: Learning abstract action embeddings for reinforcement learning," in *Submissions of the 9th International Conference on Learning Representations*, 2021.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, 2017, pp. 5998–6008.

[31] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999.

[32] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[33] L. Rabiner and B. Juang, "An introduction to hidden markov models," *ieee assp magazine*, vol. 3, no. 1, pp. 4–16, 1986.

[34] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347

[35] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *CoRR*, 2016. [Online]. Available: http://arxiv.org/abs/1606.01540

[36] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-shot visual imitation learning via meta-learning," in *Proceedings of the 1st Annual Conference on Robot Learning*, vol. 78. PMLR, 2017, pp. 357–368.

[37] M. P. Deisenroth, P. Englert, J. Peters, and D. Fox, "Multi-task policy search for robotics," in *IEEE International Conference on Robotics and Automation*. IEEE, 2014, pp. 3876–3881.

[38] A. Singh, E. Jang, A. Irpan, D. Kappler, M. Dalal, S. Levine, M. Khansari, and C. Finn, "Scalable multi-task imitation learning with autonomous improvement," in *2020 IEEE International Conference on Robotics and Automation*. IEEE, 2020, pp. 2167–2173.

[39] A. R. Kosiorek, S. Sabour, Y. W. Teh, and G. E. Hinton, "Stacked capsule autoencoders," in *Advances in Neural Information Processing Systems 32*, 2019, pp. 15 486–15 496.

[40] D. Galvin, "Three tutorial lectures on entropy and counting," *arXiv preprint arXiv:1406.7872*, 2014.

[41] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio, "A recurrent latent variable model for sequential data," in *Advances in Neural Information Processing Systems 28*, 2015, pp. 2980–2988.

[42] J. L. W. V. Jensen, "Sur les fonctions convexes et les inégalités entre les valeurs moyennes," *Acta mathematica*, vol. 30, no. 1, pp. 175–193, 1906.

[43] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of the 19th International Conference on Computational Statistics*. Springer, 2010, pp. 177–186.

APPENDIX I
IMPLEMENTATION OF THE HIERARCHICAL POLICY IN THE ONE-STEP OPTION MODEL

In this section, we give out the detailed structure design of the hierarchical policy introduced in Section III-B, i.e., $\pi_\theta(Z|S,Z')$ and $\pi_\phi(A|S,Z)$, which is proposed in [29]. This part is not our contribution, so we only provide the details for the purpose of implementation.

As mentioned in Section III-B, the structure design is based on the Multi-Head Attention (MHA) mechanism [30]. An attention function can be described as mapping a query, i.e., $q \in \mathbb{R}^{d_k}$, and a set of key-value pairs, i.e., $K = [k_1 \cdots k_n]^T \in \mathbb{R}^{n\times d_k}$ and $V = [v_1 \cdots v_n]^T \in \mathbb{R}^{n\times d_v}$, to an output. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. To be specific:

$$Attention(q,K,V) = \sum_{i=1}^{n}\left[\frac{exp(q\cdot k_i)}{\sum_{j=1}^{n} exp(q\cdot k_j)} \times v_i\right] \tag{11}$$

where q, K, V are learnable parameters, $\frac{exp(q\cdot k_i)}{\sum_{j=1}^{n} exp(q\cdot k_j)}$ represents the attention weight that the model should pay to item i. In MHA, the query and key-value pairs are first linearly projected h times to get h different queries, keys and values. Then, an attention function is performed on each of these projected versions of queries, keys and values in parallel to get h outputs which are then be concatenated and linearly projected to acquire the final output. The whole process can be represented as Equation (12), where $W_i^q \in \mathbb{R}^{d_k\times d_k}, W_i^K \in \mathbb{R}^{d_k\times d_k}, W_i^V \in \mathbb{R}^{d_v\times d_v}, W^O \in \mathbb{R}^{nd_v\times d_v}$ are the learnable parameters.

$$MHA(q,K,V) = Concat(head_1,\cdots,head_h)W^O,\ head_i = Attention(qW_i^q, KW_i^K, VW_i^V) \tag{12}$$

In this work, the option is represented as an N-dimensional one-hot vector, where N denotes the total number of options to learn. The high-level policy $\pi_\theta(Z|S,Z')$ has the structure shown as:

$$q = linear(Concat[S, W_C^T Z']),\ dense_Z = MHA(q, W_C, W_C),\ Z \sim Categorical(\cdot|dense_Z) \tag{13}$$

$W_C \in \mathbb{R}^{N\times E}$ is the option context matrix of which the i-th row represents the context embedding of the option i. W_C is also used as the key and value matrix for the MHA, so $d_k = d_v = E$ in this case. Note that W_C is only updated in the MHA module. Intuitively, $\pi_\theta(Z|S,Z')$ attends to all the option context embeddings in W_C according to S and Z'. If Z' still fits S, $\pi_\theta(Z|S,Z')$ will assign a larger attention weight to Z' and thus has a tendency to continue with it; otherwise, a new skill with better compatibility will be sampled.

As for the low-level policy $\pi_\phi(A|S,Z)$, it has the following structure:

$$dense_A = MLP(S, W_C^T Z),\ A \sim Categorical/Gaussian(\cdot|dense_A) \tag{14}$$

where MLP represents a multilayer perceptron, A follows a categorical distribution for the discrete case or a gaussian distribution for the continuous case. The context embedding corresponding to Z, i.e., $W_C^T Z$, instead of Z only, is used as input of π_ϕ since it can encode multiple properties of the option Z [39].

APPENDIX II
A LOWER BOUND OF THE DIRECTED INFORMATION OBJECTIVE

In this section, we give out the derivation of a lower bound of the directed information from the trajectory sequence $X_{0:T}$ to the local latent context sequence $Z_{0:T}$, i.e., $I(X_{0:T} \to Z_{0:T})$ as follows:

$$\begin{aligned}
I(X_{0:T}\to Z_{0:T}) &= \sum_{t=1}^{T}[I(X_{0:t}; Z_t|Z_{0:t-1})] \\
&= \sum_{t=1}^{T}[H(Z_t|Z_{0:t-1}) - H(Z_t|X_{0:t}, Z_{0:t-1})] \\
&\geq \sum_{t=1}^{T}[H(Z_t|X_{0:t-1}, Z_{0:t-1}) - H(Z_t|X_{0:t}, Z_{0:t-1})] \\
&= \sum_{t=1}^{T}[H(Z_t|X_{0:t-1}, Z_{0:t-1}) + \sum_{\substack{X_{0:t},\\ Z_{0:t-1}}} P(X_{0:t}, Z_{0:t-1}) \sum_{Z_t} P(Z_t|X_{0:t}, Z_{0:t-1}) logP(Z_t|X_{0:t}, Z_{0:t-1})]
\end{aligned} \tag{15}$$

In Equation (15), $I(Var_1; Var_2|Var_3)$ denotes the conditional mutual information, $H(Var_1|Var_2)$ denotes the conditional entropy, and the inequality holds because of the basic property related to conditional entropy: increasing conditioning cannot

increase entropy [40]. $H(Z_t|X_{0:t-1}, Z_{0:t-1})$ is the entropy of the high-level policy (introduced later), which can serve as a regulator and more convenient to obtain. Further, the second term in the last step can be processed as follows:

$$\begin{aligned}
&\sum_{Z_t} P(Z_t|X_{0:t}, Z_{0:t-1}) log P(Z_t|X_{0:t}, Z_{0:t-1}) \\
&= \sum_{Z_t} P(Z_t|X_{0:t}, Z_{0:t-1}) \left[log \frac{P(Z_t|X_{0:t}, Z_{0:t-1})}{P_\omega(Z_t|X_{0:t}, Z_{0:t-1})} + log P_\omega(Z_t|X_{0:t}, Z_{0:t-1}) \right] \\
&= D_{KL}(P(\cdot|X_{0:t}, Z_{0:t-1}) || P_\omega(\cdot|X_{0:t}, Z_{0:t-1})) + \sum_{Z_t} P(Z_t|X_{0:t}, Z_{0:t-1}) log P_\omega(Z_t|X_{0:t}, Z_{0:t-1}) \\
&\geq \sum_{Z_t} P(Z_t|X_{0:t}, Z_{0:t-1}) log P_\omega(Z_t|X_{0:t}, Z_{0:t-1})
\end{aligned} \tag{16}$$

where $D_{KL}(\cdot)$ denotes the Kullback-Leibler (KL) Divergence which is non-negative [31], $P_\omega(Z_t|X_{0:t}, Z_{0:t-1})$ is a variational estimation of the posterior distribution of Z_t given $X_{0:t}$ and $Z_{0:t-1}$, i.e., $P(Z_t|X_{0:t}, Z_{0:t-1})$, which is modeled as a recurrent neural network with the parameter set ω in our work. Based on Equation (15) and (16), we can obtain a lower bound of $I(X_{0:T} \to Z_{0:T})$ denoted as L^{DI}:

$$L^{DI} = \sum_{t=1}^{T} [\sum_{X_{0:t}, Z_{0:t}} P(X_{0:t}, Z_{0:t}) log P_\omega(Z_t|X_{0:t}, Z_{0:t-1}) + H(Z_t|X_{0:t-1}, Z_{0:t-1})] \tag{17}$$

Note that the joint distribution $P(X_{0:t}, Z_{0:t})$ has a recursive definition as follows:

$$\begin{aligned}
P(X_{0:t}, Z_{0:t}) = P(X_t|X_{0:t-1}, Z_{0:t}) P(Z_t|X_{0:t-1}, Z_{0:t-1}) P(X_{0:t-1}, Z_{0:t-1}) \\
P(X_0, Z_0) = P((S_0, A_{-1}), Z_0) = \mu(S_0)
\end{aligned} \tag{18}$$

where $\mu(S_0)$ denotes the distribution of the initial states. The second line in Equation (18) holds because A_{-1} and Z_0 are dummy variables which are only for keeping the notations consistent and never executed and set to be constant. Based on Equation (18), we can get:

$$\begin{aligned}
&P(X_{0:t}, Z_{0:t}) = \mu(S_0) \prod_{i=1}^{t} P(Z_i|X_{0:i-1}, Z_{0:i-1}) P(X_i|X_{0:i-1}, Z_{0:i}) \\
&= \mu(S_0) \prod_{i=1}^{t} P(Z_i|X_{0:i-1}, Z_{0:i-1}) P((S_i, A_{i-1})|X_{0:i-1}, Z_{0:i}) \\
&= \mu(S_0) \prod_{i=1}^{t} P(Z_i|X_{0:i-1}, Z_{0:i-1}) P(A_{i-1}|X_{0:i-1}, Z_{0:i}) \mathcal{P}(S_i|S_{i-1}, A_{i-1})
\end{aligned} \tag{19}$$

In Equation (19), $\mathcal{P}(S_i|S_{i-1}, A_{i-1})$ is the transition dynamic, $P(Z_i|X_{0:i-1}, Z_{0:i-1})$ is output of the high-level policy which decides on the current option Z_i based on the history trajectory $X_{0:i-1}$ and option choices $Z_{0:i-1}$, $P(A_{i-1}|X_{0:i-1}, Z_{0:i})$ is output of the low-level policy which outputs the action choice based on the history information and current option choice.

To sum up, we can adopt the high-level policy, low-level policy and variational posterior to get an estimation of the lower bound of the directed information objective through Monte Carlo sampling [32] according to Equation (17) and (19).

APPENDIX III
IMPLEMENTATION DETAILS OF THE VARIATIONAL POSTERIORS

The variational posterior for the local latent code, i.e., $P_\omega(Z_t|X_{0:t}, Z_{0:t-1})$, is modeled as $P_\omega(Z_t|X_t, Z_{t-1}, h_{t-1})$, where h_{t-1} is the internal hidden state of an RNN. h_{t-1} is recursively maintained with the time series using the GRU rule, i.e., $h_{t-1} = GRU(X_{t-1}, Z_{t-2}, h_{t-2})$, to embed the history information in the trajectory, i.e., $X_{0:t-1}$ and $Z_{0:t-2}$. Note that the RNN-based posterior has been used and justified in the process for sequential data [41].

APPENDIX IV
DERIVATION OF EQUATION (8) IN THE MLE OBJECTIVE

In Equation (20), Z_0 is a dummy variable which is assigned before the episode begins and never executed. It's implemented as a constant across different episodes, so we have $P(S_0, Z_0) = P(S_0) = \mu(S_0)$, where $\mu(\cdot)$ denotes the initial

state distribution. Also, we have $P(S_{t+1}, Z_{t+1}|S_t, Z_t, Z_{t+1}, A_t) = P(Z_{t+1}|S_t, Z_t, Z_{t+1}, A_t)P(S_{t+1}|S_t, Z_t, Z_{t+1}, A_t) = \mathcal{P}(S_{t+1}|S_t, A_t)$, since the transition dynamic $\mathcal{P}$ is irrelevant to the local latent codes Z based on the definition of MDP.

$$\begin{aligned}
&P_\vartheta(X_{0:T}, Z_{0:T}) \propto \mu(\widetilde{S}_0) \prod_{t=0}^{T-1} \mathcal{P}(\widetilde{S}_{t+1}|\widetilde{S}_t, \widetilde{A}_t) exp(\mathcal{R}_\vartheta(\widetilde{S}_t, \widetilde{A}_t)) \\
&= P(S_0, Z_0) \prod_{t=0}^{T-1} P(S_{t+1}, Z_{t+1}|S_t, Z_t, Z_{t+1}, A_t) exp(\mathcal{R}_\vartheta(S_t, Z_t, Z_{t+1}, A_t) \\
&= \mu(S_0) \prod_{t=0}^{T-1} \mathcal{P}(S_{t+1}|S_t, A_t) exp(\mathcal{R}_\vartheta(S_t, Z_t, Z_{t+1}, A_t))
\end{aligned} \tag{20}$$

APPENDIX V
JUSTIFICATION OF THE OBJECTIVE FUNCTION DESIGN OF H-AIRL IN EQUATION (9)

In this section, we prove that by optimizing the objective functions shown as Equation (9), we can get the solution of the MLE problem shown as Equation (8), i.e., the hierarchical reward function and policy of the expert.

In Appendix A of [12], they show that the discriminator objective is equivalent to the MLE objective where f_ϑ serves as R_ϑ, when $D_{KL}(\pi(\tau)||\pi_E(\tau))$ is minimized. The same conclusion can be acquired by simply replacing $\{S_t, A_t, \tau\}$ with $\{(S_t, Z_t), (Z_{t+1}, A_t), (X_{0:T}, Z_{0:T})\}$, i.e., the extended definition of the state, action and trajectory, in the original proof, which we don't repeat here. Then, we only need to prove that $D_{KL}(\pi_{\theta,\phi}(X_{0:T}, Z_{0:T})||\pi_E(X_{0:T}, Z_{0:T}))$ can be minimized through Equation (9):

$$\begin{aligned}
&\max_{\theta,\phi} \mathbb{E}_{(X_{0:T}, Z_{0:T})\sim\pi_{\theta,\phi}(\cdot)} \sum_{t=0}^{T-1} R_{IL}^t \\
&= \mathop{\mathbb{E}}_{X_{0:T}, Z_{0:T}} \left[\sum_{t=0}^{T-1} log D_\vartheta(S_t, Z_t, Z_{t+1}, A_t) - log(1 - D_\vartheta(S_t, Z_t, Z_{t+1}, A_t)) \right] \\
&= \mathop{\mathbb{E}}_{X_{0:T}, Z_{0:T}} \left[\sum_{t=0}^{T-1} f_\vartheta(S_t, Z_t, Z_{t+1}, A_t) - log \pi_{\theta,\phi}(Z_{t+1}, A_t|S_t, Z_t) \right] \\
&= \mathop{\mathbb{E}}_{X_{0:T}, Z_{0:T}} \left[\sum_{t=0}^{T-1} f_\vartheta(S_t, Z_t, Z_{t+1}, A_t) - log(\pi_\theta(Z_{t+1}|S_t, Z_t)\pi_\phi(A_t|S_t, Z_{t+1})) \right] \\
&= \mathop{\mathbb{E}}_{X_{0:T}, Z_{0:T}} \left[log \frac{\prod_{t=0}^{T-1} exp(f_\vartheta(S_t, Z_t, Z_{t+1}, A_t))}{\prod_{t=0}^{T-1} \pi_\theta(Z_{t+1}|S_t, Z_t)\pi_\phi(A_t|S_t, Z_{t+1})} \right] \\
&\iff \max_{\theta,\phi} \mathop{\mathbb{E}}_{X_{0:T}, Z_{0:T}} \left[log \frac{\prod_{t=0}^{T-1} exp(f_\vartheta(S_t, Z_t, Z_{t+1}, A_t))/Z_\vartheta}{\prod_{t=0}^{T-1} \pi_\theta(Z_{t+1}|S_t, Z_t)\pi_\phi(A_t|S_t, Z_{t+1})} \right]
\end{aligned} \tag{21}$$

Note that $Z_\vartheta = \sum_{X_{0:T}, Z_{0:T}} \widehat{P}_\vartheta(X_{0:T}, Z_{0:T})$ (defined in Equation (8)) is the normalized function parameterized with ϑ, so the introduction of Z_ϑ will not influence the optimization with respect to θ and ϕ and the equivalence at the last step holds. Also, the second equality shows that the hierarchical policy is recovered by optimizing an entropy-regularized policy objective where f_ϑ serves as R_ϑ. Further, we have:

$$\begin{aligned}
&\max_{\theta,\phi} \mathop{\mathbb{E}}_{X_{0:T}, Z_{0:T}} \left[log \frac{\prod_{t=0}^{T-1} exp(f_\vartheta(S_t, Z_t, Z_{t+1}, A_t))/Z_\vartheta}{\prod_{t=0}^{T-1} \pi_\theta(Z_{t+1}|S_t, Z_t)\pi_\phi(A_t|S_t, Z_{t+1})} \right] \\
&= \mathop{\mathbb{E}}_{X_{0:T}, Z_{0:T}} \left[log \frac{\mu(S_0) \prod_{t=0}^{T-1} \mathcal{P}(S_{t+1}|S_t, A_t) \prod_{t=0}^{T-1} exp(f_\vartheta(S_t, Z_t, Z_{t+1}, A_t))/Z_\vartheta}{\mu(S_0) \prod_{t=0}^{T-1} \mathcal{P}(S_{t+1}|S_t, A_t) \prod_{t=0}^{T-1} \pi_\theta(Z_{t+1}|S_t, Z_t)\pi_\phi(A_t|S_t, Z_{t+1})} \right] \\
&= \mathbb{E}_{(X_{0:T}, Z_{0:T})\sim\pi_{\theta,\phi}(\cdot)} \left[log \frac{\pi_E(X_{0:T}, Z_{0:T})}{\pi_{\theta,\phi}(X_{0:T}, Z_{0:T})} \right] \\
&= -D_{KL}(\pi_{\theta,\phi}(X_{0:T}, Z_{0:T})||\pi_E(X_{0:T}, Z_{0:T})) \\
&\iff \min_{\theta,\phi} D_{KL}(\pi_{\theta,\phi}(X_{0:T}, Z_{0:T})||\pi_E(X_{0:T}, Z_{0:T}))
\end{aligned} \tag{22}$$

where the second equality holds because of the definition of π_E (Equation (8) with f_ϑ serving as R_ϑ) and $\pi_{\theta,\phi}$ (Equation (7)).

Algorithm 1 Hierarchical Adversarial Inverse Reinforcement Learning (H-AIRL)

Input: Expert demonstrations $\{X^E_{0:T}\}$ (If the option annotations, i.e., $\{Z^E_{0:T}\}$, are provided, step 5 is not required.)
Initialize the posterior network P_ω, high-level policy π_θ, low-level policy π_ϕ, and discriminator D_ϑ
for *each training episode* **do**
    Generate M trajectories $\{(X_{0:T}, Z_{0:T})\}$ with π_θ and π_ϕ by interacting with the simulator
    Sample local latent codes corresponding to the expert trajectories using the posterior, i.e., $Z^E_{0:T} \sim P_\omega(\cdot|X^E_{0:T})$
    Update P_ω by minimizing L^{DI} (Equation (5)) using Stochastic Gradient Descent [43] with $\{(X_{0:T}, Z_{0:T})\}$
    Update D_ϑ by minimizing the cross entropy loss in Equation (9) based on $\{(X_{0:T}, Z_{0:T})\}$ and $\{(X^E_{0:T}, Z^E_{0:T})\}$
    Train π_θ and π_ϕ by maximizing the return function L defined in Equation (10) using the HRL algorithm SA [29]
end for

APPENDIX VI
JUSTIFICATION OF THE EM-STYLE ADAPTION

Given only a dataset of expert trajectories, i.e., $D_E \triangleq \{X_{0:T}\}$, we can still maximize the likelihood estimation $\mathbb{E}_{X_{0:T}\sim D_E}[logP_\vartheta(X_{0:T})]$ through an EM-style adaption:

$$
\begin{aligned}
&\mathbb{E}_{X_{0:T}\sim D_E}[logP_\vartheta(X_{0:T})] = \mathbb{E}_{X_{0:T}\sim D_E}\left[log\left[\sum_{Z_{0:T}} P_\vartheta(X_{0:T}, Z_{0:T})\right]\right] \\
&= \mathbb{E}_{X_{0:T}\sim D_E}\left[log\left[\sum_{Z_{0:T}} \frac{P_\vartheta(X_{0:T}, Z_{0:T})}{P_{\overline{\vartheta}}(Z_{0:T}|X_{0:T})} P_{\overline{\vartheta}}(Z_{0:T}|X_{0:T})\right]\right] \\
&= \mathbb{E}_{X_{0:T}\sim D_E}\left[log\left[\mathbb{E}_{Z_{0:T}\sim P_{\overline{\vartheta}}(\cdot|X_{0:T})} \frac{P_\vartheta(X_{0:T}, Z_{0:T})}{P_{\overline{\vartheta}}(Z_{0:T}|X_{0:T})}\right]\right] \\
&\geq \mathbb{E}_{X_{0:T}\sim D_E}\left[\mathbb{E}_{Z_{0:T}\sim P_{\overline{\vartheta}}(\cdot|X_{0:T})} log\frac{P_\vartheta(X_{0:T}, Z_{0:T})}{P_{\overline{\vartheta}}(Z_{0:T}|X_{0:T})}\right] \\
&= \mathbb{E}_{X_{0:T}\sim D_E, Z_{0:T}\sim P_{\overline{\omega}}(\cdot|X_{0:T})}\left[log\frac{P_\vartheta(X_{0:T}, Z_{0:T})}{P_{\overline{\vartheta}}(Z_{0:T}|X_{0:T})}\right] \\
&= \mathbb{E}_{X_{0:T}, Z_{0:T}}[logP_\vartheta(X_{0:T}, Z_{0:T})] - \mathbb{E}_{X_{0:T}, Z_{0:T}}\left[logP_{\overline{\vartheta}}(Z_{0:T}|X_{0:T})\right]
\end{aligned}
\tag{23}
$$

where we adopt the Jensen's inequality [42] in the 4-th step. Also, we note that $P_{\overline{\omega}}(Z_{0:T}|X_{0_T})$ provides a posterior distribution of $Z_{0:T}$, which corresponds to the generating process led by the hierarchical policy. As justified in V, the hierarchical policy is trained with the reward function parameterized with $\overline{\vartheta}$. Thus, the hierarchical policy is a function of $\overline{\vartheta}$, and the network $P_{\overline{\omega}}$ corresponding to the hierarchical policy provides a posterior distribution related to the parameter set $\overline{\vartheta}$, i.e., $Z_{0:T} \sim P_{\overline{\vartheta}}(\cdot|X_{0:T}) \iff Z_{0:T} \sim P_{\overline{\omega}}(\cdot|X_{0:T})$, due to which the 5-th step holds. Note that $\overline{\vartheta}, \overline{\omega}$ denote the parameters ϑ, ω before being updated in the M step.

In the second equality of Equation (23), we introduce the sampled local latent codes in the E step as discussed in Section IV-B. Then, in the M step, we optimize the objectives shown in Equation (9) for iterations, by replacing the samples in the first term of Equation (9) with $(X_{0:T}, Z_{0:T})$ collected in the E step. This is equivalent to solve the MLE problem: $max_\vartheta\, \mathbb{E}_{X_{0:T}\sim D_E, Z_{0:T}\sim P_{\overline{\omega}}(\cdot|X_{0:T})}[logP_\vartheta(X_{0:T}, Z_{0:T})]$, through which we can maximize a lower bound of the original objective, i.e., $max_\vartheta\, \mathbb{E}_{X_{0:T}\sim D_E}[logP_\vartheta(X_{0:T})]$, as shown in the last step of Equation (23). Thus, the original objective can be optimized through this EM procedure. Note that the second term in the last step is a function of the old parameter $\overline{\vartheta}$ so that it can be overlooked when optimizing with respect to ϑ.

APPENDIX VII
PSEUDO CODE OF THE OVERALL ALGORITHM

The pseudo code of H-AIRL is provided as Algorithm 1.

APPENDIX VIII
IMPLEMENTATION DETAILS OF H-GAIL

H-GAIL is a variant of our algorithm by replacing the AIRL component with GAIL. Similar with Section IV-B, we need to provide an extension of GAIL with the one-step option model, in order to learn a hierarchical policy. The extension method follows Option-GAIL [8] which is one of our baselines. H-GAIL also uses an adversarial learning framework that

contains a discriminator D_ψ and a hierarchical policy $\pi_{\theta,\phi}$, for which the objective are as follows:

$$\begin{aligned} &\max_{\psi} \mathbb{E}_{(S,A,Z,Z')\sim\pi_E(\cdot)}\left[log(1-D_\psi(S,A,Z,Z'))\right] + \mathbb{E}_{(S,A,Z,Z')\sim\pi_{\theta,\phi}(\cdot)}\left[logD_\psi(S,A,Z,Z')\right] \\ &\max_{\theta,\phi} L^{IL} = \mathbb{E}_{(X_{0:T},Z_{0:T})\sim\pi_{\theta,\phi}(\cdot)} \sum_{t=0}^{T-1}\left[-logD_\psi(S_t,A_t,Z_{t+1},Z_t)\right] \end{aligned} \tag{24}$$

where (S, A, Z, Z') denotes $(S_t, A_t, Z_{t+1}, Z_t),\ t = \{0, \cdots, T-1\}$. It can be observed that the discriminator D_ψ is trained as a classifier to distinguish the expert demonstrations (labeled as 0) and generated samples (labeled as 1), and does not have a specially-designed structure like the discriminator D_ϑ in H-AIRL (Equation (9)) so that it cannot recover the expert reward function. Note that, for H-GAIL, we only change the definition of L^{IL} in the overall objective, i.e., Equation (10), with the one in Equation (24), but keep the directed information objective L^{DI} which is not included in the objective of Option-GAIL.