GO-DICE: Goal-Conditioned Option-Aware Offline Imitation Learning via Stationary Distribution Correction Estimation

Abhinav Jain, Vaibhav Unhelkar

Rice University, Houston, TX
abhinav.jain@rice.edu, vaibhav.unhelkar@rice.edu

Abstract

Offline imitation learning (IL) refers to learning expert behavior solely from demonstrations, without any additional interaction with the environment. Despite significant advances in offline IL, existing techniques find it challenging to learn policies for long-horizon tasks and require significant re-training when task specifications change. Towards addressing these limitations, we present GO-DICE an offline IL technique for goal-conditioned long-horizon sequential tasks. GO-DICE discerns a hierarchy of sub-tasks from demonstrations and uses these to learn separate policies for sub-task transitions and action execution, respectively; this hierarchical policy learning facilitates long-horizon reasoning. Inspired by the expansive DICE-family of techniques, policy learning at both the levels transpires within the space of stationary distributions. Further, both policies are learnt with goal conditioning to minimize need for retraining when task goals change. Experimental results substantiate that GO-DICE outperforms recent baselines, as evidenced by a marked improvement in the completion rate of increasingly challenging pick-and-place Mujoco robotic tasks. GO-DICE is also capable of leveraging imperfect demonstration and partial task segmentation when available, both of which boost task performance relative to learning from expert demonstrations alone.

Introduction

Learning to make decisions and accomplish sequential tasks is a core problem in artificial intelligence (AI). Reinforcement learning (RL) addresses this problem by enabling agents to learn task policies by interacting with their environment. However, in many practical scenarios, exploratory interaction with the environment is expensive, unsafe, or even infeasible. For these scenarios, offline imitation learning (IL) offers AI agents an approach to learn task policies without any environmental interaction when task demonstrations provided by other (typically expert) agents are available. Also referred to as learning from demonstration, classic techniques for offline IL include behavioral cloning (BC) and inverse reinforcement learning (IRL) (Pomerleau 1991; Abbeel and Ng 2004). These techniques and their extensions have been used to address complex problems in robotics, human modeling, and beyond (Unhelkar, Li, and Shah 2020; Ravichandar et al. 2020; Wu et al. 2021).

Originating almost three decades ago, offline IL remains an active area of research with recent techniques aiming to solve increasingly challenging sequential tasks (Osa et al. 2018; Arora and Doshi 2021; Seo and Unhelkar 2022).[1] For instance, to enhance scalability and address tasks with high-dimensional state spaces, more recent IL techniques leverage advances in deep learning and generative adversarial training to represent and learn policies (Ho and Ermon 2016). Similarly, to address poor generalizability of classical techniques to the states absent in expert demonstrations (Ross, Gordon, and Bagnell 2011), recent IL techniques have employed imperfect demonstrations to obtain a more comprehensive coverage of the state space in the training data (Kim et al. 2021, 2022; Ma et al. 2022b). Despite these advancements, existing offline IL techniques find it challenging to learn policies for long-horizon tasks and require significant retraining when task specifications change.

Towards addressing these limitations, we introduce a goal-conditioned option-aware approach to offline IL: GO-DICE[2]. Inspired by the options framework (Sutton, Precup, and Singh 1999), GO-DICE segments available demonstrations into a sequence of sub-tasks to facilitate long-horizon reasoning. It then uses the segmentation results to learn a hierarchy of policies for transitioning between sub-tasks and action execution within a sub-task. The task segmentation and hierarchical policy learning repeats iteratively until convergence. In GO-DICE, policies at both the levels of hierarchy depend on not only state but also goals. Similar to recent RL techniques such as HER (Andrychowicz et al. 2017; Fang et al. 2018), this goal conditioning seeks to minimize the need for retraining when task goals change.

Given the task segmentations and choice of policy representation, a policy learning subroutine is necessary to learn each level of hierarchical policy. GO-DICE leverages a technique called DemoDICE for this purpose, owing to its ability to learn from imperfect demonstrations and without adversarial training (Kim et al. 2021). As a member of the *station-*

[1] Closely related is the paradigm of online imitation learning that combines demonstration data with agent's experience. However, like reinforcement learning, it is not suitable when interacting with environment is unsafe, expensive or infeasible.

[2] An extended version of this paper, which includes supplementary material (appendices and video supplements) mentioned in the text, is available at https://arxiv.org/abs/2312.10802

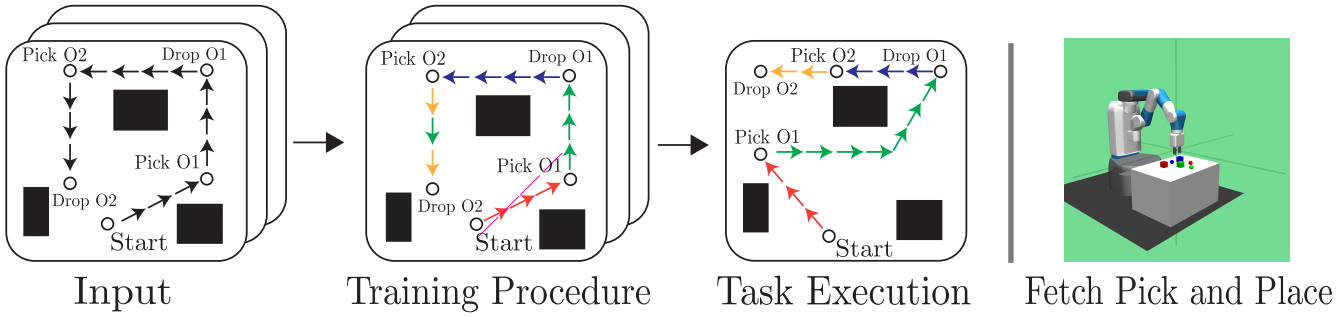


Figure 1: (Left) Schematic illustration of GO-DICE on a 2-dimensional 2-object pick and place task. Given expert and imperfect demonstrations, iteratively, GO-DICE segments them into sub-tasks (shown as colored arrows in the training procedure) and uses these segments to learn a goal-conditioned hierarchical policy. The task segmentation may be imperfect and improves as the policy estimate improves. The learned policy can be used to execute tasks, even when the underlying goals (e.g., pick and place locations) change. (Right) A snapshot of the high-dimensional Fetch Pick-and-Place environment used in experiments.

ary DIstribution Correction Estimation (DICE) family of approaches, GO-DICE too learns within the space of stationary distributions. However, in contrast to prior DICE-based techniques, GO-DICE learns option-aware goal-conditioned policies and (when available) is able to utilize annotations of task segments to boost its learning. In summary, we make three key contributions in the paper.

- First, we provide an algorithm called GO-DICE for offline imitation learning from (expert and imperfect) demonstrations and (when available) partial annotations of task segments. Fig. 1 provides a schematic illustration.
- Second, through numerical experiments, we show that GO-DICE outperforms recent baselines in challenging long-horizon Mujoco tasks (shown in Fig. 1 right) and does not require retraining when task goals change.
- Third, through an ablation study, we show that GO-DICE can successfully utilize partial annotations of task segments when available. In tasks where humans can provide these annotations, this feature of our approach can be used to boost learning performance.

Related Work

Before describing GO-DICE, we briefly highlight the concepts and IL techniques that have informed our work. A comparative overview is provided in Table 1.

IL via stationary *DI*stribution *C*orrection *E*stimation. The problem of IL has been mapped to a variety of optimization formulations. For instance, a fruitful line of research has been the use of generative adversarial optimization, resulting in techniques such as GAIL and its extensions (Ho and Ermon 2016). However, these techniques tend to require a large amount of training data, making them appropriate in settings where the learner can interact with its environment to collect this data. For offline IL, the focus of this work, the DICE family of algorithms has recently gained momentum, which involve estimating the corrections between the stationary distribution of the optimal policy and that of the provided dataset (Kostrikov, Nachum, and Tompson 2019). For instance, DemoDICE and LobsDICE optimize KL-divergence regularized state-action and state-transition stationary distribution matching objectives, respectively (Kim et al. 2021, 2022). SMODICE optimizes a more general f-divergence regularized state-occupancy only matching objective (Ma et al. 2022b). In this work, we build on these DICE-family of techniques, owing to their state-of-the-art performance in offline IL and (relative to generative adversarial training) stable training process.

	Policy Representation		Can Learn Using	
	Goal-conditioned	Option-aware	Imperfect Demos.	No Interaction
GAIL				
GoalGAIL	✓			
OptionGAIL		✓		
DemoDICE			✓	✓
GoFAR	✓		✓	✓
GO-DICE	✓	✓	✓	✓

Table 1: An overview of related IL techniques

Auxiliary Inputs in IL. Classically, IL aims to learn policies to solve Markov decision processes (MDPs) given *expert* demonstrations (Puterman 2014; Osa et al. 2018). Mathematically, policies are represented as a mapping from states to actions. Demonstrations correspond to (state, action)-tuples. More recently, guided by both computational and human-centered aspects, this classical paradigm has been extended to consider alternate policy representations and training data that involve auxiliary inputs. For instance, imperfect demonstrations have been introduced as additional inputs to facilitate generalization (Wu et al. 2019; Wang et al. 2021; Kim et al. 2021). Recognizing that human teachers can provide auxiliary inputs (such as corrections) easily in certain tasks, human-guided IL approaches have been developed (Chernova and Thomaz 2014; Unhelkar and Shah 2019; Quintero-Pena et al. 2022; Habibian, Jonnavittula, and Losey 2022). Our work is informed by such techniques and

considers four types of auxiliary inputs – goals, options, task segments, and imperfect demonstrations – to facilitate generalization and long-horizon reasoning.

Goal-conditioned IL. Goal-conditioned policy representations enable learning of a unified policy for a family of tasks, which are identical in all respects but have distinct goals. Goal-conditioned IL techniques have been developed using both GAIL and DICE frameworks. GoalGAIL performs goal-conditioned occupancy measurement matching to learn a unified policy for a variety of task goals (Ding et al. 2019); similar to GAIL, this technique requires the learner to interact with its environment. Among the DICE-family, the algorithm GoFAR performs goal-conditioned offline IL (Ma et al. 2022a). In contast, the proposed GO-DICE considers not only goals but also sub-tasks in its policy representation.

Option-aware IL. Long-horizon tasks can be viewed as composed of a sequence of sub-tasks that need to be executed in a particular order (Byrne and Russon 1998). This modeling insight has been formalized using the options framework (Sutton, Precup, and Singh 1999; Daniel et al. 2016) and offers an effective strategy to learning tasks by breaking them down into manageable sub-tasks, learning to accomplish them individually, and subsequently combining them to achieve the overarching task objective. While initially proposed for RL, this hierarchical approach has also been utilized for IL (Ranchod, Rosman, and Konidaris 2015; Le et al. 2018; Unhelkar and Shah 2019; Gupta et al. 2019; Jing et al. 2021; Orlov-Savko et al. 2022; Jamgochian et al. 2023; Nasiriany et al. 2023; Gao, Jiang, and Chen 2023; Chen, Lan, and Aggarwal 2023). Unsupervised methodologies, such as InfoGAIL and Directed InfoGAIL, leverage information-theoretic measures to initially discover latent options and subsequently imitate the expert (Li, Song, and Ermon 2017; Sharma et al. 2018). OptionGAIL, a variant of note uncovers options concurrently through an Expectation-Maximization procedure (Jing et al. 2021).

Although this line of work has proven successful in settings where the learner can interact with the environment and has a fixed set of task goals, its efficacy in entirely offline IL scenarios with dynamically varying goals remains an open question that we explore in this work. To our knowledge, GO-DICE is the first approach to offline imitation learning that considers both goal-conditioned and option-aware policies. Further, informed by practical considerations, GO-DICE includes mechanisms to boost learning from auxiliary information (namely, annotations of task segments and imperfect demonstrations) when available.

Problem Statement

Task Model. We model the tasks of interest as goal-conditioned MDPs (Schaul et al. 2015). Formally, the tasks are specified via the tuple $\big(\mathcal{S}, \mathcal{G}, \mathcal{A}, \mathbf{R}, \mathbf{T}, \mu_s, \mu_g, \gamma, T\big)$, where $(\mathcal{S}, \mathcal{A}, \mathcal{G})$ are the set of task states, actions, and goals; $\mathbf{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \mapsto \mathbb{R}$ is the goal-conditioned reward; $\mathbf{T} : \mathcal{S} \times \mathcal{A} \mapsto \Delta(\mathcal{S})$ represents the transition function; $\mu_s(s)$ represents the initial state distribution; μ_g is the set of goals; T task-horizon and $\gamma \in (0, 1]$ is the discount factor. Typically, goal-conditioned MDPs utilize a sparse reward function, which is 1 when goal is achieved and 0 otherwise.

Expert Policy. Informed by the options framework, we assume that the expert solves this goal-conditioned task by completing a set of sub-tasks (or, equivalently, options). Mathematically, in our model, the expert maintains

- a set of K discrete options, $c \in \mathcal{C} = \{1, \ldots, K\}$,
- initial option distribution, $c_0 \sim \mu_c(.|s, g)$,
- a high-level goal-conditioned policy for selecting the next option (or sub-task), $\pi_H(c|s, c', g)$, and
- a set of K low-level goal-conditioned policies for executing the chosen sub-task, $\pi_L(a|s, c, g)$.

We denote $\pi_E \equiv \{\pi_H, \pi_L\}$ to represent the expert's policy.

Inputs. The problem of offline IL assumes as inputs the task model without the reward function $\big(\mathcal{S}, \mathcal{G}, \mathcal{A}, \mathbf{T}, \mu_s, \gamma\big)$ and a set of expert demonstrations, $\mathcal{D}_E$. Each demonstration represents an execution trace, $\tau \in \mathcal{D}_E = \{s_{0:T}, a_{0:T-1}\}$ generated by the expert using π_E. Besides expert demonstrations, we consider the following auxiliary inputs: D_I, a set of imperfect demonstrations collected with unknown degree of optimality; g, task goal for each demonstration $\tau \in D_I \cup D_E$; (optionally) the number of options K; and (optionally) partial annotations of task segments.[3] Mathematically, partial annotations of task segments correspond to option labels $(c_{0:T-1})$ for the expert demonstrations $\tau \in \mathcal{D}_E$.

We refer to the problem as "semi-supervised" when the optional inputs are available. The use of optional inputs is informed by prior human-guided IL techniques that leverage auxiliary inputs which can be readily queried from a human expert (Chernova and Thomaz 2014; Unhelkar and Shah 2019). In the option-aware setting, we observe that sub-tasks change less frequently over the course of a long-horizon demonstration; i.e., the number of change-points of options is significantly lower than the demonstration length. Thus, in domains where a human expert can label change-points of sub-tasks, the optional input can be obtained with low annotation effort and help boost learning performance. As such, semi-supervised techniques that can leverage this auxiliary information when available are desirable.

Desired Output. Given the problem inputs, we consider the problem of learning an estimate of the expert policy π_E without any interaction with the environment.

GO-DICE: Goal-conditioned Option-aware Stationary DIstribution Correction Estimation

To solve this problem, we introduce the algorithm *Goal-Conditioned Option-aware stationary DIstribution Correction Estimation* (GO-DICE). GO-DICE extends the DICE family of offline IL techniques by incorporating goal conditioning and options. To derive GO-DICE, we begin with the

[3] A *task segment* is defined as a continuous sequence of state-action tuples associated with the same option. Specifically, a continuous sequence $(s_{t_1:t_2}, a_{t_1:t_2})$ is referred to as a task segment if (a) for all t in the interval $[t_1, t_2]$, c_t remains constant at some option $c \in \mathcal{C}$ and (b) c_{t_1-1} and c_{t_2+1} are distinct from c.

formulation of its underlying optimization problem and then describe the computational approach to solve it.

Optimization Problem

Inspired by DICE family of techniques (Kostrikov, Nachum, and Tompson 2019), GO-DICE utilizes distribution matching to estimate the expert policy. Briefly, DICE techniques seek to align the stationary distribution induced by the learned policy $d^{\pi}(\cdot)$ with that induced by the expert $d^{\pi_E}(\cdot)$. Mathematically, this is achieved by minimizing the KL divergence between the two distributions. As detailed in Related Work, a variety of stationary distributions have been previously considered, such as state occupancy $d^{\pi}(s)$, state-action occupancy $d^{\pi}(s,a)$, and state transitions $d^{\pi}(s,s')$. We consider a stationary distribution that is goal-conditioned and option-aware, $d^{\pi}(c_{-1},s,a,c;g)$

$$= (1-\gamma)\sum_{t=0}^{\infty}\gamma^t p\Big(c_{t-1}=c_{-1}, s_t=s, a_t=a, c_t=c; g\Big).$$

Three stationary distributions are of interest: namely, d^{π_E}, that induced by the expert policy; d^{π_O} that induced by the data set consisting of expert and imperfect demonstrations $\mathcal{D}_O = \mathcal{D}_E \cup \mathcal{D}_I$; and that induced by the learned policy d^{π}.

Primal Optimization Problem. Given the stationary distributions, we can now define the optimization problem for GO-DICE. Informed by work of (Kim et al. 2021), which also considers DICE-based learning from imperfect demonstrations albeit without goals or options, GO-DICE seeks to solve the following constrained optimization problem

$$\min_{d^{\pi}} D_{KL}(d^{\pi}||d^{\pi_E}) + \alpha D_{KL}(d^{\pi}||d^{\pi_O}) \tag{1}$$

subject to Bellman constraints: $\sum_{c,a} d^{\pi}(c',s,c,a;g) = (1-\gamma)\mu(c',s;g) + \gamma\sum_{c'',s',a'}\mathbf{T}(s|s',a')d^{\pi}(c'',s',c',a';g)$ and $d^{\pi}(c',s,c,a;g) \geq 0 \;\forall\; c,c' \in \mathcal{C}, s \in \mathcal{S}, a \in \mathcal{A}, g \in \mathcal{G}$. Intuitively, the optimization function seeks to minimize the difference between stationary distributions induced by the expert and learnt policy. The imperfect demonstrations are used as a regularizer through the term $\alpha D_{KL}(d^{\pi}||d^{\pi_O})$, where α is the regularization coefficient. This distribution regularization has been shown to facilitate offline IL by penalizing distribution drift, without the need of any on-policy sampling (Nachum et al. 2019; Lee et al. 2021; Kim et al. 2021, 2022; Ma et al. 2022a,b). The Bellman constraints ensure that d^{π} is a valid stationary distribution induced by an agent following the hierarchical policy π.

Primal to Dual Conversion. For tractable optimization, we next convert the constrained optimization problem of Eq. 1 into its dual unconstrained formulation. Using the method of Lagrange multipliers, we obtain:

$$\max_{d^{\pi}\geq 0}\min_{\nu} f(\nu,d) - D_{KL}(d^{\pi}||d^{\pi_E}) - \alpha D_{KL}(d^{\pi}||d^{\pi_O}) \tag{2}$$

$$\text{where } f(\nu,d^{\pi}) = \sum_{c',s,g}\nu(c',s,g)\Big((1-\gamma)\mu(c',s;g) + \gamma(T_* d^{\pi})(c',s,g) - \sum_{c,a} d^{\pi}(c',s,c,a;g)\Big),$$

$\nu(c',s,g)$ are the Lagrangian multipliers; and T_* is the transposed Bellman operator such that

$$(T_* d)(c',s;g) = \sum_{c'',s',a'}\mathbf{T}(s|s',a')d(c'',s',c',a';g). \tag{3}$$

The dual objective can be further simplified by introducing the notation of stationary distribution ratio

$$w(c',s,c,a,g) = \frac{d^{\pi}(c',s,c,a;g)}{d^{\pi_O}(c',s,c,a;g)}. \tag{4}$$

Using stationary distribution ratio w, Eq. 2 translates to the following maximin optimization problem

$$\max_{w\geq 0}\min_{\nu}(1-\gamma)\mathbb{E}_{\mu}[\nu(c',s;g)] + \mathbb{E}_{d^{\pi_O}}\Big[w(c',s,c,a,g)\Big(A_{\nu}(c',s,c,a,g) - (1+\alpha)\log w(c',s,c,a,g)\Big)\Big] \tag{5}$$

where, A_{ν} resembles the advantage function and is given as

$$A_{\nu} = r(c',s,c,a,g) + \gamma(T\nu)(s,c,a,g) - \nu(c',s,g) \tag{6}$$

$$r = \log\frac{d^{\pi_E}(c',s,c,a;g)}{d^{\pi_O}(c',s,c,a;g)} \tag{7}$$

$$(T\nu)(s,c,a;g) = \sum_{s'}\mathbf{T}(s'|s,a)\nu(c,s',g). \tag{8}$$

Conversion to Direct Convex Optimization. While the dual formulation avoids the need of constrained optimization, it still requires solution of a challenging maximin optimization problem. Similar to (Kim et al. 2021), we seek to enhance the stability of the optimization process by reducing it to a direct convex optimization problem. A detailed derivation of this conversion is provided in the Appendix, which results in the following direct convex optimization problem:

$$\min_{\nu}(1-\gamma)\mathbb{E}_{\mu}[\nu(c',s,g)] + (1+\alpha)\mathbb{E}_{d^{\pi_O}}[w^*(\cdot)] \tag{9}$$

where, w^* denotes the optimal importance weights given as:

$$w^*(c',s,c,a;g) = \frac{d^{\pi^*}(c',s,c,a;g)}{d^{\pi_O}(c',s,c,a,g)} = \exp\left(\frac{A_{\nu}}{1+\alpha} - 1\right) \tag{10}$$

This problem can be further stabilized via the following surrogate objective, which shares the same optimal value with Eq. 9 but is less prone to exploding gradients:

$$\min_{\nu}(1-\gamma)\mathbb{E}_{\mu}[\nu(\cdot)] + (1+\alpha)\log\mathbb{E}_{d^{\pi_O}}\left[\exp\left(\frac{A_{\nu}}{1+\alpha}\right)\right] \tag{11}$$

Learning Algorithm

Having defined the optimization problem, we now describe the computational approach to solve it. Algorithm 1 provides an overview of the approach, which trains four classes of neural networks: π, π', Ψ, ν to estimate the expert policy. Given the problem inputs, the algorithm first segments all available demonstrations into sub-tasks through a Viterbi-style subroutine (lines 4-7). Using the results of the task segmentation and demonstrations, the algorithm then updates the discriminator network, Lagrange multipliers, and the policy networks via gradient descent (lines 8-10). This process repeats iteratively until convergence. In the remainder of this section, we detail each subroutine of the algorithm along with the associated hyperparameters.

Algorithm 1: GO-DICE

Require: Expert trajectories $\mathcal{D}_E$, imperfect trajectories $\mathcal{D}_I$, task goals for each trajectory, and (optional) partial annotations of task segments
Ensure: Learned policy $\pi = \{\pi_H, \pi_L\}$
1: **Parameters:** α, λ, K, M, N
2: **Initialize:** π, π_t, Ψ, ν
3: **for** $n = 1$ to N **do**
4: **if** every M iterations **then**
5: $\pi' \leftarrow \lambda\pi' + (1-\lambda)\pi$ ▷ Update target networks
6: Segment unannotated trajectories using Eq. 15
7: **end if**
8: Update discriminator network Ψ using Eq. 12
9: Update Lagrange multipliers ν using Eq. 11
10: Update policy networks π using Eq. 13
11: **end for**

Estimating the Lagrange Multipliers. We represent the Lagrange multiplier $v(c', s, g)$ as a neural network. The network parameters are updated by solving Eq. 11 via gradient descent. Computing the associated gradients requires estimates of options (c', c) and the advantage function A_v. We defer the discussion of option inference to end of this section. To compute the advantage function, we first train a discriminator network $\Psi(c', s, c, a; g) : \mathcal{C} \times \mathcal{S} \times \mathcal{C} \times \mathcal{A} \times \mathcal{G} \mapsto (0, 1)$ with the following objective:

$$\min \quad \mathbb{E}_{d^{\pi_E}}[\log \Psi(\cdot)] + \mathbb{E}_{d^{\pi_O}}[\log(1 - \Psi(\cdot))] \tag{12}$$

The optimal discriminator corresponds to $\Psi^* = \frac{d^{\pi_O}}{d^{\pi_O} + d^{\pi_E}}$, and thus can be used to estimate the log-distribution ratio defined in Eq. 7 as $r = -\log\left(\frac{1}{\Psi^*} - 1\right)$. Given r, we can compute the advantage function and required gradient using Eq. 6 and Eq. 11, respectively.

Weighted Policy Learning. To update the policy networks, $\pi = \{\pi_H, \pi_L\}$, GO-DICE performs weighted-behavior cloning using the following objective

$$\begin{aligned} &\max \mathbb{E}_{d^{\pi^*}}\big[\log \pi(c, a|s, c', g)\big] \\ &= \max \mathbb{E}_{d^{\pi_O}}\Big[w^*(\cdot)\Big(\log \pi_H(\cdot) + \log \pi_L(\cdot)\Big)\Big] \end{aligned} \tag{13}$$

where w^* denotes the optimal importance weights of Eq. 10.

Sub-task Inference. Subroutines for learning the various networks rely on not only demonstrations but also the labels of sub-tasks for these demonstrations. Following (Jing et al. 2021), we utilize Viterbi-style decoding to infer these labels

$$\hat{c}_{0:T-1} = \arg\max p(c_{0:T-1}|s_{0:T}, a_{0:T-1}, g) \tag{14}$$

for the unannotated trajectories. To solve this decoding problem, we define forward messages $\alpha_t(c_t)$ and utilize the following recursive formulation to compute them:

$$\begin{aligned} \alpha_t(c_t) &= \max_{c_{0:t-1}} \log p(c_t, a_{0:t}|s_{0:t}, c_{0:t-1}, g) \\ &= \max_{c_{t-1}} \alpha_{t-1}(c_{t-1}) + \log \pi'_H(c_t|\cdot) + \log \pi'_L(a_t|\cdot) \end{aligned} \tag{15}$$

where, $\alpha_0(c_0) = \log \mu'_c(c_0|s_0, g) + \log \pi'_L(a_0|s_0, c_0, g)$. Once all messages are computed, we employ a back-tracing method to decode the sequence of sub-tasks by maximizing $\alpha_t(c_t)$, starting from $\hat{c}_{T-1} = \arg\max_c \alpha_{T-1}(c)$.

Note that the decoding procedure requires an estimate of the expert policy. Instead of using the latest policy estimate π, GO-DICE utilizes target policy networks π' in the option decoding procedure. The target networks are synchronized every M iterations with the main networks π using λ-weighted Polyak averaging, where M and λ are hyperparameters (lines 4-5, Algorithm 1). Unlike other DICE techniques, the optimization routines of discriminator, Lagrange multipliers, and the policy in GO-DICE are intertwined as they all depend on the decoded option labels. Due to this iterative nature of GO-DICE, we found target networks to be critical for enhancing the stability and convergence of the underlying optimization process.

Experiments

We evaluate GO-DICE on robotic manipulation tasks simulated using Mujoco (Todorov, Erez, and Tassa 2012).[4] The tasks are modeled after the benchmark task Fetch Pick and Place (`PnP`), which require a robot to pick an object and place them in desired goal location (Plappert et al. 2018). We consider three variants of this benchmark task – `PnP×1`, `PnP×2`, `PnP×3` – which include $1, 2$, and 3 objects, respectively. The task complexity increases with the number of objects, requiring increasing levels of long-horizon reasoning. Further adding to the complexity, desired goals (place locations of objects) changes across demonstrations. These tasks are chosen for evaluation due to the fact that even the simplest variant `PnP×1` is challenging for offline IL algorithms (Ding et al. 2019). Moreover, `PnP` family of tasks naturally encapsulate other primitive tasks commonly used in IL benchmarking, such as reach and grasp; thus, success in `PnP` requires the learner to also succeed in these primitive tasks. Finally, by varying the number of objects, these tasks allows us to isolate the challenge of long horizon.

Training Data. Offline IL requires a set of demonstrations as training input. To arrive at this data, we first create a handcrafted expert policy for each task. Expert demonstrations $\tau \in \mathcal{D}_E$ are generated using this expert policy, ensuring successful task completion. Imperfect demonstrations $\tau \in \mathcal{D}_I$ are generated via two sources: noisy version of expert policy and randomly generated policies. For `PnP×1` and `PnP×2`, the data includes 25 expert and 75 imperfect demonstrations. For the more challenging `PnP×3`, the data includes 50 expert and 100 imperfect demonstrations.

Baselines. We benchmark against three offline imitation learning techniques: Behavioral cloning (BC), GoFAR, and g-DemoDICE. GoFAR is the most recent IL technique in the DICE family and learns goal-conditioned policies (Ma et al. 2022a). g-DemoDICE is a one-option equivalent of our algorithm. It can be seen as a simple extension of

[4] Please see the Appendix for additional details regarding experimental tasks, implementation details, and results.

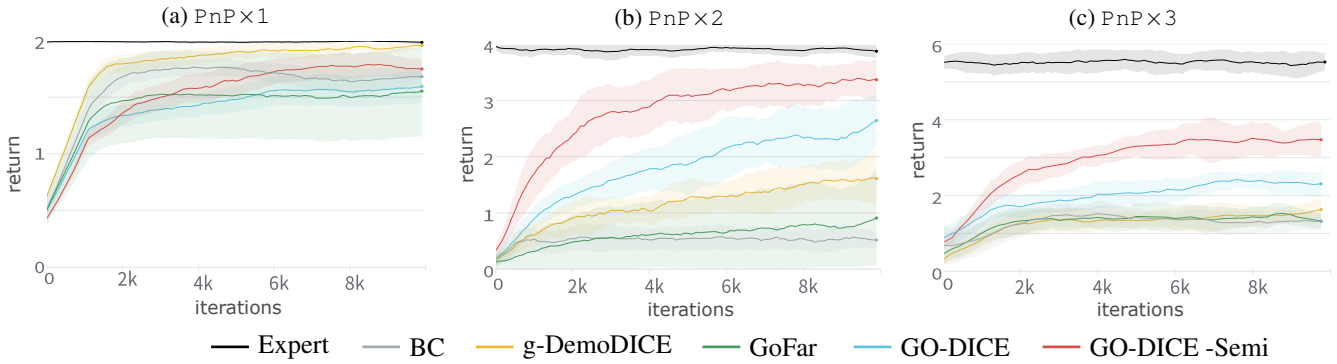


Figure 2: Performance curves comparing GO-DICE with baseline techniques across three benchmark tasks. Each plot represents the average return from 5 runs, with shaded regions indicating standard deviation.

DemoDICE (Kim et al. 2021) by incorporating goal conditioning. Similar to DemoDICE, it is designed to learn from both expert and imperfect demonstrations. All baselines and our technique receive the goal-conditioned demonstrations (i.e., $\mathcal{D}_I, \mathcal{D}_E$ with g) as inputs. To evaluate the ability to learn from auxiliary inputs, we also compare against the semi-supervised version of our approach, denoted as GO-DICE-Semi. This version also uses Algorithm 1 but additionally receives number of options $K_{\text{GO-DICE-Semi}}$ and sub-task labels for expert demonstrations (but not the imperfect demonstrations) as inputs. $K_{\text{GO-DICE-Semi}} = 3, 6$, and 9 was provided as an input for the three tasks, respectively.

Hyper-parameters. Each model underwent training for $N = 10k$ iterations. All DICE-based algorithms were regularized with a replay value of $\alpha = 0.05$. For GO-DICE, target network updates were managed using (M, λ) pairs: $(20, 0.95)$ for one-object, $(20, 0.5)$ for two-object, and $(50, 0.5)$ for three-object PnP tasks. The optimal values of the (M, λ)-tuple were chosen through a grid search of hyperparameters. After parameter tuning, the option counts in GO-DICE were selected as $K = 2$, 3, and 9 for the one-, two-, and three-object tasks, respectively.

Evaluation Metric. Performance is quantified via cumulative reward (averaged over 10 evaluation episodes), where each successful pick or place earns a reward of 1. The reward function is not known to the learning algorithms.

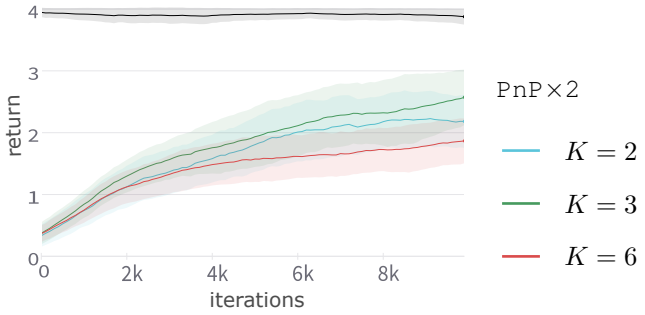
Expert Policy. Guided by the experimental evaluations of (Ding et al. 2019), we train an expert policy to produce expert demonstrations. This trained expert is able to achieve near-optimal performance. The variations in expert performance (Fig. 2) arise due to the sub-optimality of the expert policy in situations where the target goals of objects are in close proximity.

Results

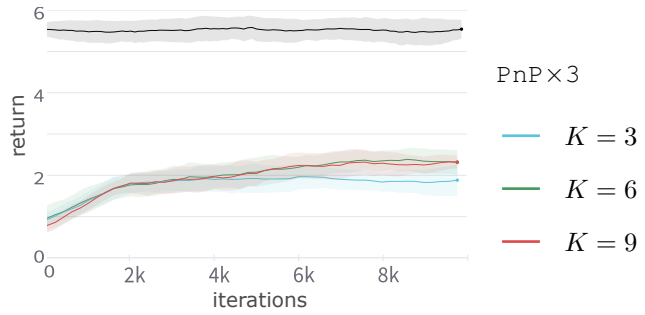
We now pose several research questions, conduct experiments to answer them, and finally present our findings.

R1. How does the performance of GO-DICE measure against the baselines in long-horizon tasks? Fig. 2 reveals a distinct trend. Most techniques perform near-optimally in the single-object task as seen in Figs. 2a. However, the performance disparity between GO-DICE and its baselines amplifies with the introduction of multi-object tasks, evident from Figs. 2b and 2c. The driving force behind this difference can be linked to GO-DICE's adeptness in discovering and decoding sub-tasks (such as reaching, grabbing, and placing objects) and seamlessly transitioning between them. However, the performance of GO-DICE diminishes in the three-object tasks. One reason for this decrement could be the close object proximities leading to unintended collisions, displacing already placed objects. This behavior can potentially be rectified with self-correcting expert demonstrations, an aspect we plan to explore in the future. When equipped with expert-driven trajectory segmentation, GO-DICE-Semi realizes even greater efficiency as reflected in its elevated average returns. *In conclusion, this experiment suggests that GO-DICE is capable of discerning and executing sub-task hierarchies, which are pivotal for successfully tackling long-horizon tasks.*

R2. How does the choice of hyperparameter K affect GO-DICE's learning performance? GO-DICE differentiates from baselines primarily by leveraging a set of K discrete options for task segmentation. In general, K is unknown that needs to be set as a hyperparameter. Through this experiment, we investigate the effect of K on model's convergence and overall performance, understanding if a specific option count optimally balances complexity with efficiency. In particular, we train GO-DICE with varying number of options. From Figs. 3a-3b it is evident that merely augmenting the number of options does not guarantee improved performance. This is demonstrated by the suboptimal results with $K = 6$ options in PnP×2 and no improvement for $K = 9$ options in PnP×3. Yet, when expert segmentations for these options are introduced in a semi-supervised setting, there is a marked enhancement in performance, as seen in Figs. 2b-2c. *These results suggest that increasing the option count with no supervision can boost performance up to a certain threshold. Beyond this, performance may decline, likely because redundant transitions overshadow the benefits of expressive task segmentation.*

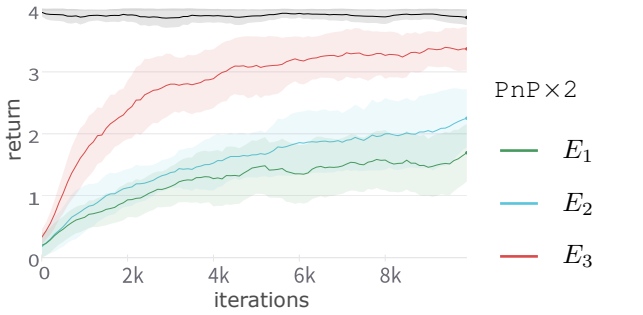


(a) PnP×2

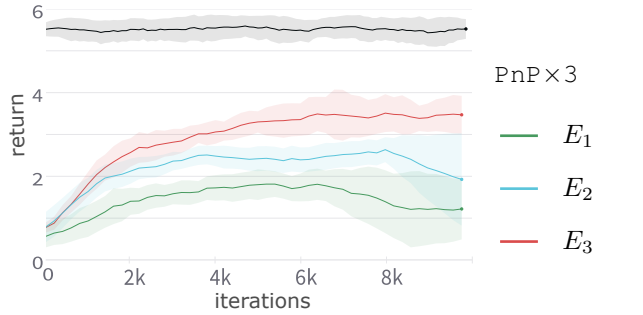


(b) PnP×3

Figure 3: GO-DICE: Effect of K on the return of learned policy (y-axis). The x-axis denotes the training iteration.



(a) PnP×2



(b) PnP×3

Figure 4: GO-DICE-Semi: Effect of sub-task labeling on return of learned policy (denoted on y-axis).

Task	$\pi_L^{\texttt{PnP}\times\texttt{n}}$	$\pi_L^{\texttt{PnP}\times\texttt{1}}$
PnP×2	2.9 ± 0.83	2.6 ± 0.49
PnP×3	3.5 ± 0.45	3.2 ± 1.13

Table 2: GO-DICE: Potential for Zero-Shot Transfer. Averages returns of the learned low-level policies (labeled in column) on long-horizon tasks (labeled in rows).

R3. How does sub-task labeling affect GO-DICE-Semi's learning performance? The semi-supervised version of GO-DICE boosts learning performance using sub-task labels provided by human experts. The notion of sub-task may differ across annotators. As such, we aim to identify and distinguish various means of segmentation and explore their influence on performance. To conduct these experiment, we identify three core primitives for the pick and place tasks: `reach`, `grasp`, and `place`. Based on these primitives, we explore three sub-task labeling methods: (i) based on the primitive alone: $\mathcal{C}^{E_1} = \{\texttt{reach}, \texttt{grasp}, \texttt{place}\}$; (ii) based on the object being manipulated: $\mathcal{C}^{E_2} = \{\texttt{object}_1, \ldots, \texttt{object}_n\}$; and (iii) based on both primitive and object: $\mathcal{C}^{E_3} = \{\texttt{reach}_{\texttt{object}_i}, \texttt{grasp}_{\texttt{object}_i}, \texttt{place}_{\texttt{object}_i}\}_{i=1}^{n}$. These three methods offer task segmentation at different levels of granularity and from different perspectives. Fig. 4 demonstrates that an expert-curated, finely-segmented sub-task labels (E_3) yield superior performance. At the same time, Figure 3 indicates that simply increasing K is insufficient. *This suggests expert guidance becomes particularly critical when tasks demand finer segmentations. Further, Figure 4 also confirms that GO-DICE-Semi is able to learn from different types of sub-task annotations.*

R4. Zero-shot Transfer: Can low-level policies from simpler tasks be used to execute more complex tasks without additional training? In this final experiment, we assess whether low-level policies learnt using GO-DICE are transferable to more complex tasks that utilize similar primitives. This experiment is exploratory, as GO-DICE is not designed for zero-shot transfer. Towards this question, we first train policies for PnP×1, PnP×2, and PnP×3 using GO-DICE-Semi with $\mathcal{C}^{E_3}$ labeling method. To realize zero-shot transfer, we then utilize $\pi_L^{\texttt{PnP}\times\texttt{1}}$ (i.e., the low-level policy of PnP×1) to complete PnP×2 and PnP×3. We compare the performance of this zero-shot transfer policy, with the policy trained directly on PnP×2 and PnP×3. As summarized in Table 2, we observe that the sub-task policies from PnP×1 (represented as $\pi_L^{\texttt{PnP}\times\texttt{1}}$), when applied to more-complex tasks perform comparably to the policies learned specifically for those tasks (represented as $\pi_L^{\texttt{PnP}\times\texttt{n}}$). *This exploratory analysis suggests that low-level policies learned by GO-DICE for simpler task fit well within the hierarchical structure established by GO-DICE for more complex tasks that involve similar options connotation.*

Video Demonstrations. The supplementary material (available at https://arxiv.org/abs/2312.10802) includes videos of both expert and learned behaviors. These videos showcase both successful instances and failure cases.

Conclusion

We introduce GO-DICE an approach for offline IL, which estimates stationary distribution ratios to derive goal-conditioned option-aware policies. By segmenting long-horizon demonstrations, GO-DICE discerns a hierarchy of sub-tasks, learning distinct micro-policies for each segment and a macro-policy for sub-task transitions, while maintaining the flexibility to adapt to changing goals across different tasks. We evaluate our approach on robotic manipulation tasks, which have been challenging for previous offline IL techniques due to their long horizon and changing goals. Our R1 experiment showcases GO-DICE's superior performance compared to recent offline IL baselines in these tasks. Through experiments R2 and R3, we also evaluate the effect of expert sub-task annotations and associated hyperparameters. Notably, our approach was able to leverage expert annotations of task segments to further enhance its learning performance. Beyond providing a promising approach for solving long-horizon tasks, these experiments also highlight the impact of auxiliary inputs for robot learning.

Limitations and Future Work. Our work also motivates several directions of future work. First, while GO-DICE is able to outperform recent baselines, the performance improvement is lower as the number of objects increases. Multiple reasons could be behind this observation, including difficulty in inferring sub-tasks over long horizons, complexity of multi-object manipulation, and need for additional training data. Future work that investigates the underlying root cause can enhance performance of offline IL on long-horizon tasks; GO-DICE can serve as a useful starting point for this investigation. Second, the evaluations (though conducted on challenging long-horizon tasks) are limited to the robotics domain. We encourage replication studies that evaluate the generality of the proposed approach on tasks derived from other domains.

Third, our experiments suggest that low-level policies learned with GO-DICE could be used for more challenging long-horizon tasks, given that a user or a different algorithm can detail the high-level policy or sub-task sequence. As such, a promising near-term direction is to explore the potential of GO-DICE for offline pre-training of online IL or RL techniques that address long-horizon tasks; currently, most techniques utilize behavioral cloning for pre-training. Finally, human-centered evaluation and subsequent development of human-guided IL techniques that utilize GO-DICE as a subroutine are of high interest.

Ethical Statement

Ensuring safety is essential for ethical deployment of learning-based AI systems. Our work contributes an approach to the paradigm of offline IL, which inherently facilitates safe learning by removing the requirement of (potentially unsafe) exploration.

Acknowledgements

We thank the anonymous reviewers for their detailed and constructive feedback. This research was supported in part by NSF award #2205454, the Army Research Office through Cooperative Agreement Number W911NF-20-2-0214, and Rice University funds.

References

Abbeel, P.; and Ng, A. Y. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*.

Andrychowicz, M.; Wolski, F.; Ray, A.; Schneider, J.; Fong, R.; Welinder, P.; McGrew, B.; Tobin, J.; Pieter Abbeel, O.; and Zaremba, W. 2017. Hindsight experience replay. *Advances in neural information processing systems*, 30.

Arora, S.; and Doshi, P. 2021. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297: 103500.

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

Chen, J.; Lan, T.; and Aggarwal, V. 2023. Option-Aware Adversarial Inverse Reinforcement Learning for Robotic Control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 5902–5908. IEEE.

Chernova, S.; and Thomaz, A. L. 2014. *Robot learning from human teachers*. Morgan & Claypool Publishers.

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

Ding, Y.; Florensa, C.; Abbeel, P.; and Phielipp, M. 2019. Goal-conditioned imitation learning. *Advances in neural information processing systems*, 32.

Fang, M.; Zhou, C.; Shi, B.; Gong, B.; Xu, J.; and Zhang, T. 2018. DHER: Hindsight experience replay for dynamic goals. In *International Conference on Learning Representations*.

Gao, C.; Jiang, Y.; and Chen, F. 2023. Transferring hierarchical structures with dual meta imitation learning. In *Conference on Robot Learning*, 762–773. PMLR.

Gupta, A.; Kumar, V.; Lynch, C.; Levine, S.; and Hausman, K. 2019. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. *arXiv preprint arXiv:1910.11956*.

Habibian, S.; Jonnavittula, A.; and Losey, D. P. 2022. Here's what I've learned: Asking questions that reveal reward learning. *ACM Transactions on Human-Robot Interaction (THRI)*, 11(4): 1–28.

Ho, J.; and Ermon, S. 2016. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29.

Jamgochian, A.; Buehrle, E.; Fischer, J.; and Kochenderfer, M. J. 2023. SHAIL: Safety-Aware Hierarchical Adversarial Imitation Learning for Autonomous Driving in Urban Environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 1530–1536. IEEE.

Jing, M.; Huang, W.; Sun, F.; Ma, X.; Kong, T.; Gan, C.; and Li, L. 2021. Adversarial option-aware hierarchical imitation

learning. In *International Conference on Machine Learning*, 5097–5106. PMLR.

Kim, G.-H.; Lee, J.; Jang, Y.; Yang, H.; and Kim, K.-E. 2022. LobsDICE: Offline Learning from Observation via Stationary Distribution Correction Estimation. *Advances in Neural Information Processing Systems*, 35: 8252–8264.

Kim, G.-H.; Seo, S.; Lee, J.; Jeon, W.; Hwang, H.; Yang, H.; and Kim, K.-E. 2021. Demodice: Offline imitation learning with supplementary imperfect demonstrations. In *International Conference on Learning Representations*.

Kostrikov, I.; Nachum, O.; and Tompson, J. 2019. Imitation learning via off-policy distribution matching. *arXiv preprint arXiv:1912.05032*.

Le, H.; Jiang, N.; Agarwal, A.; Dudík, M.; Yue, Y.; and Daumé III, H. 2018. Hierarchical imitation and reinforcement learning. In *International conference on machine learning*, 2917–2926. PMLR.

Lee, J.; Jeon, W.; Lee, B.; Pineau, J.; and Kim, K.-E. 2021. Optidice: Offline policy optimization via stationary distribution correction estimation. In *International Conference on Machine Learning*, 6120–6130. PMLR.

Li, Y.; Song, J.; and Ermon, S. 2017. Infogail: Interpretable imitation learning from visual demonstrations. *Advances in neural information processing systems*, 30.

Ma, J. Y.; Yan, J.; Jayaraman, D.; and Bastani, O. 2022a. Offline goal-conditioned reinforcement learning via f-advantage regression. *Advances in Neural Information Processing Systems*, 35: 310–323.

Ma, Y.; Shen, A.; Jayaraman, D.; and Bastani, O. 2022b. Versatile offline imitation from observations and examples via regularized state-occupancy matching. In *International Conference on Machine Learning*, 14639–14663. PMLR.

Nachum, O.; Dai, B.; Kostrikov, I.; Chow, Y.; Li, L.; and Schuurmans, D. 2019. Algaedice: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*.

Nasiriany, S.; Gao, T.; Mandlekar, A.; and Zhu, Y. 2023. Learning and Retrieval from Prior Data for Skill-based Imitation Learning. In *Conference on Robot Learning*, 2181–2204. PMLR.

Orlov-Savko, L.; Jain, A.; Gremillion, G. M.; Neubauer, C. E.; Canady, J. D.; and Unhelkar, V. 2022. Factorial Agent Markov Model: Modeling Other Agents' Behavior in presence of Dynamic Latent Decision Factors. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 982–990.

Osa, T.; Pajarinen, J.; Neumann, G.; Bagnell, J. A.; Abbeel, P.; Peters, J.; et al. 2018. An algorithmic perspective on imitation learning. *Foundations and Trends® in Robotics*, 7(1-2): 1–179.

Plappert, M.; Andrychowicz, M.; Ray, A.; McGrew, B.; Baker, B.; Powell, G.; Schneider, J.; Tobin, J.; Chociej, M.; Welinder, P.; et al. 2018. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*.

Pomerleau, D. A. 1991. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1): 88–97.

Puterman, M. L. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Quintero-Pena, C.; Chamzas, C.; Sun, Z.; Unhelkar, V.; and Kavraki, L. E. 2022. Human-guided motion planning in partially observable environments. In *2022 International Conference on Robotics and Automation (ICRA)*, 7226–7232. IEEE.

Ranchod, P.; Rosman, B.; and Konidaris, G. 2015. Nonparametric bayesian reward segmentation for skill discovery using inverse reinforcement learning. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 471–477. IEEE.

Ravichandar, H.; Polydoros, A. S.; Chernova, S.; and Billard, A. 2020. Recent advances in robot learning from demonstration. *Annual review of control, robotics, and autonomous systems*, 3: 297–330.

Ross, S.; Gordon, G.; and Bagnell, D. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 627–635. JMLR Workshop and Conference Proceedings.

Schaul, T.; Horgan, D.; Gregor, K.; and Silver, D. 2015. Universal value function approximators. In *International conference on machine learning*, 1312–1320. PMLR.

Seo, S.; and Unhelkar, V. 2022. Semi-Supervised Imitation Learning of Team Policies from Suboptimal Demonstrations. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Sharma, A.; Sharma, M.; Rhinehart, N.; and Kitani, K. M. 2018. Directed-info gail: Learning hierarchical policies from unsegmented demonstrations using directed information. *arXiv preprint arXiv:1810.01266*.

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

Todorov, E.; Erez, T.; and Tassa, Y. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, 5026–5033. IEEE.

Unhelkar, V. V.; Li, S.; and Shah, J. A. 2020. Semi-supervised learning of decision-making models for human-robot collaboration. In *Conference on Robot Learning*, 192–203. PMLR.

Unhelkar, V. V.; and Shah, J. A. 2019. Learning models of sequential decision-making with partial specification of agent behavior. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 2522–2530.

Wang, Y.; Xu, C.; Du, B.; and Lee, H. 2021. Learning to weight imperfect demonstrations. In *International Conference on Machine Learning*, 10961–10970. PMLR.

Wu, Y.-H.; Charoenphakdee, N.; Bao, H.; Tangkaratt, V.; and Sugiyama, M. 2019. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, 6818–6827. PMLR.

Wu, Z.; Lian, W.; Unhelkar, V.; Tomizuka, M.; and Schaal, S. 2021. Learning dense rewards for contact-rich manipulation tasks. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 6214–6221. IEEE.