Hierarchical Imitation Learning of Team Behavior from Heterogeneous Demonstrations

Sangwon Seo
Rice University
Houston, TX, USA
sangwon.seo@rice.edu

Vaibhav Unhelkar
Rice University
Houston, TX, USA
vaibhav.unhelkar@rice.edu

ABSTRACT

Successful collaboration requires team members to stay aligned, especially in complex sequential tasks. Team members must dynamically coordinate which subtasks to perform and in what order. However, real-world constraints like partial observability and limited communication bandwidth often lead to suboptimal collaboration. Even among expert teams, the same task can be executed in multiple ways. To develop multi-agent systems and human-AI teams for such tasks, we are interested in data-driven learning of multimodal team behaviors. Multi-Agent Imitation Learning (MAIL) provides a promising framework for data-driven learning of team behavior from demonstrations, but existing methods struggle with heterogeneous demonstrations, as they assume that all demonstrations originate from a single team policy. Hence, in this work, we introduce DTIL: a hierarchical MAIL algorithm designed to learn multimodal team behaviors in complex sequential tasks. DTIL represents each team member with a hierarchical policy and learns these policies from heterogeneous team demonstrations in a factored manner. By employing a distribution-matching approach, DTIL mitigates compounding errors and scales effectively to long horizons and continuous state representations. Experimental results show that DTIL outperforms MAIL baselines and accurately models team behavior across a variety of collaborative scenarios.

CCS CONCEPTS

• **Computing methodologies** → *Learning latent representations*; **Learning from demonstrations**; *Multi-agent systems*; **Apprenticeship learning**; *Semi-supervised learning settings.*

KEYWORDS

Multi-Agent Imitation Learning, Teamwork, Behavior Modeling

ACM Reference Format:
Sangwon Seo and Vaibhav Unhelkar. 2025. Hierarchical Imitation Learning of Team Behavior from Heterogeneous Demonstrations. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

1 INTRODUCTION

Imitation learning (IL) is a paradigm for training agent behaviors using demonstrations [1]. IL typically assumes that the demonstrations are generated by an expert following a single, optimal policy. Under this assumption, IL algorithms learn an estimate of the expert's policy. Compared to reinforcement learning (RL), another popular framework for training agents, IL offers a key advantage in many practical applications: it does not require hand-engineered rewards. Designing rewards often requires extensive domain expertise, and in many cases, informative rewards can be challenging for end-users to engineer. Moreover, even with hand-engineered rewards, the agent can learn suboptimal or reward hacking behaviors [3, 40]. In contrast, often end-users can more easily provide demonstrations of desirable agent behavior [4, 8, 27].

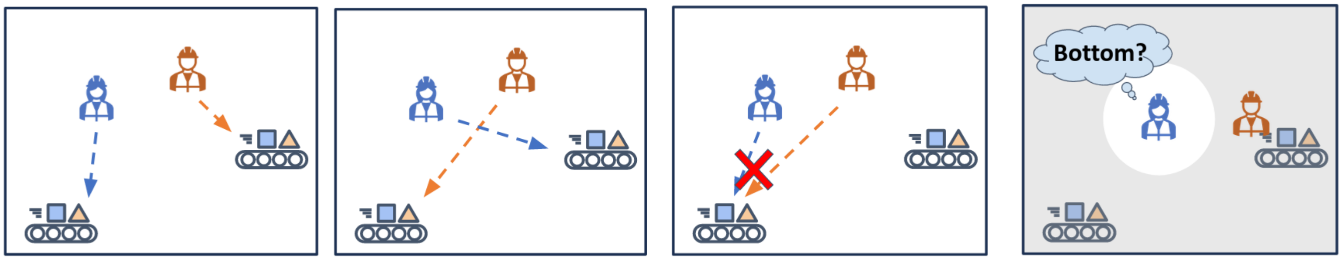
Despite this advantage, traditional IL algorithms face challenges while learning complex behaviors from demonstration collected by human end-users. A key challenge is that conventional IL methods consider a *single agent* that has *full observability* of the environment. In reality, human end-users often perform complex tasks as teams rather than individually. As a result, demonstrations available for learning often involve multiple agents interacting with one another and their environment. Since the dynamics of the environment depend on these interactions, specialized IL algorithms are needed to learn multi-agent behaviors.

To address this challenge, recent approaches have extended imitation learning to multi-agent settings [6, 38, 44, 46]. These *Multi-agent IL* (MAIL) methods typically assume that demonstrations are generated by a single, well-defined multi-agent policy [23]. However, due to practical challenges, this assumption is difficult to satisfy in complex multi-agent tasks encountered in the real world. As illustrated in Fig. 1a, complex multi-agent tasks often consist of multiple subtasks. Demonstrations of such tasks frequently involve a variety of subtask allocations, including suboptimal ones. Suboptimality can arise from various factors, including the decentralized nature of multi-agent task execution and the partial observability that individual agents have of the task environment and other agents [29, 32, 34]. Consequently, real-world demonstrations inherently exhibit multiple modes of multi-agent behavior, diverging from the assumptions of most existing MAIL methods.

In single-agent settings, *hierarchical imitation learning* methods that explicitly impose a hierarchical structure on an expert's decision-making have been employed to address the challenge of modeling multimodal behaviors [15, 16, 26, 36, 37, 42]. However, little work has been done to extend these methods to multi-agent settings. To our knowledge, *Bayesian Team Imitation Learner* (BTIL) is the only approach that applies hierarchical imitation learning in a multi-agent context [35]. However, BTIL is built on variational inference and tabular representations, making it difficult to scale to tasks with high-dimensional states and long horizons.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

An extended version of this paper, which includes supplementary material mentioned in the text, is available at http://tiny.cc/dtil-appendix



(a) Multiple Near-Optimal Strategies **(b) Suboptimal Teamwork** **(c) Partial Observability**

Figure 1: Motivating Example: Consider a team whose members must coordinate on the fly to complete subtasks at two conveyor belts. Each member has limited observability, perceiving only their immediate surroundings. For example, the unshaded area for the blue person in (c). As shown in (a), this task allows multiple near-optimal strategies, enabling teams to execute it in different ways based on their shared preferences. However, practical constraints – such as partial observability – can lead to suboptimal coordination and team performance. For instance, if multiple members gather at the same subtask location, it results in inefficient task allocation, where one subtask remains unattended while two members redundantly perform the same task (b). Like many real-world scenarios, this task engenders heterogeneous and potentially suboptimal demonstrations of teamwork. This paper focuses on learning models of team behavior in this challenging setting from demonstrations.

To enable MAIL for more complex tasks, this paper introduces *Deep Team Imitation Learner* (DTIL), a multi-agent hierarchical imitation learning algorithm. DTIL rigorously extends single-agent hierarchical imitation learning to collaborative tasks conducted in partially observable environments, enabling the learning of multimodal team behaviors from heterogeneous demonstrations, even in tasks with long horizons and continuous state representations. At its core, DTIL leverages the state-action distribution matching framework, a mainstay of state-of-the-art IL methods due to its performance and scalability [10, 14].

While recent hierarchical IL methods have applied distribution matching in single-agent settings [16, 36], its extension to learning multimodal team behavior under partial observability remains unexplored. Notably, key theoretical results in distribution-matching-based hierarchical imitation learning, such as Theorem 1 (Bijection) in [16] and Theorem 2.4 (Convergence) in [36], have only been proven under full observability. Thus, additional theoretical justification is required for learning multimodal team behavior in partially observable settings. Hence, we first extend these theoretical results to the partially observable multi-agent hierarchical imitation learning setting. Next, leveraging these theoretical results, we derive DTIL to effectively learn the hierarchical team policies. Finally, we evaluate DTIL on a suite of collaborative tasks, demonstrating that it outperforms MAIL baselines in modeling team behavior across multiple scenarios.

2 RELATED WORKS

We begin with a brief overview of related research.

2.1 Imitation Learning of Multimodal Behavior

Extensive research has been conducted on learning multimodal behaviors from demonstrations. In works such as [13, 21], the authors extend Generative Adversarial Imitation Learning (GAIL) to learn a policy that depends on a learned latent state. This learned latent state effectively encodes different modes of the behavior. However, these methods assume the latent states remain static during a task execution; thus, their methods are unsuitable for modeling agent behavior whose latent states can change during the tasks. Other approaches, such as [30], use Conditional Variational Autoencoders (CVAE) to capture multimodal human behavior, but the learned latent space responsible for generating multimodality lacks grounding and difficult to associate with specific subtasks.

Informed by the Option framework [41], another line of research leverages hierarchical policies to model multimodal behavior. Hierarchical policies typically consider two levels: high-level policies that govern decision-making over extended temporal intervals, and low-level policies responsible for executing specific actions within shorter time frames [16, 18]. To learn such policies from demonstrations, various approaches have been explored; e.g., variational inference [26, 42], hierarchical behavior cloning [17, 18, 47], and hierarchical variants of GAIL [7, 16, 20, 37]. Most recently, [36] propose a factored distribution-matching approach to train hierarchical policies. While all these methods show remarkable performance in single-agent tasks, their extension to multi-agent scenarios has been rarely explored and often lacks theoretical grounds.

2.2 Multi-agent Imitation Learning

Learning team behavior from demonstration can be framed as a multi-agent imitation learning problem. Since [38] introduced the multi-agent variant of generative adversarial imitation learning (called MA-GAIL), several extensions have been proposed to enhance its training efficiency and scalability [5, 6, 24, 31, 44, 46]. However, these methods generally assume that the demonstrations originate from a single multi-agent policy, limiting their ability to capture diverse team behaviors. Despite the importance of accounting for multimodality when modeling team behavior, only a few approaches have incorporated latent states into MAIL. Among these, [19] model agent roles as latent variables, while [43] represent strategies as latent features. However, both methods assume static latent states and do not consider their dynamics. To address this gap, [35] propose Bayesian Team Imitation Learner (BTIL), a multi-agent extension of [42], which can learn hierarchical policies of all team members from team demonstrations. Nonetheless, BTIL struggles with large, complex tasks and suffers from compounding errors. In contrast, DTIL overcomes these limitations by utilizing function approximators (e.g., neural networks) and augmenting demonstrations with online samples collected during training.

3 BACKGROUND

In this section, we present preliminaries on distribution-matching-based imitation learning and introduce the mathematical model of team behavior.

3.1 Imitation Learning via Distribution Matching

Using the Markov Decision Process (MDP) framework, an agent's behavior is defined by a policy $\pi(a|s)$, which represents the probability distribution of an action a given a state s. The goal of imitation learning is to minimize the discrepancy (represented as a loss L) between the learner's policy π and the expert's policy π_E: $\min_\pi L(\pi, \pi_E)$. However, due to the inaccessibility of π_E, this objective is often ill-defined and highly challenging to solve.

To address this, Ho and Ermon [14] reformulate imitation learning as a problem of matching the occupancy measures of the learner and the expert. The (normalized) occupancy measure of a policy π is defined as $\rho_\pi(s, a) \doteq (1-\gamma)\sum_{t=0}^{\infty} \gamma^t p(s^t{=}s, a^t{=}a|\pi)$, implying the stationary distribution over states s and actions a induced by π. Thanks to the one-to-one correspondence between a policy $\pi(s|a)$ and its occupancy measure $\rho_\pi(s, a)$ [28], matching the occupancy measures is equivalent to matching the policies. This can be formalized as:

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

where ρ_π is the learner's occupancy measure, ρ_E is the expert's occupancy measure, and D_f denotes the f-divergence [11]. While direct access to ρ_E is still infeasible, it can be approximated using the empirical distribution calculated from expert demonstrations D. Due to its performance and scalability, since its introduction, the distribution matching approach has become a mainstream technique in imitation learning, giving rise to numerous variants, including the following multi-agent and hierarchical ones.

Multi-Agent Variants. Assuming a unique equilibrium in multi-agent behaviors, Song et al. [38] formulate an occupancy measure matching problem in multi-agent settings:

$$\arg\min_{\pi} \sum_{i=1}^{n} D_f\left(\rho_{\pi_i,\pi_{-i}}(s,a_i) \middle\| \rho_{\pi_E}(s,a_i)\right) \tag{1}$$

where π_{-i} is joint policies except the i-th agent's policy, π_E denotes multi-agent expert policy at equilibrium, and $\rho_{\pi_i,\pi_{-i}}(s,a_i) \doteq (1-\gamma)\sum_{t=0}^{\infty}\gamma^t p(s^t{=}s, {a_i}^t{=}a_i|\pi_i, \pi_{-i})$. This objective function implies that we can iteratively minimize the objective with respect to individual policies $\pi_1, \cdots, \pi_n$, and the updates can be calculated similarly to the single-agent problem by considering other agents' policies as part of the environment dynamics.

Hierarchical Variants. Jing et al. [16] extend occupancy measure matching approach to hierarchical imitation learning. They model an agent behavior as an option policy $\tilde{\pi} = (\pi_L, \pi_H)$, where $\pi_L(a|s,x)$ and $\pi_H(x|s,x^-)$ are referred to as a low- and high-level policies, respectively, with x being an option. Additionally, they define an option-occupancy measure corresponding to $\tilde{\pi}$ as

$$\rho_{\tilde{\pi}}(s,a,x,x^-) \doteq (1-\gamma)\sum_{t=0}^{\infty}\gamma^t p(s^t{=}s, a^t{=}a, x^t{=}x, x^{t-1}{=}x^-|\tilde{\pi})$$

and prove the one-to-one correspondence between $\tilde{\pi}$ and $\rho_{\tilde{\pi}}$. Thus, hierarchical imitation learning can also be cast as distribution matching between two option-occupancy measures $\rho_{\tilde{\pi}}$ and ρ_E:

$$\arg\min_{\tilde{\pi}} D_f\left(\rho_{\tilde{\pi}}(s,a,x,x^-) \middle\| \rho_{\pi_E}(s,a,x,x^-)\right) \tag{2}$$

3.2 Model of Team Behavior

We borrow a model of team behavior introduced in [33] to represent our multi-agent behavior in sequential team tasks. The model consists of a decentralized partially observable MDP (Dec-POMDP) to capture the task dynamics and an Agent Markov models (AMM) to represent agents' multimodal behavior [25, 42].

Dec-POMDP. Dec-POMDP is a probabilistic model representing the dynamics of the partially observable sequential multi-agent tasks. It is expressed as a tuple $\mathcal{M} = (n, S, \times A_i, T, \mu_0, \times\Omega_i, \{O_i\}, \gamma)$, where n is the number of agents, S is the set of states s, A_i is the set of the i-th agent's actions a_i, Ω_i is the set of the i-th agent's observations o_i, $T(s'|s,a)$ denotes the probability of a state s' transitioning from a state s and a joint action $a{=}(a_1, \cdots, a_n)$, $\mu_0(s)$ is an initial state distribution, and γ is a discount factor. We define an extended action set as $A_i^+ = A_i \cup \{\#\}$, where the symbol # denotes "Not Available". $O_i : S \times A^+ \to \Omega_i$ is an observation function for the i-th agent, which maps a pair of state s and previous joint action a^- to an individual observation $o_i \in \Omega_i$. The values of previous actions a_i^- at time $t{=}0$ are set as #. We denote capital letters without subscripts as joint spaces or joint functions, e.g., $A = \times A_i$ for a joint action space and $O = \prod_i O_i$ for a joint observation function.

Agent Markov Model (AMM). When faced with complex team tasks, humans typically break them down into subtasks and dynamically adjust their plan regarding which subtasks to perform and in what order. Once they decide on the next subtask, they execute the necessary actions to complete it. AMM is designed to account for this hierarchical human behavior, and thus, is equivalent to hierarchical policies of Option framework with a one-step option [16, 41]. Given a task model $\mathcal{M}$, AMM defines the behavior model of the i-th agent as a tuple $(X_i, \pi_i, \zeta_i; \mathcal{M})$, where X_i is the set of the possible subtasks, $\pi_i(a_i|o_i, x_i)$ denotes a subtask-driven policy, and $\zeta_i(x_i|o_i, x_i^-)$ is the probability of an agent choosing their next subtask x_i based on an observation o_i and the current subtask x_i^{-1}. Following [16], we define the value of the previous subtask at time $t{=}0$ as # and express the initial distribution of subtasks as $\zeta_i(x_i|o_i, x_i^-{=}\#)$. Similar to previous works [16, 35], we assume the set of possible subtasks, X, is finite and given as prior knowledge. We then represent the AMM for the i-th agent simply as (π_i, ζ_i), omitting the non-learnable components X_i and $\mathcal{M}$.

[1]In the following sections, we will omit the subscript i from the function inputs, e.g., $\pi_i(a|o,x) \doteq \pi_i(a_i|o_i,x_i)$, when it is clear the functions pertain to individual observations, actions, and subtasks.

4 PROBLEM FORMULATION

While Seo et al. [33] emphasize the need for modeling team behavior under partial observability and propose a corresponding mathematical model, few multi-agent imitation learning methods have been developed to address such complex teamwork models. To our knowledge, BTIL is the only approach to learning multi-model team behavior from demonstrations [35]. However, BTIL does not account for partial observability, and its applicability is limited to small, discrete domains, as both high- and low-level policies are constrained to categorical distributions. Thus, a practical method for learning team behavior models that addresses multimodality, partial observability, and scalability is still lacking. To derive such a method, we first formulate the problem of multi-agent imitation learning from heterogeneous demonstrations.

4.1 Formalizing Hierarchical Multi-Agent Distribution Matching

Inspired by the recent success of distribution matching-based imitation learning, we aim to apply this method to learn the team behavior model. Similar to the hierarchical variants for fully observable single-agent scenarios in Sec. 3.1, we define oaxx-occupancy measure for the i-th agent given a task model $\mathcal{M}$ and joint agent models $\mathcal{N}_{1:n}$ as:

$$\rho_{\mathcal{N}_i,\mathcal{N}_{-i}}(o_i, a_i, x_i, x_i^-) \doteq (1-\gamma)\sum_{t=0}^{\infty}\gamma^t p({o_i}^t{=}o_i, {a_i}^t{=}a_i, {x_i}^t{=}x_i, {x_i}^{t-1}{=}x_i^- | \mathcal{N}_{1:n}, \mathcal{M})$$

The notation $\rho_{\mathcal{N}_i,\mathcal{N}_{-i}}$, borrowed from MA-GAIL [38], represents the occupancy measure induced by the agent i's policy $\mathcal{N}_i$ and other agents' policy $\mathcal{N}_{-i}$. Unless ambiguous, we simply denote $\rho_i \doteq \rho_{\mathcal{N}_i,\mathcal{N}_{-i}}$.

By combining this occupancy measure with the multi-agent variant of distribution matching introduced in Sec. 3.1, we can formulate the distribution-matching problem for team behavior with n agents as follows:

$$\arg\min_{\mathcal{N}_{1:n}} \sum_{i=1}^{n} D_f\left(\rho_i(o_i, a_i, x_i, x_i^-)\middle\|\rho_E(o_i, a_i, x_i, x_i^-)\right) \tag{3}$$

where ρ_E denotes the oaxx-occupancy measure of the expert team model (π_E, ζ_E). In Sec. 5, we provide theoretical justification for using Eq. 3 as the imitation learning objective. While this extension seems natural, the theoretical results in the existing literature are insufficient to guarantee that occupancy measure matching is equivalent to policy matching in partially observable multi-agent scenarios.

Additionally, informed by IDIL [36], we aim to adopt a factored approach to minimize the objective above. This factored approach enables us to leverage existing non-adversarial imitation learning methods, such as IQLearn [10], which demonstrate more stable training compared to generative adversarial approaches. However, since the theoretical foundations for factored distribution matching are also developed under the assumption of full observability, further theoretical analysis is necessary to ensure its applicability in partially observable multi-agent settings. We provide this analysis in Sec. 5.

4.2 Problem Statement

Since we cannot know which subtasks each member has in mind at the time of task execution, demonstrations contain only observations and actions. We define the set of d demonstrations as $D = \{\tau_m\}_{m=1}^d$, where $\tau{=}(o,a)^{0:h}$ is a trajectory of a team's task execution. We denote an individual trajectory of the i-th agent and the set of them as $\tau_i = (o_i, a_i)^{0:h}$ and $D_i = \{\tau_{m,i}\}_{m=1}^d$, respectively, adding a subscript i. The sequence of expert's subtasks corresponding to the m-th demonstration is defined as $\chi_m = ({x_m}^{0:h})$. Since the labels of the subtasks are challenging to collect in practice, only a small portion of them (e.g., for $l(\leq d)$ demonstrations) are optionally available. Thus, our goal is to learn agent models $\{(\pi,\zeta)\}_{1:n}$ that exhibit the behaviors of n team members from the following inputs: a multi-agent task model $\mathcal{M}$, the set of possible subtasks X, heterogeneous demonstrations D, and optionally partial labels of subtasks $\{\chi_m\}_{m=1}^l$.

5 LEARNING MODEL OF TEAMWORK VIA FACTORED DISTRIBUTION MATCHING

As mentioned in Sec. 4.1, matching occupancy measures does not always guarantee matching the team behavior models unless a one-to-one correspondence is established between the agent model (i.e., partial observation-based hierarchical policy) $\mathcal{N}_i = (\pi_i, \zeta_i)$ and its oaxx-occupancy measure ρ_i. Therefore, we first present this one-to-one correspondence, which extends the *Theorem 1* from [16] to multi-agent partially-observable settings.

Theorem 5.1. *For each agent i, given a multi-agent task model $\mathcal{M}$ and other agents' models $\mathcal{N}_{-i}$, suppose ρ_i is the oaxx-occupancy measure for a stationary agent model $\mathcal{N}_i = (\pi_i, \zeta_i; \mathcal{M})$ where*

$$\zeta_i(x|o,x^-) = \frac{\sum_a \rho_i(o,a,x,x^-)}{\sum_{a,x}\rho_i(o,a,x,x^-)},\ \pi_i(a|o,x) = \frac{\sum_{x^-}\rho_i(o,a,x,x^-)}{\sum_{x^-,a}\rho_i(o,a,x,x^-)}.$$

Then, $\mathcal{N}_i = (\pi_i, \zeta_i; \mathcal{M})$ is the only agent model whose oaxx-occupancy measure is ρ_i.

This can be proved similarly to *Theorem 1* of [16] after deriving the stationary distributions of policy $\tilde{\pi}(w|v)$ and state transition $\tilde{T}(v'|v,w)$, where $v \doteq (o,x^-)$ and $w \doteq (x,a)$. The complete proof is provided in the Appendix. This theorem implies that we can consider the imitation learning of agent models $\mathcal{N}_{1:n}$ as matching the oaxx-occupancy measures between $\rho_{\mathcal{N}_i,E_{-i}}$ and ρ_E for all i. Here, $\rho_{\mathcal{N}_i,E_{-i}}$ denotes the oaxx-occupancy measure induced by the i-th agent model $\mathcal{N}_i$ with other agents' models given as expert models $E_{-i} \doteq (\pi_{E-i}, \zeta_{E-i})$. Thus, we can factorize the occupancy measure matching of the joint team model as follows: $\arg\min_{\mathcal{N}_{1:n}}\sum_{i=1}^n D_f\left(\rho_{\mathcal{N}_i,E_{-i}}(\cdot)\middle\|\rho_E(\cdot)\right)$. Due to the one-to-one correspondence, the following two problems lead to the same optimal solution, $\mathcal{N}_E$:

$$\begin{aligned}&\arg\min_{\mathcal{N}_{1:n}}\sum_{i=1}^{n} D_f\left(\rho_{\mathcal{N}_i,E_{-i}}(\cdot)\middle\|\rho_E(\cdot)\right)\\ &= \arg\min_{\mathcal{N}_{1:n}}\sum_{i=1}^{n} D_f\left(\rho_{\mathcal{N}_i,\mathcal{N}_{-i}}(\cdot)\middle\|\rho_E(\cdot)\right) = \mathcal{N}_E.\end{aligned}$$

This justifies our objective function, Eq 3, for learning the expert team behavior model via distribution matching.

Algorithm 1 DTIL: Deep Team Imitation Learner

1: **Input**: Data $D = \{\tau_m\}_{m=1}^{d}$ and $\{\chi_m\}_{m=1}^{l}$.
2: **Initialize**: (θ_i, ϕ_i) for all $i = 1:n$ where $\mathcal{N}_{\theta_i,\phi_i} = (\pi_{\theta_i}, \zeta_{\phi_i})$
3: **repeat**
4: *E-step*: Infer expert intents $\{\chi_m\}_{m=l}^{d}$ with D and $(\pi_\theta^k, \zeta_\phi^k)$ for all $i = 1:n$; and define $\tilde{D} \doteq D \cup \{\hat{\chi}_m\}_{m=1}^{d}$
5: Collect rollouts $R = \{(o, x, a)^{0:h}\}$ using $(\pi_\theta^k, \zeta_\phi^k)$
6: *M-step*: Update $\pi_{\theta_i}^{k+1}$ via Eq. 5 and $\zeta_{\phi_i}^{k+1}$ via Eq. 6 using $\tilde{D}_i, R_i$ for all $i = 1:n$
7: **until** Convergence

Seo and Unhelkar [36] suggest that matching oaxx-occupancy measure amounts to matching two factored occupancy measures, $\rho(o, a, x)$ and $\rho(o, x, x^-)$, simultaneously with their expert counterparts in the single-agent problem. We refer to these factored occupancy measures as oax-occupancy measure and oxx-occupancy measure, respectively, and define them for each agent i as:

$$\rho_i(o, a, x) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t p({o_i}^t{=}o_i, {a_i}^t{=}a_i, {x_i}^t{=}x_i|\mathcal{N}_{1:n}, \mathcal{M})$$

$$\rho_i(o, x, x^-) = (1-\gamma)\sum_{t=0}^{\infty}\gamma^t p({o_i}^t{=}o_i, {x_i}^t{=}x_i, {x_i}^{t-1}{=}x_i^-|\mathcal{N}_{1:n}, \mathcal{M})$$

With the factored occupancy measures, we can further factorize Eq. 3 as follows:

$$\arg\min_{\mathcal{N}_{1:n}} \sum_{i=1}^{n}\Big(D_f\left(\rho_i(o_i, a_i, x_i)\big\|\rho_E(o_i, a_i, x_i)\right) + D_f\left(\rho_i(o_i, x_i, x_i^-)\big\|\rho_E(o_i, x_i, x_i^-)\right)\Big) \quad (4)$$

The proof for Eq. 4, along with the adjusted theorems that formulate this factored objective for the multi-agent scenario, is provided in the Appendix.

6 DEEP TEAM IMITATION LEARNER

With the theoretical foundations established in the previous section, we now present DTIL, a practical algorithm designed to minimize Eq. 4. The distribution matching framework enables policies to be represented using deep neural networks and efficiently learns them by leveraging additional interactions with the environment. As a result, DTIL is capable of learning team behavior models even in highly complex tasks.

As mentioned in Sec. 3.1, the expert occupancy measure is typically estimated from expert demonstrations, i.e., $\rho(o, a, x, x^-) \approx \mathbb{E}_D[\mathbb{1}(o, a, x, x^-)]$. However, as our demonstration D does not contain the labels of subtasks, we cannot compute this empirical distribution. Thus, similar to [16], we take an expectation-maximization (EM) approach to iteratively optimize Eq. 4. Alg. 1 outlines DTIL. In line 4 (E-step), it predicts unknown expert intents from D using the current estimate of agent models $(\pi_{\theta_i}^k, \zeta_{\phi_i}^k)$. In line 5, it collects online samples by interacting with the environment. Then, in line 6 (M-step), it updates agent model parameters (θ, ϕ) via occupancy measure matching.

E-step. For each iteration k, DTIL infers the unknown subtasks of demonstration $\tau = (o^{0:h}, a^{0:h})$ based on the maximum a posteriori (MAP) estimation. Given the current estimate of agent models $\mathcal{N}_{\theta,\phi}^k = (\pi_\theta^k, \zeta_\phi^k)$, we can express the MAP estimation as follows: $\hat{\chi} = \arg\max_{x^{0:h}} p(x^{0:h}|o^{0:h}, a^{0:h}, \mathcal{N}_{\theta,\phi}^k)$. Similar to the Viterbi algorithm, this can be effectively computed via dynamic programming [16, 35]. Since its computation can be decentralized for each agent, its time complexity is $O(nh|\bar{X}|^2)$ where $|\bar{X}| \doteq \max_{i=1:n}|X_i|$. The derivations are provided in the Appendix. With this estimate, we can obtain subtask-augmented trajectories $\bar{\tau} = (o, a, \hat{x})^{0:h}$. From $\tilde{D} \doteq D \cup \{\chi_m\}_{m=1}^{d} = \{(o, a, \hat{x})^{0:h}\}$, we can compute the estimates of the expert occupancy measures for the k-th iteration, denoted by $\rho_E^k(o, a, x, x^-)$, $\rho_E^k(o, a, x)$, and $\rho_E^k(o, x, x^-)$, respectively.

M-step. We incrementally update the agent model parameters (θ, ϕ) to minimize the difference between the learner's occupancy measure $\rho_\mathcal{N}$ and the k-th estimate of expert occupancy measure ρ_E^k. Similar to IDIL [36], DTIL takes a factored approach to minimize Eq. 4. Specifically, when updating the low-level policy parameters θ_i, it assumes ζ_i is fixed and minimizes only the first term of Eq. 4 with respect to π_i:

$$\arg\min_{\pi_i} D_f\left(\rho_{\pi_i}(o_i, a_i, x_i)\Big\|\rho_E^k(o_i, a_i, x_i)\right) \quad (5)$$

If we introduce $u \doteq (o, x)$, this is the same as the occupancy-measure matching of a conventional policy $\pi(a|u) \doteq \pi(a|o, x)$.

Similarly, DTIL updates the high-level policy parameters ϕ_i by only minimizing the second term of Eq. 4 with respect to ζ_i, fixing π_i:

$$\arg\min_{\zeta_i} D_f\left(\rho_{\zeta_i}(o_i, x_i, x_i^-)\Big\|\rho_E^k(o_i, x_i, x_i^-)\right) \quad (6)$$

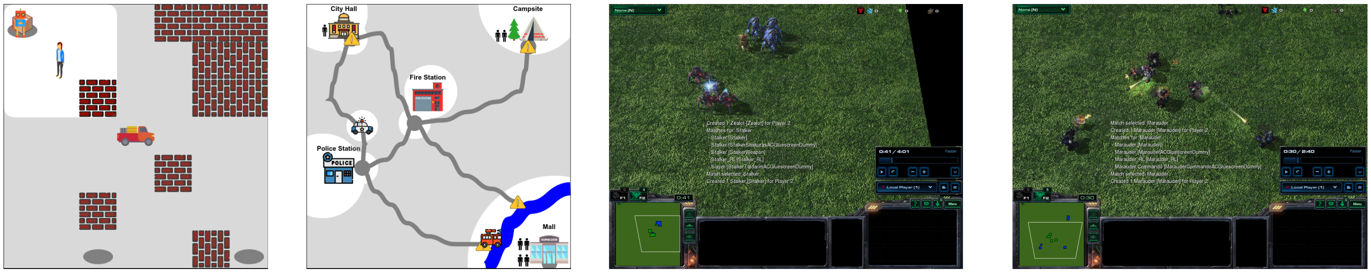
This also reduces to conventional imitation learning of a policy $\pi(x|v) \doteq \zeta(x|o, x^-)$, if we define $v \doteq (o, x^-)$. In this work, we opt for IQ-Learn [10] for both Eq. 5 and Eq. 6 to compute the gradients of θ and ϕ, respectively.

7 CONVERGENCE PROPERTIES

While the optimization of Eq. 4 will provide us with agent models whose occupancy measure is close to that of experts, it is not guaranteed that our practical, factored approach of iteratively minimizing each term of Eq. 4 will converge. Although IDIL provides theoretical analysis regarding the convergence of this factored distribution matching, their analysis is made under the assumption of full observability, thereby inapplicable to our setting. Thus, we provide a theoretical analysis regarding the convergence of DTIL in this section.

We start the analysis by defining two approximations of the expert oaxx-occupancy measure, $\check{\rho}_E^k$ and $\grave{\rho}_E^k$. These approximations are computed from the estimates of the expert's oax-occupancy and oxx-occupancy measures, i.e., $\rho_E^k(o, a, x)$ and $\rho_E^k(o, x, x^-)$, with the estimate of expert models $\mathcal{N}^k = (\pi^k, \zeta^k)$:

$$\check{\rho}_E^k(o_i, a_i, x_i, x_i^-) \doteq \rho_E^k(o_i, x_i, x_i^-)\pi_i^k(a_i|o_i, x_i)$$
$$\grave{\rho}_E^k(o_i, a_i, x_i, x_i^-) \doteq \rho_E^k(o_i, a_i, x_i)p(x_i^-|o_i, a_i, x_i, \mathcal{N}^k).$$



(a) ***Movers*** **(b)** ***Flood*** **(c)** ***Protoss-5v5*** **(d)** ***Terran-5v5***

Figure 2: Snapshopts of Experimental Domains

We can draw a relationship between the oax− occupancy measure matching and the oaxx− occupancy measure matching problems as follows:

Lemma 7.1. *Define $\Delta(\theta, \theta^k) \doteq \epsilon_1$ and $\Delta(\phi, \phi^k) \doteq \epsilon_2$. If π_θ is an K_1-Lipschitz function of θ, ζ_ϕ is an K_2-Lipschitz function of ϕ, and $\max(K_1, K_2)(|\epsilon_1| + |\epsilon_2|)$ is sufficiently small, then*

$$D_f\left(\rho_{\pi,\zeta}(o_i, a_i, x_i) \middle\| \rho_E^k(o_i, a_i, x_i)\right) = D_f\left(\rho_{\pi,\zeta}(o_i, a_i, x_i, x_i^-) \middle\| \grave{\rho}_E^k(o_i, a_i, x_i, x_i^-)\right)$$

The proof of Lemma 7.1 is based on the first-order approximation of f and provided in the Appendix. This implies that in reasonable conditions (e.g., smoothness of neural networks and compactness of parameter space), if we update θ only by a small amount via Eq. 5, our objective function, Eq. 3, also decreases.

Then, along with *Lemma 2.3* from [36], we can derive the following theorem for the convergence of DTIL.

Theorem 7.2. *Let $L^k \doteq D_f\left(\rho_i(o_i, a_i, x_i, x_i^-) \middle\| \rho_E^k(o_i, a_i, x_i, x_i^-)\right)$, and $p_E(o_i, a_i)$ denote the stationary distributions of o, a computed from the expert demonstrations D. If (1) the conditions of lemma 7.1 is satisfied and (2) $\check{\rho}_E^k \approx \grave{\rho}_E^k \approx p(x_i, x_i^- | o_i, a_i, \mathcal{N}_{1:n}^k) p_E(o_i, a_i)$, then $L^{k+1} \leq L^k$.*

The proof is built on the convexity of the f and the minimization of f-divergence. Its details are provided in the Appendix. With Thm. 7.2, since the objective function, L, is always positive, it will eventually converge to a local optimum. Note that without any information regarding the rules or labels of the subtasks, multiple solutions can exist to exhibit the expert demonstrations D. Our approach allows for semi-supervision by incorporating expert subtask labels in the E-step. As the experimental results demonstrate, semi-supervision can help disambiguate the models, finding one closer to the actual expert team behavior.

8 EXPERIMENTS

Through numerical experiments, we now assess DTIL's performance against MAIL baselines across multiple domains.

8.1 Experimental Setup

8.1.1 Domains. We evaluate DTIL across multiple domains with varying complexity, including the *Multi-Jobs-n* suite, *Movers*, *Flood*, and the *SMACv2* suite. The key characteristics of our experimental domains are presented in Table 1. Our domains include both continuous and discrete observation and action spaces, with varying numbers of agents (2-5) who are either subtask-agnostic or subtask-driven. Please refer to Figs. 3–1 for illustrations of *Multi-Jobs-n* domains. Remaining domains are illustrated in Fig. 2.

Table 1: Key Characteristics of Experimental Domains. "Subtask" refers to whether agents are subtask-driven. "Dim" denotes the dimension or cardinality of a space.

	Experts		**Observation Space**		**Action Space**	
Domain	# agents	Subtask	Type	*Dim*	Type	*Dim*
MJ-2	2	Yes	Cont.	6	Cont.	2
MJ-3	2	Yes	Cont.	6	Cont.	2
Movers	2	Yes	Disc.	45	Disc.	6
Flood	2	Yes	Disc.	56	Disc.	6
Protoss	5	No	Cont.	90	Disc.	11
Terran	5	No	Cont.	80	Disc.	11

The *Multi-Jobs-n* simulates the motivating example introduced in Fig. 1 in continuous observation and action spaces. *Movers* and *Flood* are collaborative team tasks in partially observable environments introduced by [33], considering only discrete states and actions. These domains are designed to admit multiple near-optimal strategies, allowing agents to exhibit multimodal behaviors. We create synthetic agents exhibiting hierarchical behavior and generate 50 and 100 demonstrations for training and testing, respectively, for each domain. *SMACv2* is a challenging benchmark for multi-agent reinforcement learning [9]. We consider two domains in this suite: *Protoss-5v5* and *Terran-5v5*, where a team of five agents is tasked with defeating five enemies. We obtain a multi-agent policy via MAPPO [45] and generate 50 trajectories per domain for training. In all domains, team members must make decentralized decisions in partially observable environments. For more details, please refer to the Appendix.

Baselines. We compare our approach with Behavior Cloning (BC), MA-GAIL (MG) [38], Independent-IQL (IIQL), and MA-OptionGAIL (MOG). BC is a supervised learning approach to learning policies, which serves as a fundamental baseline for imitation learning [22].

Table 2: Average cumulative task reward with multimodal team behavior. MOG-s and DTIL-s represent the results with 20% supervision.

Method	*MJ*-2	*MJ*-3	movers	rescue
Expert	24.1 ± 3.8	28.7 ± 4.6	-99 ± 32	4.6 ± 2.0
BC	9.6 ± 3.2	11.8 ± 1.5	-150 ± 0	0.0 ± 0.0
MG	6.9 ± 2.0	10.4 ± 1.2	-150 ± 0	4.5 ± 0.5
IIQL	14.7 ± 0.7	$\mathbf{27.8 \pm 1.6}$	$\mathbf{-107 \pm 9}$	$\mathbf{5.6 \pm 0.2}$
MOG	6.1 ± 1.3	7.4 ± 2.0	-150 ± 0	3.6 ± 0.6
DTIL	$\mathbf{16.8 \pm 5.9}$	27.0 ± 1.0	-108 ± 7	5.4 ± 0.4
BTIL	-	-	-150 ± 0	0.6 ± 0.4
MOG-s	11.8 ± 1.3	10.2 ± 2.0	-150 ± 0	3.8 ± 0.7
DTIL-s	$\mathbf{21.6 \pm 1.7}$	$\mathbf{27.3 \pm 1.3}$	$\mathbf{-99 \pm 14}$	$\mathbf{4.9 \pm 0.1}$

Table 3: Average cumulative task reward and the rate of wins in *SMACv2* domains.

Method	*Protoss*		*Terran*	
	Reward	Wins	Reward	Wins
Expert	18.0 ± 4.8	0.55 ± 0.50	11.9 ± 3.2	0.60 ± 0.49
BC	6.5 ± 0.1	0.17 ± 0.06	4.7 ± 1.8	0.07 ± 0.06
MG	7.8 ± 1.0	0.27 ± 0.06	7.2 ± 1.3	0.27 ± 0.6
IIQL	8.4 ± 0.7	$\mathbf{0.47 \pm 0.06}$	7.8 ± 0.7	$\mathbf{0.47 \pm 0.06}$
MOG	6.8 ± 1.0	0.13 ± 0.06	6.2 ± 1.3	0.23 ± 0.06
DTIL	$\mathbf{9.9 \pm 1.2}$	$\mathbf{0.47 \pm 0.06}$	$\mathbf{8.2 \pm 1.3}$	$\mathbf{0.47 \pm 0.06}$

MA-GAIL is a generative adversarial training-based MAIL algorithm, which employs the centralized training with decentralized execution (CTDE) approach [12]. Independent-IQL is a naive multi-agent extension of IQLearn [10], which applies IQLearn to each agent independently. Since this baseline also takes non-adversarial training, we can compare the effect of hierarchical structure in modeling team behavior with this baseline. To our knowledge, no approach exists for learning hierarchical policies in complex multi-agent domains. Thus, we present MA-OptionGAIL as a baseline, which learns a hierarchical policy of each agent separately via Option-GAIL [16]. For discrete domains, *Movers* and *Flood*, we also report the performance of BTIL [35].

Metrics. Similar to other imitation learning algorithms [10, 38], we use the cumulative task reward to evaluate the algorithms. In *SMACv2* domains, we also consider the win rate in battles. If the learned multi-agent policy aligns with the expert team behavior, it will achieve a task reward similar to the expert's. However, a high task reward does not necessarily indicate alignment with the expert team model, as the algorithms might learn only one optimal policy, resulting in unimodal rather than multimodal behavior. Therefore, we also measure the accuracy of subtask inference. The closer the learned model is to the expert model, the more accurately it can predict expert subtasks from their demonstrations. We use the MAP estimation (the E-step of Alg. 1) for subtask inference.

Table 4: Accuracy of Subtask Inference. We represents the 1-st agent of the *MJ*-2 domain as *MJ*-2-1, and similarly for other agents.

Agent	Random	BTIL	MOG-s	DTIL-s
MJ-2-1	≈ 0.50	-	0.61 ± 0.09	$\mathbf{0.75 \pm 0.04}$
MJ-2-2	≈ 0.50	-	0.63 ± 0.17	$\mathbf{0.75 \pm 0.07}$
MJ-3-1	≈ 0.33	-	0.49 ± 0.04	$\mathbf{0.78 \pm 0.07}$
MJ-3-2	≈ 0.33	-	0.68 ± 0.06	$\mathbf{0.72 \pm 0.08}$
Movers-1	≈ 0.25	$\mathbf{0.90 \pm 0.01}$	0.35 ± 0.15	0.78 ± 0.02
Movers-2	≈ 0.25	$\mathbf{0.91 \pm 0.01}$	0.46 ± 0.07	0.78 ± 0.07
Flood-1	≈ 0.25	0.53 ± 0.04	0.31 ± 0.06	$\mathbf{0.61 \pm 0.08}$
Flood-2	≈ 0.25	$\mathbf{0.62 \pm 0.01}$	0.25 ± 0.12	0.57 ± 0.03

8.2 Results

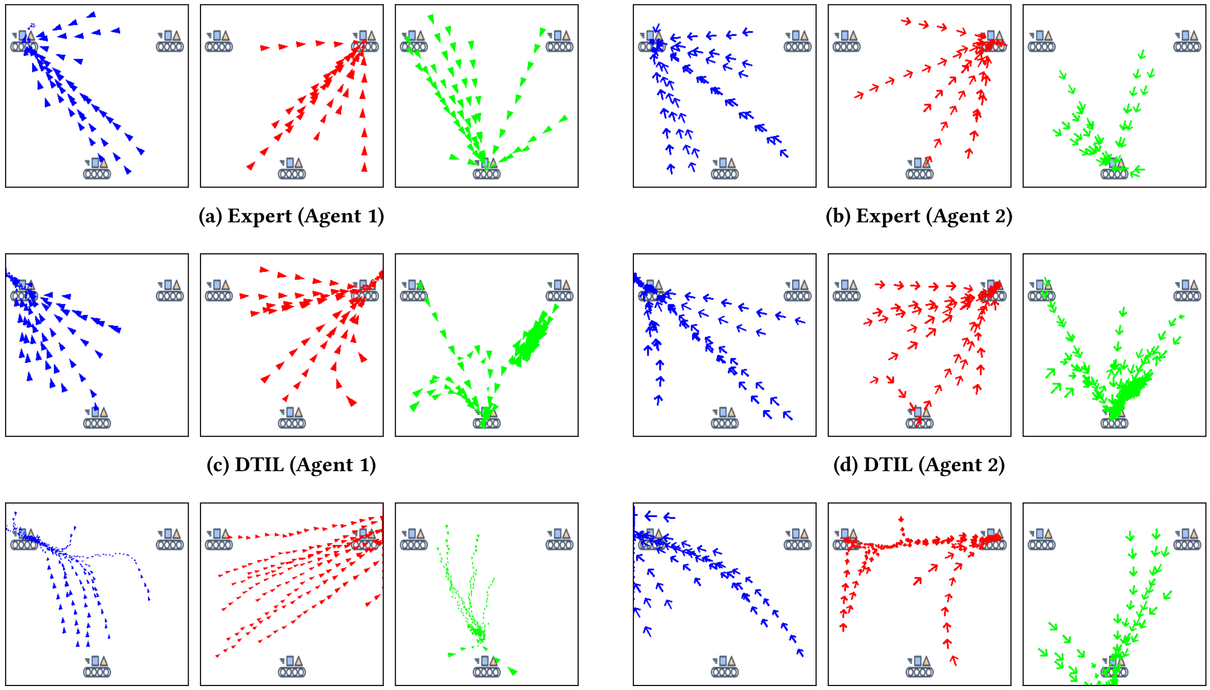
8.2.1 DTIL achieves expert-level task performance. Table 2 shows the task rewards averaged over three trials for *Multi-Jobs-n* (*MJ-n*), *Movers*, and *Flood*. We observe that IQLearn-based approaches, IIQL and DTIL, generally perform better than approaches based on generative adversarial imitation learning. Between MG and MOG, MG performed better, likely because MG additionally utilizes other agents' information during training. In contrast, MOG and DTIL take only individual observation-action trajectories and do not utilize any other information that might break the partial observability condition even during the training phase.

While DTIL outperformed IIQL in *Multi-Jobs*-2, it performed on par in other domains. We believe this is due to the domains being too simple, allowing subtask-agnostic approaches to extract one optimal solution from demonstrations. In more complex domains, we could observe an improvement in task performance due to the hierarchical structure of our agent model. As shown in Table 3, DTIL achieved the highest task reward and win rate in both the *SMACv2* domains: *Protoss-5v5* and *Terran-5v5*. We want to highlight that even though IIQL often achieves high task rewards, it can neither learn multimodal behavior nor utilize semi-supervision. On the other hand, DTIL can improve its performance by augmenting subtask labels. As shown in Table 2, DTIL achieved a task reward similar to the expert task reward in all domains with 20% supervision.

8.2.2 DTIL accurately learns multimodal team behavior. As mentioned in Sec. 4, our goal is to learn the different team behaviors generated by expert teams rather than a unimodal team policy. Additionally, in human-AI teaming applications, an AI agent must accurately interpret its human teammate's high-level plan. To achieve this, it is essential to learn a model that exhibits hierarchical behavior aligned with expert team members. Table. 4 presents the accuracy of subtask inference computed with 20%-supervision models. Note that without any supervision, we cannot associate the learned latent values with the actual subtasks. In all cases, DTIL outperforms MOG and the random baselines.[2]

8.2.3 DTIL outperforms BTIL in more complex tasks. As demonstrated in Table 2, DTIL outperformed BTIL in terms of task reward.

[2]Note that other DNN-based baselines cannot be utilized for subtask inference.



(a) Expert (Agent 1) (b) Expert (Agent 2)

(c) DTIL (Agent 1) (d) DTIL (Agent 2)

(e) MA-OptionGAIL (Agent 1) (f) MA-OptionGAIL (Agent 2)

Figure 3: Visualization of individual *Multi-Jobs*-3 trajectories generated by the expert and learned models conditioned on a fixed subtask. The directions of the triangles and arrows represent the actions of agents at each position. The three colors represent the three fixed subtasks. Both learned models (DTIL and MA-OptionGAIL) are trained with 20 % supervision of subtask labels.

While BTIL's overall performance was generally below that of online methods such as MA-OptionGAIL and DTIL, it performed slightly better than BC. We attribute this to BTIL's offline nature, which makes it prone to compounding errors. This implies that if a BTIL agent encounters a state that was not present in the training dataset, it struggles to select the appropriate action, as it has not learned anything about that state. On the other hand, BTIL's subtask inference performance was on par with, or even superior to, DTIL. As shown in Table 4, BTIL achieved approximately 0.9 accuracy in subtask inference for *Movers*. However, we emphasize that this level of performance is only feasible in small domains, as BTIL cannot scale up to domains with larger state spaces.

8.2.4 Team models learned using DTIL generated behaviors that are qualitatively similar to those generated by expert teams. To interpret the learned behavior associated with each subtask (x), we visualized the paths generated by ten different seeds (seed=0:9) for each model in Figure 3. For this visualization, we intentionally set the part of an agent's observation related to other agents to zero, eliminating their influence on the behavior of the agent being inspected. Figure 3 shows that DTIL's subtask-driven behavior closely resembles the expert's when trained with partial supervision of subtask labels. In contrast, MA-OptionGAIL trajectories tend to be noisy and unfocused on a specific subgoal, even with a fixed x. We believe the superior performance of DTIL stems from its factored structure, in which it learns separate Q-functions for π and ζ. Given that Q-functions can be interpreted as reward functions [10], our approach effectively learns a hierarchical reward corresponding to each level of the policy. However, since MA-OptionGAIL does not learn separate Q-functions, it is difficult to determine whether it truly captures a hierarchical policy structure or merely optimizes the joint policy $\pi(a, x|o, x^{-})$.

9 CONCLUSION

This work introduces DTIL, an algorithm for learning generative models of team behavior from heterogeneous demonstrations. Experiments show that DTIL outperforms state-of-the-art multi-agent imitation learning baselines and captures expert team behavior across six diverse teamwork domains. Additionally, DTIL can generate a wide range of expert team behaviors. DTIL also motivates future research directions. First, DTIL assumes a known, finite set of subtasks, though real-world subtasks may be difficult to define a prior or represent as scalars. Future MAIL methods should explore more expressive hierarchical representations. Second, by enabling generative models of team behavior, DTIL can enable novel human-AI teaming applications, such as AI-enabled team coaching [32, 34] and end-user programming of multi-agent systems [2, 39]. We invite developers of these and other impactful applications to utilize DTIL and make additional details available at http://tiny.cc/dtil-appendix

ACKNOWLEDGMENTS

This research was supported in part by NSF award #2205454, ARO CA# W911NF-20-2-0214, and Rice University funds.

REFERENCES

[1] Pieter Abbeel and Andrew Y Ng. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*. 1.

[2] Gopika Ajaykumar, Maureen Steele, and Chien-Ming Huang. 2021. A survey on end-user robot programming. *ACM Computing Surveys (CSUR)* 54, 8 (2021), 1–36.

[3] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565* (2016).

[4] Saurabh Arora and Prashant Doshi. 2021. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence* (2021), 103500.

[5] Raunak P Bhattacharyya, Derek J Phillips, Changliu Liu, Jayesh K Gupta, Katherine Driggs-Campbell, and Mykel J Kochenderfer. 2019. Simulating emergent properties of human driving behavior using multi-agent reward augmented imitation learning. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 789–795.

[6] Raunak P Bhattacharyya, Derek J Phillips, Blake Wulfe, Jeremy Morton, Alex Kuefler, and Mykel J Kochenderfer. 2018. Multi-agent imitation learning for driving simulation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 1534–1539.

[7] Jiayu Chen, Dipesh Tamboli, Tian Lan, and Vaneet Aggarwal. 2023. Multi-task hierarchical adversarial inverse reinforcement learning. In *International Conference on Machine Learning*. PMLR, 4895–4920.

[8] Sonia Chernova and Andrea L Thomaz. 2022. *Robot learning from human teachers*. Springer Nature.

[9] Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob Nicolaus Foerster, and Shimon Whiteson. 2023. SMACv2: An Improved Benchmark for Cooperative Multi-Agent Reinforcement Learning. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. https://openreview.net/forum?id=5OjLGiJW3u

[10] Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. 2021. Iq-learn: Inverse soft-q learning for imitation. *Advances in Neural Information Processing Systems* 34 (2021), 4028–4039.

[11] Seyed Kamyar Seyed Ghasemipour, Richard Zemel, and Shixiang Gu. 2020. A divergence minimization perspective on imitation learning methods. In *Conference on Robot Learning*. PMLR, 1259–1277.

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. 2672–2680.

[13] Karol Hausman, Yevgen Chebotar, Stefan Schaal, Gaurav Sukhatme, and Joseph J Lim. 2017. Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets. *Advances in neural information processing systems* 30 (2017).

[14] Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning. *Advances in neural information processing systems* 29 (2016).

[15] Abhinav Jain and Vaibhav Unhelkar. 2024. GO-DICE: Goal-conditioned Option-aware Offline Imitation Learning via Stationary Distribution Correction Estimation. In *AAAI Conference on Artificial Intelligence*.

[16] Mingxuan Jing, Wenbing Huang, Fuchun Sun, Xiaojian Ma, Tao Kong, Chuang Gan, and Lei Li. 2021. Adversarial option-aware hierarchical imitation learning. In *International Conference on Machine Learning*. PMLR, 5097–5106.

[17] Thomas Kipf, Yujia Li, Hanjun Dai, Vinicius Zambaldi, Alvaro Sanchez-Gonzalez, Edward Grefenstette, Pushmeet Kohli, and Peter Battaglia. 2019. Compile: Compositional imitation learning and execution. In *International Conference on Machine Learning*. PMLR, 3418–3428.

[18] Hoang Le, Nan Jiang, Alekh Agarwal, Miroslav Dudík, Yisong Yue, and Hal Daumé III. 2018. Hierarchical imitation and reinforcement learning. In *International conference on machine learning*. PMLR, 2917–2926.

[19] Hoang M Le, Yisong Yue, Peter Carr, and Patrick Lucey. 2017. Coordinated multi-agent imitation learning. In *International Conference on Machine Learning*. PMLR, 1995–2003.

[20] Sang-Hyun Lee and Seung-Woo Seo. 2020. Learning compound tasks without task-specific knowledge via imitation and self-supervised learning. In *International Conference on Machine Learning*. PMLR, 5747–5756.

[21] Yunzhu Li, Jiaming Song, and Stefano Ermon. 2017. Infogail: Interpretable imitation learning from visual demonstrations. *Advances in neural information processing systems* 30 (2017).

[22] Ziniu Li, Tian Xu, Yang Yu, and Zhi-Quan Luo. 2022. Rethinking ValueDice: Does it really improve performance? *arXiv preprint arXiv:2202.02468* (2022).

[23] Xiaomin Lin, Stephen C Adams, and Peter A Beling. 2019. Multi-agent inverse reinforcement learning for certain general-sum stochastic games. *Journal of Artificial Intelligence Research* 66 (2019), 473–502.

[24] Minghuan Liu, Ming Zhou, Weinan Zhang, Yuzheng Zhuang, Jun Wang, Wulong Liu, and Yong Yu. 2020. Multi-agent interactions modeling with correlated policies. *arXiv preprint arXiv:2001.03415* (2020).

[25] Frans A Oliehoek and Christopher Amato. 2016. *A concise introduction to decentralized POMDPs*. Vol. 1. Springer.

[26] Liubove Orlov-Savko, Abhinav Jain, Gregory M Gremillion, Catherine E Neubauer, Jonroy D Canady, and Vaibhav Unhelkar. 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.

[27] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, and Jan Peters. 2018. An algorithmic perspective on imitation learning. *Foundations and Trends in Robotics* 7, 1-2 (2018), 1–179.

[28] Martin L Puterman. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

[29] Denise L Reyes and Eduardo Salas. 2019. What makes a team of experts an expert team? (2019).

[30] Edward Schmerling, Karen Leung, Wolf Vollprecht, and Marco Pavone. 2018. Multimodal probabilistic model-based planning for human-robot interaction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 3399–3406.

[31] Prasanth Sengadu Suresh, Yikang Gui, and Prashant Doshi. 2023. Dec-AIRL: Decentralized Adversarial IRL for Human-Robot Teaming. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 1116–1124.

[32] Sangwon Seo, Bing Han, Rayan E. Harari, Roger D. Dias, Marco A. Zenati, Eduardo Salas, and Vaibhav Unhelkar. 2025. Socratic: Enhancing Human Teamwork via AI-enabled Coaching. In *Proceedings of the 24th International Conference on Autonomous Agents and Multiagent Systems*.

[33] Sangwon Seo, Bing Han, and Vaibhav Unhelkar. 2023. Automated Task-Time Interventions to Improve Teamwork using Imitation Learning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 335–344.

[34] Sangwon Seo, Lauren R. Kennedy-Metz, Marco A. Zenati, Julie A. Shah, Roger D. Dias, and Vaibhav V. Unhelkar. 2021. Towards an AI Coach to Infer Team Mental Model Alignment in Healthcare. In *2021 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*. 39–44. https://doi.org/10.1109/CogSIMA51574.2021.9475925

[35] Sangwon Seo and Vaibhav Unhelkar. 2022. Semi-Supervised Imitation Learning of Team Policies from Suboptimal Demonstrations. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

[36] Sangwon Seo and Vaibhav Unhelkar. 2024. IDIL: Imitation Learning of Intent-Driven Expert Behavior. In *Proceedings of the 2024 International Conference on Autonomous Agents and Multiagent Systems*.

[37] Arjun Sharma, Mohit Sharma, Nicholas Rhinehart, and Kris M Kitani. 2018. Directed-info gail: Learning hierarchical policies from unsegmented demonstrations using directed information. *arXiv preprint arXiv:1810.01266* (2018).

[38] Jiaming Song, Hongyu Ren, Dorsa Sadigh, and Stefano Ermon. 2018. Multi-agent generative adversarial imitation learning. *Advances in neural information processing systems* 31 (2018).

[39] Laura Stegner, Yuna Hwang, David Porfirio, and Bilge Mutlu. 2024. Understanding On-the-Fly End-User Robot Programming. In *Proceedings of the 2024 ACM Designing Interactive Systems Conference*. 2468–2480.

[40] Richard S Sutton. 2018. Reinforcement learning: An introduction. *A Bradford Book* (2018).

[41] Richard S Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence* 112, 1-2 (1999), 181–211.

[42] Vaibhav V Unhelkar and Julie A Shah. 2019. Learning models of sequential decision-making with partial specification of agent behavior. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 2522–2530.

[43] Chen Wang, Claudia Pérez-D'Arpino, Danfei Xu, Li Fei-Fei, Karen Liu, and Silvio Savarese. 2022. Co-gail: Learning diverse strategies for human-robot collaboration. In *Conference on Robot Learning*. PMLR, 1279–1290.

[44] Fan Yang, Alina Vereshchaka, Changyou Chen, and Wen Dong. 2020. Bayesian Multi-type Mean Field Multi-agent Imitation Learning. *Advances in Neural Information Processing Systems* 33 (2020).

[45] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. 2022. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

[46] Lantao Yu, Jiaming Song, and Stefano Ermon. 2019. Multi-agent adversarial inverse reinforcement learning. In *International Conference on Machine Learning*. PMLR, 7194–7201.

[47] Zhiyu Zhang and Ioannis Paschalidis. 2021. Provable hierarchical imitation learning via em. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 883–891.