Multi-Agent Generative Adversarial Imitation Learning

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Imitation learning algorithms can be used to learn a policy from expert demonstrations without access to a reward signal. However, most existing approaches are not applicable in multi-agent settings due to the existence of multiple (Nash) equilibria and non-stationary environments. We propose a new framework for multi-agent imitation learning for general Markov games, where we build upon a generalized notion of inverse reinforcement learning. We further introduce a practical multi-agent actor-critic algorithm with good empirical performance. Our method can be used to imitate complex behaviors in high-dimensional environments with multiple cooperative or competing agents.

1 Introduction

Reinforcement learning (RL) methods are becoming increasingly successful at optimizing reward signals in complex, high dimensional environments [1]. A key limitation of RL, however, is the difficulty of designing suitable reward functions for complex and not well-specified tasks [2, 3]. If the reward function does not cover all important aspects of the task, the agent could easily learn undesirable behaviors [4]. This problem is further exacerbated in multi-agent scenarios, such as multiplayer games [5], multi-robot control [6] and social interactions [7]; in these cases, agents do not even necessarily share the same reward function.

Imitation learning methods address these problems via expert demonstrations [8–11]; the agent directly learns desirable behaviors by imitating an expert. Notably, inverse reinforcement learning (IRL, [12, 13]) frameworks assume that the expert is (approximately) optimal under some reward function and recovers one that rationalizes the expert behavior; an agent policy is subsequently learned through RL. Unfortunately, this paradigm is not suitable for general multi-agent settings due to environment being non-stationary to individual agents [14] and multiple equilibrium solutions [15].

In this paper, we propose a new framework for multi-agent imitation learning for general Markov games. We will integrate multi-agent RL with a suitable extension of multi-agent inverse RL. The resulting procedure strictly generalizes Generative Adversarial Imitation Learning (GAIL) [16] in the single agent case. Learning corresponds to a two-player game between a generator and a discriminator. The generator controls the policies of agents in a distributive fashion, and the discriminator contains a classifier for each agent that is trained to distinguish that agent's behavior from that of the corresponding expert. We can incorporate prior knowledge into the discriminators, including the presence of cooperative or competitive agents. In addition, we propose a novel multi-agent natural policy gradient algorithm that addresses the high variance issue [14, 17]. Empirical results demonstrate that our method can imitate complex behaviors in high-dimensional environments with multiple cooperative or competitive agents.

2 Preliminaries

2.1 Markov games

We consider an extension of Markov decision processes (MDPs) called Markov games [18]. A Markov game (MG) for N agents is defined via a set of states $\mathcal{S}$, N sets of actions $\{\mathcal{A}_i\}_{i=1}^N$. The function $T : \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_N \to \mathcal{P}(\mathcal{S})$ describes the transition between states, where $\mathcal{P}(\mathcal{S})$ denotes the set of probability distributions over the set $\mathcal{S}$. Each agent i obtains a reward given by a function $r_i : \mathcal{S} \times \mathcal{A}_1 \times \cdots \times \mathcal{A}_N \to \mathbb{R}$. Each agent i aims to maximize its own total expected return $R_i = \sum_{t=0}^T \gamma^t r_{i,t}$, where γ is the discount factor and T is the time horizon, by selecting actions through a (stationary and Markovian) stochastic policy $\pi_i : \mathcal{S} \times \mathcal{A}_i \to [0, 1]$. The initial states are determined by a distribution $\eta : \mathcal{S} \to [0, 1]$. The joint policy is defined as $\pi(a|s) = \prod_{i=1}^N \pi_i(a_i|s)$, where we use variables without subscript i to denote the concatenation of all variables for all agents (e.g. π denotes the joint policy $\prod_{i=1}^N \pi_i$ in a multi-agent setting, r denotes all rewards, a denotes actions of all agents) and use expectation with respect to a policy π to denote an expectation with respect to the trajectories it generates (e.g. $\mathbb{E}_\pi[r(s, a)] \triangleq \mathbb{E}[\sum_{t=0}^\infty \gamma^t r(s_t, a_t)]$, where $s_0 \sim \eta$, $a_t \sim \pi(a_t|s_t)$, $s_{t+1} \sim T(s_{t+1}|a_t, s_t)$). We use subscript $-i$ to denote *all agents except for* i.

2.2 Reinforcement learning and Nash equilibrium

In reinforcement learning (RL), the goal of each agent is to maximize total expected return $\mathbb{E}_\pi[r(s, a)]$ given access to the reward signal r. In single agent RL, an optimal Markovian policy exists but the optimal policy might not be unique (e.g., for an identically zero reward). An entropy regularizer can be introduced to resolve this ambiguity. The optimal policy is found via the following RL procedure:

$$\text{RL}(r) = \arg\max_{\pi \in \Pi} H(\pi) + \mathbb{E}_\pi[r(s, a)], \tag{1}$$

where $H(\pi) \triangleq \mathbb{E}_\pi[-\pi(a|s)]$ is the γ-discounted causal entropy [19] of policy $\pi \in \Pi$.

In Markov games, however, the optimal policy of an agent depends on other agents' policies. One approach is to use an equilibrium solution concept, such as Nash equilibrium [15]. Informally, a set of policies $\{\pi_i\}_{i=1}^N$ is a Nash equilibrium if no agent can achieve higher reward by unilaterally changing its policy, i.e. $\forall i \in [1, N], \forall \hat{\pi}_i \neq \pi_i, \mathbb{E}_{\pi_i,\pi_{-i}}[r_i] \geq \mathbb{E}_{\hat{\pi}_i,\pi_{-i}}[r_i]$. The process of finding a Nash equilibrium can be defined as a constrained optimization problem [20]:

$$\min_{\pi \in \Pi, v \in \mathbb{R}^{\mathcal{S} \times N}} f_r(\pi, v) = \sum_{i=1}^N \Big(\sum_{s \in \mathcal{S}} v_i(s) - \mathbb{E}_{a_i \sim \pi_i(\cdot|s)} q_i(s, a_i)\Big) \tag{2}$$

$$v_i(s) \geq q_i(s, a_i) \equiv \mathbb{E}_{\pi_{-i}}[r_i(s, a) + \gamma \sum_{s' \in \mathcal{S}} T(s'|s, a) v_i(s')] \quad \forall i \in [1, N], s \in \mathcal{S}, a_i \in \mathcal{A}_i \tag{3}$$

where the joint action a includes actions a_{-i} sampled from π_{-i} and a_i. The constraints enforce the Nash equilibrium condition – when the constraints are satisfied, every $(v_i(s) - q_i(s, a_i))$ is non-negative. Hence $f_r(\pi, v)$ is always non-negative for a feasible (π, v). Moreover, this objective has a global minimum of zero, and π forms a Nash equilibrium if and only if $f_r(\pi, v)$ reaches zero while being a feasible solution ([21], Theorem 2.4).

2.3 Inverse reinforcement learning

Suppose we do not have access to the reward signal r, but have demonstrations $\mathcal{D}$ provided by an expert (N experts in Markov games). Imitation learning aims to learn policies that behave similarly to these demonstrations. In Markov games, we assume all experts/players operate in the same environment, and the trajectories $\mathcal{D} = \{(s_j, a_j)\}_{j=1}^M$ are collected by sampling $s_0 \sim \eta(s), a_t = \pi_E(a_t|s_t), s_{t+1} \sim T(s_{t+1}|s_t, a_t)$. We assume that once we obtain $\mathcal{D}$, we cannot ask for additional expert interactions with the environment (unlike in DAgger [22] or CIRL [23]).

We consider the framework of singe-agent Maximum Entropy IRL ([8, 16]) to recover a reward function r that rationalizes the expert behavior π_E:

$$\text{IRL}(\pi_E) = \arg\max_{r \in \mathbb{R}^{\mathcal{S} \times \mathcal{A}}} \mathbb{E}_{\pi_E}[r(s, a)] - \left(\max_{\pi \in \Pi} H(\pi) + \mathbb{E}_\pi[r(s, a)]\right)$$

In practice, expectations with respect to π_E are evaluated using samples from $\mathcal{D}$.

The IRL objective is ill-defined [12, 10] and there are often multiple valid solutions to the problem when we consider all $r \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$. To resolve this ambiguity, Ho et al. introduce a convex reward function regularizer $\psi : \mathbb{R}^{\mathcal{S}\times\mathcal{A}} \to \mathbb{R}$, which can be used to restrict rewards to be linear in a pre-determined set of features [16]:

$$\mathrm{IRL}_{\psi}(\pi_E) = \operatorname*{arg\,max}_{r\in\mathbb{R}^{\mathcal{S}\times\mathcal{A}}} \psi(r) + \mathbb{E}_{\pi_E}[r(s,a)] - \left(\max_{\pi\in\Pi} H(\pi) + \mathbb{E}_{\pi}[r(s,a)]\right) \tag{4}$$

2.4 Imitation by matching occupancy measures

[16] interpret the imitation learning problem as matching two occupancy measures, i.e., the distribution over states and actions encountered when navigating the environment with a policy. Formally, for a policy π, it is defined as $\rho_\pi(s,a) = \pi(a|s)\sum_{t=0}^{\infty}\gamma^t T(s_t = s|\pi)$. [16] draw a connection between IRL and occupancy measure matching, showing that the former is a dual of the latter:

Proposition 1 (Proposition 3.1 in [16]). $\mathrm{RL}\circ\mathrm{IRL}_{\psi}(\pi_E) = \arg\min_{\pi\in\Pi} -H(\pi) + \psi^\star(\rho_\pi - \rho_{\pi_E})$.

Here $\psi^\star(x) = \sup_y x^\top y - \psi(y)$ is the convex conjugate of ψ, which could be interpreted as a measure of similarity between the occupancy measures of expert policy and agent's policy. One instance of $\psi = \psi_{\mathrm{GA}}$ gives rise to the Generative Adversarial Imitation Learning (GAIL) method:

$$\psi^\star_{\mathrm{GA}}(\rho_\pi - \rho_{\pi_E}) = \max_{D\in(0,1)^{\mathcal{S}\times\mathcal{A}}} \mathbb{E}_{\pi_E}[\log(D(s,a))] + \mathbb{E}_{\pi}[\log(1 - D(s,a))] \tag{5}$$

The resulting imitation learning method from Proposition 1 involves a *discriminator* (a classifier D) competing with a *generator* (a policy π). The discriminator attempts to distinguish real vs. synthetic trajectories (produced by π) by optimizing (5). The generator, on the other hand, aims to perform optimally under that reward function, thus "fooling" the discriminator with synthetic trajectories that are difficult to distinguish from the expert ones.

3 Generalizing IRL to Markov games

Extending imitation learning to multi-agent settings is difficult because there are multiple rewards (one for each agent) and the notion of optimality is complicated by the need to consider an equilibrium solution [15]. We use $\mathrm{MARL}(r)$ to denote the set of (stationary and Markovian) policies that form a Nash equilibrium under r and have the minimum γ-discounted causal entropy (among all equilibria):

$$\mathrm{MARL}(r) = \operatorname*{arg\,min}_{\pi\in\Pi, v\in\mathbb{R}^{\mathcal{S}\times N}} f_r(\pi, v) - H(\pi) \tag{6}$$
$$v_i(s) \geq q_i(s, a_i) \quad \forall i \in [1, N], s \in \mathcal{S}, a_i \in \mathcal{A}_i$$

where q is defined as in Eq. 3. Our goal is to define a suitable inverse operator MAIRL, in analogy to IRL in Eq. 4. The key idea of Eq. 4 is to choose a reward that creates a *margin* between the expert and every other policy. However, the *constraints* in the Nash equilibrium optimization (Eq. 6) can make this challenging. Instead we derive an equivalent Lagrangian formulation of (6), where we move the constraints into the objective function.

3.1 Equivalent constraints via temporal difference learning

Intuitively, the Nash equilibrium constraints imply that any agent i cannot improve π_i via 1-step temporal difference learning. Based on this notion, we can derive equivalent versions of the constraints corresponding to t-step temporal difference (TD) learning.

Theorem 1. *Let $\hat{v}_i(s;\pi,r)$ be the solution to the Bellman equation*

$$\hat{v}_i(s;\pi,r) = \mathbb{E}_\pi[r_i(s,a) + \gamma\sum_{s'\in\mathcal{S}} T(s'|s,a)\hat{v}_i(s')]$$

Denote $\hat{q}_i^{(t)}(\{s^{(j)}, a^{(j)}\}_{j=0}^{t-1}, s^{(t)}, a_i^{(t)})$ as the discounted expected return for the i-th agent conditioned on visiting the trajectory $\{s^{(j)}, a^{(j)}\}_{j=0}^{t-1}, s^{(t)}$ in the first $t-1$ steps and choosing action $a_i^{(t)}$ at the t

step, when other agents use policy π_{-i}:

$$\hat{q}_i^{(t)}(\{s^{(j)}, a^{(j)}\}_{j=0}^{t-1}, s^{(t)}, a_i^{(t)}) = \sum_{j=0}^{t-1} \gamma^j r_i(s^{(j)}, a^{(j)}) + \gamma^t \mathbb{E}_{\pi_{-i}}[r_i(s^{(t)}, a^{(t)}) + \gamma \sum_{s' \in \mathcal{S}} T(s'|s, a^{(t)}) \hat{v}_i(s')]$$

where $\hat{v}_i(s)$ is short for $\hat{v}_i(s; \pi, r)$. Then π is Nash equilibrium if and only if

$$\hat{v}_i(s^{(0)}) \geq \mathbb{E}_{\pi_{-i}}[\hat{q}_i^{(t)}(\{s^{(j)}, a^{(j)}\}_{j=0}^{t-1}, s^{(t)}, a_i^{(t)})] \triangleq Q_i^{(t)}(\{s^{(j)}, a_i^{(j)}\}_{j=0}^{t})$$
$$\forall t \in \mathbb{N}^+, i \in [N], j \in [0, t], s^{(j)} \in \mathcal{S}, a^{(j)} \in \mathcal{A}$$

Intuitively, Theorem 1 states that if we replace the 1-step constraints with $(t+1)$-step constraints, we obtain the same solution as MARL(r), since $(t+1)$-step TD updates (over one agent at a time) is still stationary with respect to a Nash equilibrium solution. So the constraints can be unrolled for t steps and rewritten as $\hat{v}_i(s^{(0)}) \geq Q_i^{(t)}(\{s^{(j)}, a_i^{(j)}\}_{j=0}^{t})$.

3.2 Multi-agent inverse reinforcement learning

We are now ready to construct the Lagrangian dual of the primal in Equation 6, using the equivalent formulation from Theorem 1. The first observation is that for any policy π, $f(\pi, \hat{v}) = 0$ when $\hat{v}$ is defined as in Theorem 1 (see Lemma 1 in appendix). Therefore, we only need to consider the "unrolled" constraints from Theorem 1, obtaining the following dual problem

$$\min_{\pi} \max_{\lambda \geq 0} L_r^{(t+1)}(\pi, \lambda) = \sum_{i=1}^{N} \sum_{\tau_i \in \mathcal{T}_i^t} \lambda(\tau_i)(Q_i^{(t)}(\tau_i) - \hat{v}_i(s^{(0)})) \tag{7}$$

where $\mathcal{T}_i(t)$ is the set of all length-t trajectories of the form $\{s^{(j)}, a_i^{(j)}\}_{j=0}^{t}$, with $s^{(0)}$ as initial state, λ is a vector of $N \cdot |\mathcal{T}_i(t)|$ Lagrange multipliers, and $\hat{v}$ is defined as in Theorem 1.

In the following theorem, we show that for a specific choice of λ we can recover the difference of the sum of expected rewards between two policies, a performance gap similar to the one used in single agent IRL in Eq. (4). This amounts to "relaxing" the primal problem.

Theorem 2. *For any two policies $\pi^\star$ and π, let*

$$\lambda_\pi^\star(\tau_i) = \eta(s^{(0)})\pi_i(a_i^{(0)}|s^{(0)}) \prod_{j=1}^{t} \pi_i(a_i^{(j)}|s^{(j)}) \sum_{a_{-i}^{(j-1)}} T(s^{(j)}|s^{(j-1)}, a^{(j-1)})\pi_{-i}^\star(a_{-i}^{(j)}|s^{(j)})$$

be the probability of generating the sequence τ_i using policy π_i and $\pi_{-i}^\star$. Then

$$\lim_{t \to \infty} L_r^{(t+1)}(\pi^\star, \lambda_\pi^\star) = \sum_{i=1}^{N} \mathbb{E}_{\pi_i, \pi_{-i}^\star}[r_i(s, a)] - \sum_{i=1}^{N} \mathbb{E}_{\pi_i^\star, \pi_{-i}^\star}[r_i(s, a)] \tag{8}$$

We provide a proof in Appendix A.3. Intuitively, the $\lambda^\star(\tau_i)$ weights correspond to the probability of generating trajectory τ_i when the policy is π_i for agent i and $\pi_{-i}^\star$ for the other agents. As $t \to \infty$, this amounts to the expected total reward, and is equivalent to computing the first term $\mathbb{E}_{\pi_i, \pi_{-i}^\star}[r_i]$. The marginal of $\lambda^\star$ over the initial states is the initial state distribution, so $\sum_s \hat{v}(s)\eta(s) = \mathbb{E}_{\pi_i^\star, \pi_{-i}^\star}[r_i]$ gives the second term. Theorem 2 motivates the following definition of multi-agent IRL with regularizer ψ.

$$\text{MAIRL}_\psi(\pi_E) = \arg\max_r \psi(r) + \sum_{i=1}^{N}(\mathbb{E}_{\pi_E}[r_i]) - \left(\max_\pi \sum_{i=1}^{N}(\beta H_i(\pi_i) + \mathbb{E}_{\pi_i, \pi_{E_{-i}}}[r_i])\right), \tag{9}$$

where $H_i(\pi_i) = \mathbb{E}_{\pi_i, \pi_{E_{-i}}}[-\pi_i(a_i|s)]$ is the discounted causal entropy for policy π_i when other agents follow $\pi_{E_{-i}}$, and β is a hyper-parameter controlling the strength of the entropy regularization term as in [16]. This formulation is a strict generalization to the single agent IRL in [16].

Corollary 2.1. *If $N = 1$, $\beta = 1$ then MAIRL$_\psi(\pi_E)$ = IRL$_\psi(\pi_E)$.*

Furthermore, if the regularization ψ is additively separable, and for each agent i, π_{E_i} is the unique optimal response to other experts $\pi_{E_{-i}}$, we obtain the following:

Theorem 3. *Assume that $\psi(r) = \sum_{i=1}^N \psi_i(r_i)$, and that MARL$(r)$ has a unique solution*[1] *for all $r \in$ MAIRL$_\psi(\pi_E)$, then*

$$\textit{MARL} \circ \textit{MAIRL}_\psi(\pi_E) = \arg\min_{\pi \in \Pi} \sum_{i=1}^N -\beta H_i(\pi_i) + \psi_i^\star(\rho_{\pi_{i,E_{-i}}} - \rho_{\pi_E})$$

where $\pi_{i,E_{-i}}$ denotes π_i for agent i and $\pi_{E_{-i}}$ for other agents.

The above theorem suggests that ψ-regularized multi-agent inverse reinforcement learning is seeking, for each agent i, a policy whose occupancy measure is close to one where we replace policy π_i with expert π_{E_i}, as measured by the convex function $\psi_i^\star$.

However, we do not assume access to the expert policy π_E during training, so it is not possible to obtain $\rho_{\pi_{i,E_{-i}}}$. In the settings of this paper, we consider an alternative approach where we match the occupancy measure between ρ_{π_E} and ρ_π instead. We can obtain our practical algorithm if we select an adversarial reward function regularizer and remove the effect from entropy regularizers.

Proposition 2. *If $\beta = 0$, and $\psi(r) = \sum_{i=1}^N \psi_i(r_i)$ where $\psi_i(r_i) = \mathbb{E}_{\pi_E}[g(r_i)]$ if $r_i > 0$; $+\infty$ otherwise, and $g(x) = -x - \log(1 - e^x)$ if $x < 0$; $+\infty$ otherwise, then*

$$\arg\min_\pi \sum_{i=1}^N \psi_i^\star(\rho_{\pi_i, \pi_{E_{-i}}} - \rho_{\pi_E}) = \arg\min_\pi \sum_{i=1}^N \psi_i^\star(\rho_{\pi_i, \pi_{-i}} - \rho_{\pi_E}) = \pi_E$$

4 Practical multi-agent imitation learning

Despite the recent successes in deep RL, RL algorithms are notoriously hard to train because of high variance gradient estimates. This is further exacerbated in Markov games since an agent's optimal policy depends on other agents [14, 17]. In this section, we address these problems and propose practical algorithms for multi-agent imitation.

4.1 Multi-agent generative adversarial imitation learning

We select $\psi_i = \psi_{\text{GA}}$ to be our reward function regularizer; this corresponds to the two-player game introduced in Generative Adversarial Imitation Learning (GAIL, [16]). For each agent i, we have a discriminator (denoted as D_{ω_i}) mapping state action-pairs to *scores* optimized to discriminate expert demonstrations from behaviors produced by π_i. Implicitly, D_{ω_i} plays the role of a reward function for the generator, which in turn attempts to train the agent to maximize its reward thus fooling the discriminator. We optimize the following objective:

$$\min_\theta \max_\omega \mathbb{E}_{\pi_\theta}[\sum_{i=1}^N \log D_{\omega_i}(s, a_i)] + \mathbb{E}_{\pi_E}[\sum_{i=1}^N \log(1 - D_{\omega_i}(s, a_i))] \tag{10}$$

We update π_θ through reinforcement learning, where we also use a baseline V_ϕ to reduce variance. We outline the algorithm – Multi-Agent GAIL (MAGAIL) – in Appendix B.

We can augment the reward regularizer $\psi(r)$ using an indicator $y(r)$ denoting whether r fits our prior knowledge; the augmented reward regularizer $\hat{\psi} : \mathbb{R}^{\mathcal{S} \times \mathcal{A}} \to \mathbb{R} \cup \{\infty\}$ is then: $\psi(r)$ if $y(r) = 1$ and ∞ if $y(r) = 0$. We introduce three types of $y(r)$ for common settings.

Centralized The easiest case is to assume that the agents are fully cooperative, i.e. they share the same reward function. Here $y(r) = \mathbb{I}(r_1 = r_2 = \ldots r_n)$ and $\psi(r) = \psi_{\text{GA}}(r)$. One could argue this corresponds to the GAIL case, where the RL procedure operates on multiple agents (a joint policy).

Decentralized We make no prior assumptions over the correlation between the rewards. Here $y(r) = \mathbb{I}(r_i \in \mathbb{R}^{\mathcal{O}_i \times \mathcal{A}_i})$ and $\psi_i(r_i) = \psi_{\text{GA}}(r_i)$. This corresponds to one discriminator for each agent which discriminates the trajectories as observed by agent i. However, these discriminators are not learned independently as they interact indirectly via the environment.

[1]The set of Nash equilibria is not always convex, so we have to assume MARL(r) returns a unique solution.

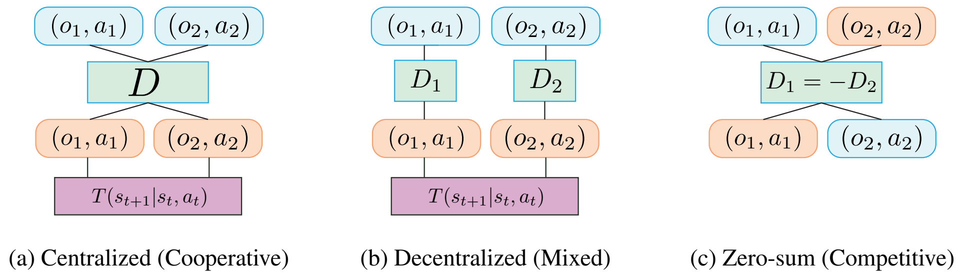

Figure 1: Different MAGAIL algorithms obtained with different priors on the reward structure. The discriminator tries to assign higher rewards to top row and low rewards to bottom row. In centralized and decentralized, the policy operates with the environment to match the expert rewards. In zero-sum, the policy do not interact with the environment; expert and policy trajectories are paired together as input to the discriminator.

Zero Sum Assume there are two agents that receive opposite rewards, so $r_1 = -r_2$. As such, ψ is no longer additively separable. Nevertheless, an adversarial training procedure can be designed using the following fact:

$$v(\pi_{E_1}, \pi_2) \geq v(\pi_{E_1}, \pi_{E_2}) \geq v(\pi_1, \pi_{E_2})$$

where $v(\pi_1, \pi_2) = \mathbb{E}_{\pi_1, \pi_2}[r_1(s, a)]$ is the expected outcome for agent 1. The discriminator could maximize the reward for trajectories in (π_{E_1}, π_2) and minimize the reward for trajectories in (π_2, π_{E_1}).

These three settings are in summarized in Figure 1.

4.2 Multi-agent actor-critic with Kronecker factors

To optimize over the generator parameters θ in Eq. (10) we wish to use an algorithm for multi-agent RL that has good sample efficiency in practice. Our algorithm, which we refer to as Multi-agent Actor-Critic with Kronecker-factors (MACK), is based on Actor-Critic with Kronecker-factored Trust Region (ACKTR, [24]), a state-of-the-art natural policy gradient [25, 26] method in deep RL. MACK uses the framework of centralized training with decentralized execution [17]; policies are trained with additional information to reduce variance but such information is not used during execution time. We let the advantage function of every agent agent be a function of all agents' observations and actions:

$$A_{\phi_i}^{\pi_i}(s, a_t) = \sum_{j=0}^{k-1}(\gamma^j r(s_{t+j}, a_{t+j}) + \gamma^k V_{\phi_i}^{\pi_i}(s_{t+k}, a_{-i,t})) V_{\phi_i}^{\pi_i}(s_t, a_{-i,t}) \tag{11}$$

where $V_{\phi_i}^{\pi_i}(s_k, a_{-i})$ is the baseline for i, utilizing the additional information (a_{-i}) for variance reduction. We use K-FAC to update both θ and ϕ but without trust regions to schedule the learning rate – a linear decay learning rate schedule achieves similar empirical performance.

MACK has some notable differences from Multi-Agent Deep Deterministic Policy Gradient [14]. On the one hand, MACK does not assume knowledge of other agent's policies nor tries to infer them; the value estimator merely collects experience from other agents (and treats them as black boxes). On the other hand, MACK does not require gradient estimators such as Gumbel-softmax [27, 28] to optimize over discrete actions, which is necessary for DDPG [29].

5 Experiments

We evaluate the performance of (centralized, decentralized, and zero-sum versions) of MAGAIL under two types of environments. One is a particle environment which allows for complex interactions and behaviors; the other is a control task, where multiple agents try to cooperate and move a plank forward. We collect results by averaging over 5 random seeds. Our implementation is based on OpenAI baselines [30]; please refer to Appendix C for implementation details.

We compare our methods (centralized, decentralized, zero-sum MAGAIL) with two baselines. The first is behavior cloning (BC), which learns a maximum likelihood estimate for a_i given each state s

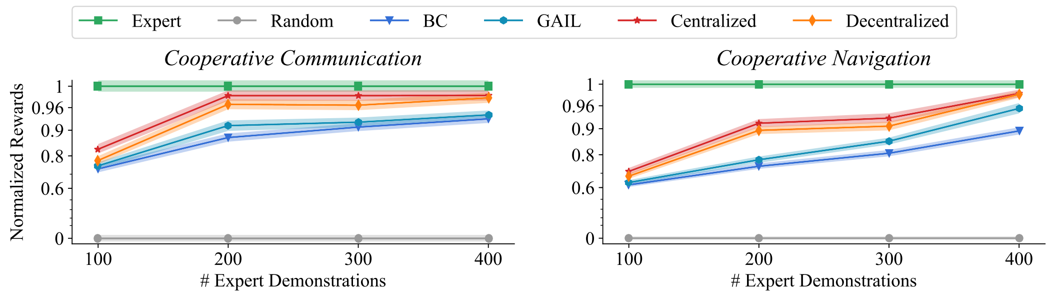

Figure 2: Average true reward from cooperative tasks. Performance of experts and random policies are normalized to one and zero respectively. We use inverse log scale for better comparison.

and does not require actions from other agents. The second baseline is the GAIL IRL baseline that operates on each agent separately – for each agent we first pretrain the other agents with BC, and then train the agent with GAIL; we then gather the trained GAIL policies from all the agents and evaluate their performance.

5.1 Particle environments

We first consider the particle environment proposed in [14], which consists of several agents and landmarks. We consider two cooperative environments and two competitive ones. All environments have an underlying true reward function that allows us to evaluate the performance of learned agents.

The environments include: **Cooperative Communication** – two agents must cooperate to reach one of three colored landmarks. One agent ("speaker") knows the goal but cannot move, so it must convey the message to the other agent ("listener") that moves but does not observe the goal. **Cooperative Navigation** – three agents must cooperate through physical actions to reach three landmarks; ideally, each agent should cover a single landmark. **Keep-Away** – two agents have contradictory goals, where agent 1 tries to reach one of the two targeted landmarks, while agent 2 (the adversary) tries to keep agent 1 from reaching its target. The adversary does not observe the target, so it must act based on agent 1's actions. **Predator-Prey** – three slower cooperating adversaries must chase the faster agent in a randomly generated environment with obstacles; the adversaries are rewarded by touching the agent while the agent is penalized.

For the cooperative tasks, we use an analytic expression defining the expert policy; for the competitive tasks, we use MACK to train expert policies based on the true underlying rewards (using larger policy and value networks than the ones that we use for imitation). We then use the expert policies to simulate trajectories $\mathcal{D}$, and then do imitation learning on $\mathcal{D}$ as demonstrations, where we assume the underlying rewards are unknown. Following [31], we pretrain our Multi-Agent GAIL methods and the GAIL baseline using behavior cloning as initialization to reduce sample complexity for exploration. We consider 100 to 400 episodes of expert demonstrations, each with 50 timesteps, which is close to the amount of timesteps used for the control tasks in [16]. Moreover, we randomly sample the starting position of agent and landmarks each episode, so our policies have to learn to generalize when they encounter new settings.

5.1.1 Cooperative tasks

We evaluate performance in cooperative tasks via the average expected reward obtained by all the agents in an episode. In this environment, the starting state is randomly initialized, so generalization is crucial. We do not consider the zero-sum case, since it violates the cooperative nature of the task. We display the performance of centralized, decentralized, GAIL and BC in Figure 2.

Naturally, the performance of BC and MAGAIL increases with more expert demonstrations. MAGAIL performs consistently better than BC in all the settings; interestingly, in the cooperative communication task, centralized MAGAIL is able to achieve expert-level performance with only 200 demonstrations, but BC fails to come close even with 400 trajectories. Moreover, the centralized MAGAIL performs slightly better than decentralized MAGAIL due to the better prior, but decentralized MAGAIL still learns a highly correlated reward between two agents.

Table 1: Average agent rewards in competitive tasks. We compare behavior cloning (BC), GAIL (G), Centralized (C), Decentralized (D), and Zero-Sum (ZS) methods. Best marked in bold (high vs. low rewards is preferable depending on the agent vs. adversary role).

Task	Predator-Prey								
Agent	Behavior Cloning					G	C	D	ZS
Adversary	BC	G	C	D	ZS	Behavior Cloning			
Rewards	-93.20	-93.71	-93.75	-95.22	**-95.48**	-90.55	-91.36	**-85.00**	-89.4

Task	Keep-Away								
Agent	Behavior Cloning					G	C	D	ZS
Adversary	BC	G	C	D	ZS	Behavior Cloning			
Rewards	24.22	24.04	23.28	23.56	**23.19**	26.22	26.61	**28.73**	27.80

5.1.2 Competitive tasks

We consider all three types of Multi-Agent GAIL (centralized, decentralized, zero-sum) and BC in both competitive tasks. Since there are two opposing sides, it is hard to measure performance directly. Therefore, we compare by letting (agents trained by) BC play against (adversaries trained by) other methods, and vice versa. From Table 1, decentralized and zero-sum MAGAIL often perform better than centralized MAGAIL and BC, which suggests that the selection of the suitable prior $\hat{\psi}$ is important for good empirical performance. More details for all the particle environments are in the appendix.

5.2 Cooperative control

In some cases we are presented with sub-optimal expert demonstrations because the environment has changed; we consider this case in a cooperative control task [32], where N bipedal walkers cooperate to move a long plank forward; the agents have incentive to collaborate since the plank is much longer than any of the agents. The expert demonstrates its policy on an environment with no bumps on the ground and heavy weights, while we perform imitation in an new environment with bumps and lighter weights (so one is likely to use too much force). Agents trained with BC tend to act more aggressively and fail, whereas agents trained with centralized MAGAIL can adapt to the new environment. With 10 (imperfect) expert demonstrations, BC agents have a chance of failure of 39.8$\%$ (with a reward of 1.26), while centralized MAGAIL agents fail only 26.2$\%$ of the time (with a reward of 26.57). We show videos of respective policies in the supplementary.

6 Related work and discussion

There is a vast literature on single-agent imitation learning [33]. Behavior Cloning (BC) learns the policy through supervised learning [34]. Inverse Reinforcement Learning (IRL) assumes the expert policy optimizes over some unknown reward, recovers the reward, and learns the policy through reinforcement learning (RL). BC does not require knowledge of transition probabilities or access to the environment, but suffers from compounding errors and covariate shift [35, 22].

Most existing work in multi-agent imitation learning assumes the agents have very specific reward structures. The most common case is fully cooperative agents, where the challenges mainly lie in other factors, such as unknown role assignments [36], scalability to swarm systems [37] and agents with partial observations [38]. In non-cooperative settings, [39] consider the case of IRL for two-player zero-sum games and cast the IRL problem as Bayesian inference, while [40] assume agents are non-cooperative but the reward function is a linear combination of pre-specified features.

Our work is the first to propose a general multi-agent IRL framework that bridges the gap between state-of-the art multi-agent reinforcement learning methods [14, 17] and implicit generative models such as generative adversarial networks [41]. Experimental results demonstrate that it is able to imitate complex behaviors in high-dimensional environments with both cooperative and adversarial interactions. An interesting research direction is to explore new techniques for gathering expert demonstration; for example, when the expert is allowed to aid the agents by participating in part of the agent's learning process [23].

References

[1] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, S. Legg, and K. Kavukcuoglu, “Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures,” *arXiv preprint arXiv:1802.01561*, 2018.

[2] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. Dragan, “Inverse reward design,” in *Advances in Neural Information Processing Systems*, pp. 6768–6777, 2017.

[3] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, “Concrete problems in ai safety,” *arXiv preprint arXiv:1606.06565*, 2016.

[4] D. Amodei and J. Clark, “Faulty reward functions in the wild,” 2016.

[5] P. Peng, Q. Yuan, Y. Wen, Y. Yang, Z. Tang, H. Long, and J. Wang, “Multiagent bidirectionally-coordinated nets for learning to play starcraft combat games,” *arXiv preprint arXiv:1703.10069*, 2017.

[6] L. Matignon, L. Jeanpierre, A.-I. Mouaddib, *et al.*, “Coordinated multi-robot exploration under communication constraints using decentralized markov decision processes.,” in *AAAI*, 2012.

[7] J. Z. Leibo, V. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel, “Multi-agent reinforcement learning in sequential social dilemmas,” in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 464–473, International Foundation for Autonomous Agents and Multiagent Systems, 2017.

[8] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, “Maximum entropy inverse reinforcement learning.,” in *AAAI*, vol. 8, pp. 1433–1438, Chicago, IL, USA, 2008.

[9] P. Englert and M. Toussaint, “Inverse kkt–learning cost functions of manipulation tasks from demonstrations,” in *Proceedings of the International Symposium of Robotics Research*, 2015.

[10] C. Finn, S. Levine, and P. Abbeel, “Guided cost learning: Deep inverse optimal control via policy optimization,” in *International Conference on Machine Learning*, pp. 49–58, 2016.

[11] B. Stadie, P. Abbeel, and I. Sutskever, “Third person imitation learning,” in *ICLR*, 2017.

[12] A. Y. Ng, S. J. Russell, *et al.*, “Algorithms for inverse reinforcement learning.,” in *Icml*, pp. 663–670, 2000.

[13] P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in *Proceedings of the twenty-first international conference on Machine learning*, p. 1, ACM, 2004.

[14] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, “Multi-agent actor-critic for mixed cooperative-competitive environments,” *arXiv preprint arXiv:1706.02275*, 2017.

[15] J. Hu, M. P. Wellman, *et al.*, “Multiagent reinforcement learning: theoretical framework and an algorithm.,” in *ICML*, vol. 98, pp. 242–250, Citeseer, 1998.

[16] J. Ho and S. Ermon, “Generative adversarial imitation learning,” in *Advances in Neural Information Processing Systems*, pp. 4565–4573, 2016.

[17] J. Foerster, Y. Assael, N. de Freitas, and S. Whiteson, “Learning to communicate with deep multi-agent reinforcement learning,” in *Advances in Neural Information Processing Systems*, pp. 2137–2145, 2016.

[18] M. L. Littman, “Markov games as a framework for multi-agent reinforcement learning,” in *Proceedings of the eleventh international conference on machine learning*, vol. 157, pp. 157–163, 1994.

[19] M. Bloem and N. Bambos, “Infinite time horizon maximum causal entropy inverse reinforcement learning,” in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, pp. 4911–4916, IEEE, 2014.

[20] J. Filar and K. Vrieze, *Competitive Markov decision processes*. Springer Science & Business Media, 2012.

[21] H. Prasad and S. Bhatnagar, "A study of gradient descent schemes for general-sum stochastic games," *arXiv preprint arXiv:1507.00093*, 2015.

[22] S. Ross, G. Gordon, and D. Bagnell, "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*, pp. 627–635, 2011.

[23] D. Hadfield-Menell, S. J. Russell, P. Abbeel, and A. Dragan, "Cooperative inverse reinforcement learning," in *Advances in neural information processing systems*, pp. 3909–3917, 2016.

[24] Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba, "Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation," in *Advances in neural information processing systems*, pp. 5285–5294, 2017.

[25] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural computation*, vol. 10, no. 2, pp. 251–276, 1998.

[26] S. M. Kakade, "A natural policy gradient," in *Advances in neural information processing systems*, pp. 1531–1538, 2002.

[27] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[28] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," *arXiv preprint arXiv:1611.00712*, 2016.

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

[30] P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Openai baselines." `https://github.com/openai/baselines`, 2017.

[31] Y. Li, J. Song, and S. Ermon, "Infogail: Interpretable imitation learning from visual demonstrations," *arXiv preprint arXiv:1703.08840*, 2017.

[32] J. K. Gupta and M. Egorov, "Multi-agent deep reinforcement learning environment." `https://github.com/sisl/madrl`, 2017.

[33] J. A. Bagnell, "An invitation to imitation," tech. rep., CARNEGIE-MELLON UNIV PITTSBURGH PA ROBOTICS INST, 2015.

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

[35] S. Ross and D. Bagnell, "Efficient reductions for imitation learning.," in *AISTATS*, pp. 3–5, 2010.

[36] H. M. Le, Y. Yue, and P. Carr, "Coordinated multi-agent imitation learning," *arXiv preprint arXiv:1703.03121*, 2017.

[37] A. Šošic, W. R. KhudaBukhsh, A. M. Zoubir, and H. Koeppl, "Inverse reinforcement learning in swarm systems," *stat*, vol. 1050, p. 17, 2016.

[38] K. Bogert and P. Doshi, "Multi-robot inverse reinforcement learning under occlusion with interactions," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pp. 173–180, International Foundation for Autonomous Agents and Multiagent Systems, 2014.

[39] X. Lin, P. A. Beling, and R. Cogill, "Multi-agent inverse reinforcement learning for zero-sum games," *arXiv preprint arXiv:1403.6508*, 2014.

[40] T. S. Reddy, V. Gopikrishna, G. Zaruba, and M. Huber, "Inverse reinforcement learning for decentralized non-cooperative multiagent systems," in *Systems, Man, and Cybernetics (SMC), 2012 IEEE International Conference on*, pp. 1930–1935, IEEE, 2012.

[41] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[42] J. Martens and R. Grosse, “Optimizing neural networks with kronecker-factored approximate curvature,” in *International Conference on Machine Learning*, pp. 2408–2417, 2015.