

Multi-Agent Inverse Q-Learning from Demonstrations

Nathaniel Haynam¹ Adam Khoja¹ Dhruv Kumar¹
Vivek Myers¹ Erdem Biyik^{1,2}

Abstract—When reward functions are hand-designed, deep reinforcement learning algorithms often suffer from reward misspecification, causing them to learn suboptimal policies in terms of the intended task objectives. In the single-agent case, inverse reinforcement learning (IRL) techniques attempt to address this issue by inferring the reward function from expert demonstrations. However, in multi-agent problems, misalignment between the learned and true objectives is exacerbated due to increased environment non-stationarity and variance that scales with multiple agents. As such, in multi-agent general-sum games, multi-agent IRL algorithms have difficulty balancing cooperative and competitive objectives. To address these issues, we propose Multi-Agent Marginal Q-Learning from Demonstrations (MAMQL), a novel sample-efficient framework for multi-agent IRL. For each agent, MAMQL learns a critic marginalized over the other agents’ policies, allowing for a well-motivated use of Boltzmann policies in the multi-agent context. We identify a connection between optimal marginalized critics and single-agent soft-Q IRL, allowing us to apply a direct, simple optimization criterion from the single-agent domain. Across our experiments on three different simulated domains, MAMQL significantly outperforms previous multi-agent methods in average reward, sample efficiency, and reward recovery by often more than 2-5x. We make our code available at <https://sites.google.com/view/mamql>.

I. INTRODUCTION

Inverse reinforcement learning (IRL) has been widely used to approach many complex problems in control and decision-making, such as autonomous driving [1], [2], resource management [3], [4], and team sports [5], [6]. These scenarios often involve multiple agents, where each agent must consider both the cooperative and competitive nature of the environment. IRL aims to train agents to learn the optimal policy through the use of expert demonstrations. While effective in single-agent cases, we cannot naively apply single-agent IRL in multi-agent environments because the other agents’ strategies influence the policy of the agent we are training. Multi-agent approaches to IRL (MAIRL), such as MA-AIRL [7], MAAC [8], and MA-DAC [9], effectively combat the non-stationarity of multi-agent environments caused by conflicting agent policies. However, they often struggle to robustly recover rewards that explain the expert demonstrations, especially in general-sum games. For example, in autonomous driving, each agent has competitive and cooperative objectives. While the competitive objective of arriving at the destination quickly is clearly defined, cooperating with other agents to avoid collisions in various traffic control systems is more ambiguous. Such an environment

has multiple equilibria with agents optimizing for a mix of cooperative/competitive objectives; though straightforward to demonstrate, learning correct objective functions for autonomous driving is unsolved. Thus, we are interested in the following question: how can we learn the agents’ objective functions in a multi-agent system that explain the cooperative and competitive behaviors seen in the expert demonstrations?

To solve this problem, we propose Multi-Agent Marginal Q-Learning from Demonstrations (MAMQL). MAMQL learns a marginalization of the action-value function of each agent by taking the expectation of their value function over the action space of all other agents. MAMQL then uses the marginalized action-value function to approximate a reward function that is representative of the mixed cooperative/competitive behaviors observed in the expert demonstrations.

To test our algorithm, we work with three multi-agent environments of interest. First, we introduce a new general-sum gridworld environment called “Gems.” The second is Overcooked [10], a standard benchmark environment for cooperative multi-agent setups. Third, we utilize the Highway environment [11], an autonomous driving simulator with scalability for testing the algorithms in scenarios with varying numbers of autonomous vehicles. All three environments provide a rich space of strategies where effective cooperation differs significantly from behavior that does not acknowledge the other agents. Thus, for each environment, the expert demonstrations may contain different equilibria, making reward recovery relatively difficult. Yet, we observe MAMQL is a significant improvement compared to the state-of-the-art methods, illustrating its robustness and sample-efficiency.

II. RELATED WORK

Our approach connects ideas from inverse reinforcement learning and imitation learning to exploit the unique structure of multi-agent problems.

A. Inverse Reinforcement Learning

In traditional reinforcement learning (RL), an agent operating in a Markov decision process (MDP) is trained to maximize the cumulative expected reward by interacting with the environment. Inverse reinforcement learning (IRL) solves the opposite problem: given expert demonstrations, recover the reward function [12], [13], [14], [15], [16], [17], [18]. After performing IRL, the recovered reward function can be used to train a policy that mimics the expert’s behavior. However, with suboptimal experts, as is typically the case with

¹Electrical Engineering and Computer Sciences, UC Berkeley

²Computer Science, University of Southern California

human data, inferring the true reward becomes increasingly difficult. One way to address this issue is the MaxEnt IRL framework [19], [20] which models suboptimality by merely assuming experts are rational in proportion to the reward they expect to receive. Another approach, IQ-Learn [21], uses an inverse soft-Q learning framework to directly infer policies from demonstrations. This reduces the non-stationarity of the environment, streamlining the extraction of reward and policy functions, a property we borrow in MAMQL. Our contribution is to show that inverse soft-Q learning can be extended to the multi-agent setting by learning marginalized critics for each agent. A direct extension of IQ-Learn to the multi-agent context serves as a baseline for our algorithm.

B. Imitation Learning

Broadly, imitation learning (IL) is a technique that learns a policy by mimicking an expert's behavior [22], [16], [23], [24]. Both IRL and imitation learning (IL) utilize an expert demonstration dataset to learn optimal agent policies, though IRL also infers rewards, potentially at additional cost [16].

In multi-agent environments, IL encounters difficulties attaining a stable equilibrium during joint optimization [25]. Our proposed MAMQL approach, in addition to learning reward functions, learns policies that empirically satisfy a generalized version of the Nash Equilibrium similar to [26], addressing the limitations of existing IL methods that do not sufficiently capture both cooperative and competitive objectives. This approach enhances the robustness of learned policies and their similarity to those of expert agents, which is particularly advantageous in scenarios involving strategic interactions among multiple agents.

C. Multi-Agent Learning

The extension of IRL and IL to the multi-agent case introduces a number of complications. With more than one agent, the IRL objective becomes to learn a reward function for each agent, while the IL objective requires producing a joint policy as agents' actions depend on their interactions with the other agents, e.g., through conventions [27], [28]. The notion of an optimal policy is also much more challenging to define in this context since the expected return of one agent's policy depends on the policies of all other agents. Extensive non-stationarity hampers theoretical guarantees and causes entire classes of algorithms to exhibit slow convergence to occasionally suboptimal equilibria in practice compared to expert trajectories [8]. Some techniques, such as MA-AIRL [7], adapt soft methods that model bounded rationality into the multi-agent case, allowing it to connect the significant literature in single-agent MaxEnt IRL [19] to the multi-agent case where maximum entropy techniques have seen less exploration. Our approach adopts a similar framework that interprets an agent's policy as a generalized Boltzmann policy to construct a joint optimization objective.

An additional feature of interest in the multi-agent context is the structure of reward functions. Zero-sum games, as well as fully cooperative games where all agents share the same reward function, act as simplifying assumptions

[29]. Recent work has attempted to tackle multi-agent IRL in such settings [30]. Our algorithm seeks to relax these assumptions by allowing mixed cooperative and competitive objectives, thus dealing with general-sum environments. In these settings, agents may be balancing complex interactions between incentives, working together toward shared goals in some contexts and competing in others, forcing IRL methods to infer complex equilibria from expert data.

III. PROBLEM FORMULATION

We operate in a Markov game modeled by the tuple $\langle n, \mathcal{S}, \mathcal{A}, T, \mathcal{R}, p_0 \rangle$, where n is the number of agents, \mathcal{S} is the set of states, with each state denoted by $s \in \mathcal{S}$, \mathcal{A} is a *discrete* set of actions symmetrically available to each agent, $T : \mathcal{S} \times \mathcal{A}^n \rightarrow \Delta(\mathcal{S})$ is the transition function, $\mathcal{R} : \mathcal{S} \times \mathcal{A}^n \rightarrow \mathbb{R}^n$ gives the true rewards for each agent, and $p_0 \in \Delta(\mathcal{S})$ is the distribution of starting states. For a vector of actions $\mathbf{a} \in \mathcal{A}^n$, we adopt the notation (a_i, \mathbf{a}_{-i}) to denote a vector of actions where i^{th} agent's action is a_i and \mathbf{a}_{-i} represents the actions of all agents except i . We denote a joint policy as π where $\pi_i : \mathcal{S} \rightarrow \Delta(\mathcal{A})$ is the policy of the i^{th} agent, and π_{-i} is the marginal policy of all agents except i .

Given the dynamics information $\langle n, \mathcal{S}, \mathcal{A}, T, p_0 \rangle$ without the reward function, and a dataset ρ_E of expert transitions of the form (s, \mathbf{a}, s') , our goal is to reconstruct \mathcal{R} . In a general-sum setting, this means we learn a distinct reward function \mathcal{R}_i for each agent i , naturally accommodating cooperative and competitive objectives. If the environment is purely cooperative, \mathcal{R}_i would converge to near-identical functions, whereas in environments with distinct roles or asymmetries, each agent's reward function may diverge from the others to reflect its unique objectives. Moreover, we preserve agent indices in expert data (s, \mathbf{a}, s') to ensure that each agent i learns from transitions relevant to its own perspective, which is especially important when agents fulfill different roles (e.g., Overcooked chefs, or vehicles with unique routes in a traffic network). Although one could shuffle indices in perfectly symmetric tasks, real-world domains typically exhibit role-specific state and action distributions, making fixed indexing the more natural approach.

IV. APPROACH

We develop our approach by studying the behavior of Boltzmann policies in a multi-agent equilibrium. Yu et al. [7] argue that joint policies which satisfy a desirable optimality condition, where each π_i is the optimal response modulated by a rationality parameter $\lambda > 0$ to fixed π_{-i} , must satisfy the recurrence

$$\pi_i(a_i | s) = \frac{\exp \lambda \mathbb{E}_{\tilde{\mathbf{a}}_{-i} \sim \pi_{-i}(\cdot | s)}[Q_i^\pi(s, a_i, \tilde{\mathbf{a}}_{-i})]}{\sum_{\tilde{a}_i \in \mathcal{A}} \exp \lambda \mathbb{E}_{\tilde{\mathbf{a}}_{-i} \sim \pi_{-i}(\cdot | s)}[Q_i^\pi(s, \tilde{a}_i, \tilde{\mathbf{a}}_{-i})]} \quad (1)$$

for all state-action pairs, where Q_i^π is the true expected discounted return of π_i given fixed π_{-i} . Given a learned critic $\hat{Q}_i^\pi : \mathcal{S} \times \mathcal{A}^n \rightarrow \mathbb{R}$ which estimates expected return for agent i , a policy π_i satisfying Eq. (1) for \hat{Q}_i^π in place of

Q_i^π can be interpreted as a “generalized Boltzmann policy” with respect to the critic.

As a single-agent IRL method, IQ-Learn [21] assumes expert policies to be generalized Boltzmann policies with respect to a learned critic. With this assumption, it produces a non-adversarial optimization objective, unlike previous methods involving adversarial optimization like GAIL [16]. We follow a similar path with generalized Boltzmann policies to construct our algorithms in the multi-agent context.

A. Multi-Agent Marginal Q-Learning (MAMQL)

Let us first consider a learning setup where, for each agent i , we learn a critic $Q_{\psi_i} : \mathcal{S} \times \mathcal{A}^n \rightarrow \mathbb{R}$. If we wish to compute π as a generalized Boltzmann policy given the critics, the form of Eq. (1) does not provide an explicit construction, as $\pi_i(a_i | s)$ depends on $\pi_{-i}(a_{-i} | s)$. Although it is possible to learn policies alongside critics and enforce Eq. (1) through joint optimization, we simplify the setup by learning *marginalized* critics $\bar{Q}_{\psi_i} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$. That is, for a fixed π_{-i} , we require that the marginalized critic is defined as the expectation over $\tilde{\mathbf{a}}_{-i}$ sampled according to $\pi_{-i}(\cdot | s)$, i.e.,

$$\bar{Q}_{\psi_i}(s, a_i) = \mathbb{E}_{\tilde{\mathbf{a}}_{-i} \sim \pi_{-i}(\cdot | s)}[Q_i^{\pi_i}(s, a_i, \tilde{\mathbf{a}}_{-i})]. \quad (2)$$

A set of policies satisfying Eq. (1) must be Boltzmann policies of a set of marginalized critics \bar{Q}_{ψ_i} satisfying Eq. (2) by direct substitution. Thus, we will henceforth treat all policies π as Boltzmann with respect to a set of marginalized critics so that

$$\pi_i(a_i | s) = \frac{\exp \lambda \bar{Q}_{\psi_i}(s, a_i)}{\sum_{a'} \exp \lambda \bar{Q}_{\psi_i}(s, a')}. \quad (3)$$

Next, we generalize the soft Bellman condition given by Haarnoja et al. [31] to the multi-agent setting. We take the soft value function for agent i given π as

$$V_i^\pi(s) = \mathbb{E}_{\tilde{\mathbf{a}} \sim \pi(\cdot | s)}[Q_i^\pi(s, \tilde{\mathbf{a}}) - \log \pi_i(\tilde{\mathbf{a}}_i | s)] \quad (4)$$

where we note that the second term inside the expectation penalizes the entropy of π_i only, rather than π . We substitute Eq. (3) in Eq. (4) and simplify to get:

$$\begin{aligned} V_i^\pi(s) &= V_{\psi_i}^\pi(s) := (1 - \lambda) \mathbb{E}_{a_i \sim \pi_i(\cdot | s)}[\bar{Q}_{\psi_i}(s, a_i) \\ &\quad + \log \sum_{a' \in \mathcal{A}} \exp \lambda \bar{Q}_{\psi_i}(s, a')]. \end{aligned} \quad (5)$$

We note a connection between Eq. (1) and maximum entropy reinforcement learning¹: for fixed π_{-i} , the choice of \bar{Q}_{ψ_i} which makes the Boltzmann policy π_i satisfy Eq. (1) must also satisfy a marginalized soft Bellman condition:

$$\begin{aligned} \mathbb{E}_{\tilde{\mathbf{a}}_{-i} \sim \pi_{-i}(\cdot | s)}[\mathcal{R}_i(s, a_i, \tilde{\mathbf{a}}_{-i})] &= \bar{R}_{\psi_i}^\pi(s, a_i) = \bar{Q}_{\psi_i}(s, a_i) \\ &\quad - \gamma \mathbb{E}_{\tilde{\mathbf{a}}_{-i} \sim \pi_{-i}(\cdot | s), s' \sim T(\cdot | s, a_i, \tilde{\mathbf{a}}_{-i})}[V_{\psi_i}^\pi(s')]. \end{aligned} \quad (6)$$

For a set of marginalized critics satisfying Eq. (2), all choices of s and a_i in Eq. (6) introduce $|\mathcal{S}| |\mathcal{A}|$ constraints on \mathcal{R}_i . Namely, a learned reward estimate R_{μ_i} for \mathcal{R}_i must,

for each (s, a_i) pair and fixed marginal critics, satisfy the constraint

$$\bar{R}_{\psi_i}^\pi(s, a_i) = \mathbb{E}_{\tilde{\mathbf{a}}_{-i} \sim \pi_{-i}(\cdot | s)} R_{\mu_i}(s, a_i, \tilde{\mathbf{a}}_{-i}). \quad (7)$$

Let us trace back through these results to emphasize their role. We found that joint policies satisfying desirable optimality conditions Eq. (1) can be expressed with the Boltzmann policies of a set of marginalized critics. The non-stationarity of expected return for an agent’s policy (since it depends on other agents’ policies) is taken into account implicitly by the definition of the marginalized critics (Eq. (2)), allowing for a simple expression of the joint policy π in terms of the Boltzmann policies of the learned critics (Eq. (3)). Each marginal critic is also an optimal soft-Q function in the single-agent MDP induced by fixed π_{-i} , since it satisfies Eq. (6), which is equivalent to the soft Bellman condition for that MDP. The marginalized critics provide a set of conditions which the experts’ reward functions must meet (Eq. (7)).

B. Training

The marginal Q -function \bar{Q}_{ψ_i} is an optimal soft- Q function in the single-agent MDP induced by fixed π_{-i} . However, identifying an optimal soft Q -function whose Boltzmann policy best satisfies an expert dataset in the single-agent setting is exactly the question considered by Garg et al. [21]. In the discrete setting, the authors show that the optimal soft Q -function is obtained by a concave objective in terms of the learned Q -function and its soft value estimate. Translating their objective to our setting gives the critic loss function for agent i :

$$\min_{\psi_i} (1 - \gamma) \mathbb{E}_{s_0 \sim p_0}[V_{\psi_i}^\pi(s_0)] - \mathbb{E}_{(s, a) \sim \rho_E}[\phi(\bar{R}_{\psi_i}^\pi(s, a))] \quad (8)$$

where ρ_E is the state-action distribution of the expert dataset, and ϕ is a concave reward regularizer. Several examples of regularizers in terms of common divergence functions are given in Appendix B of [21]. For example, $\phi(x) = x$ corresponds to total variation divergence while $\phi(x) = x - \frac{x^2}{4}$ corresponds to Pearson χ^2 divergence. It is the latter regularizer we use in our experiments.

This objective allows for offline learning, but, following the discussion by Garg et al. [21], Section 5.1, we achieve better performance in an online version by sampling $\mathbb{E}_{(s, a, s') \sim \rho_\pi}[V^\pi(s) - \gamma V^\pi(s')]$ to substitute $(1 - \gamma) \mathbb{E}_{s_0 \sim p_0}[V_{\psi_i}^\pi(s_0)]$ in our loss function, where ρ_π is the distribution of transitions induced by joint policy π .

The learned marginalized critics do not directly provide estimates of rewards for individual state-action pairs. However, Eq. (7) shows that we recover a set of constraints for R_{μ_i} . To reconstruct R_{μ_i} , we sample state-action pairs from ρ_π and update according to an MSE objective which enforces those constraints:

$$\begin{aligned} \min_{\mu_i} \mathbb{E}_{(s, a) \sim \rho_\pi} &\left[\left(\bar{R}_{\psi_i}^\pi(s, a) \right. \right. \\ &\quad \left. \left. - \mathbb{E}_{\tilde{\mathbf{a}}_{-i} \sim \pi_{-i}(\cdot | s)} R_{\mu_i}(s, a, \tilde{\mathbf{a}}_{-i}) \right)^2 \right] + \beta(R_{\mu_i}) \end{aligned} \quad (9)$$

¹See Section 3.1 of [7] for relevant discussion in a fixed-horizon setting.

where β is a reward regularization component. The pseudocode of the online method is provided in Algorithm 1.

One practical consideration is that it is often difficult, given an expert transition (s, a, s') , to exactly compute the term $\mathbb{E}_{\tilde{a}_{-i} \sim \pi_{-1}(\cdot|s), s'' \sim T(\cdot|s, a_i, \tilde{a}_{-i})}[V_{\psi_i}^\pi(s'')]$ from Eq. (6), as it in general requires computing $|\mathcal{A}|^{n-1}$ transitions. Even unbiased sampling is unwieldy, requiring importance sampling that leverages state-action occupancy measures from the expert dataset. We found an acceptable alternative is the biased one-sample estimate of the term as $V_{\psi_i}^\pi(s')$, since intuitively the expert joint actions a_{-i} which resulted in s' should be likely to be sampled by π_{-i} once it is performing well enough. This is the approach we employ in our experiments, which we will present next.

Algorithm 1 MAMQL

```

1: Input: Environment  $\langle n, \mathcal{S}, \mathcal{A}, T, p_0 \rangle$ 
   Expert transitions  $\rho_E = \{(s_i, a_i, s'_i)\}_{i=1}^N$ 
   Buffer  $\rho_\pi$  of agents' rollout transitions
   Regularization function  $\phi$ 
   Discount factor  $\gamma$ , learning rate  $\alpha$ 
2: Initialize  $\psi_i$ , for marginal critics  $\bar{Q}_{\psi_i}$  and each agent  $i$ 
3: Initialize  $\mu_i$ , for reward function  $R_{\mu_i}$  and each agent  $i$ 
4: repeat
5:   Add rollout trajectory to  $\rho_\pi$  using joint policy  $\pi_\psi$ 
6:   for  $i = 1$  to  $N$  do
7:     Sample expert and rollout transitions
        $\chi_E \sim \rho_E$  and  $\chi_\pi \sim \rho_\pi$ 
8:      $\psi_i \leftarrow \psi_i - \alpha \nabla_{\psi_i} (\mathbb{E}_{\chi_\pi} [V_{\psi_i}^\pi(s) - \gamma V_{\psi_i}^\pi(s')] - \mathbb{E}_{\chi_E} [\phi(\bar{R}_{\psi_i}^\pi(s, a_i))])$  (5, 6)
9:     Update  $\mu_i$  using the sample mean over  $\chi_\pi$  (9)
10:  until convergence
11: output Learned policies  $\pi_\psi$  and reward functions  $R_\mu$ 
```

V. EXPERIMENTS

To test the performance of MAMQL, we conducted experiments on three different simulated multi-agent environments. We will first introduce those environments in the next subsection. We provide visuals of the environments in Fig. 1.

A. Environments

As our simplest testbed, we introduce a new environment called “Gems,” a grid world where two agents (Red and Blue) collect gems. The gems are also red and blue and can only be collected by the agent of the same color and give 1 a reward. Additionally, there are purple gems, which are 6× as rewarding and can be collected by both agents, but only when they are standing on (possibly distinct) purple gems simultaneously. The game has a horizon of 45 moves. The action space consists of moving one step in the four cardinal directions and a stopping action.

We also performed experiments in the Overcooked environment [10], [32], [33], [34]. This environment has two agents who can move and interact with various ingredients and cooking tools. Each agent receives a reward of 10

when anyone delivers finished onion soup to a customer. Each agent’s action space involves moving in each cardinal direction, interacting with objects, or stopping. Overcooked has a maximum time horizon of 50 steps.

Finally, we performed experiments in the multi-agent intersection scenario from Highway-Env [35], [36], [11], [37]. We modified the original environment with only two agents to test our algorithm’s ability to learn mixed cooperative and competitive behaviors in traffic with many agents. Specifically, our modified environment consists of four agents. Each agent can decrease or increase their velocity to successfully negotiate right-of-way at a four-way intersection to arrive at a target destination without collision. Each agent receives a reward of 2 for arriving at their individual destination and a reward of -4 for colliding with another agent(s). Each agent’s action space involves decreasing, maintaining, or increasing velocity. An episode concludes when an agent crashes, or all agents reach their destination. While the agents are expected to demonstrate cooperation to avoid collisions, the discount factor introduces competition between them to get the right-of-way.

B. Baselines

We compare MAMQL against three alternative baselines. The first one is behavioral cloning where we learn a policy for each agent separately. Similarly, our second baseline is IQ-Learn [21] where we again learn the policies and the reward functions separately for different agents.

We also implemented a direct multi-agent extension of IQ-Learn, as the “IQ-Learn MA” baseline, by expanding the type signature of agent i ’s critic to include all agents actions (i.e., from type signature $\mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ of a single-agent critic to $\mathcal{S} \times \mathcal{A}^n \rightarrow \mathbb{R}$ which uses all agent actions from sampled transitions, but otherwise follows the IQ-Learn implementation exactly). Lastly, we compare MAMQL against the MA-AIRL [7] algorithm.

C. Datasets

For the Gems environment, we built an expert dataset from rollouts of expert RL policies trained using MADDPG [8], with a CNN architecture consisting of 3 convolutional layers with an MLP head with 3 hidden layers, with batch normalization between convolutional layers and a LeakyReLU activation function. To help train the experts, we also introduced reward shaping to encourage agents to approach purple gems. For Overcooked, we used a preexisting RL agent with an LSTM+MLP architecture [10] to produce our expert dataset. For the multi-agent Highway-Env scenario we trained an expert RL policy using DQN with social attention [38].

For each environment, we constructed several datasets of simulated expert transitions ranging from 500 to 70,000 environment steps.

D. Implementation Details

For the Gems environment, we ran all methods by training CNN architectures that have 3 convolutional layers and an



Fig. 1. On the left, the proposed multi-agent grid world environment where above square agents are working together to get a purple circular gem. At the center, the Overcooked cramped environment with one agent adding onions to the soup and the other agent ready with another. On the right, a four agent intersection from the bottom agent's viewpoint. The weighted line marks priority level of collision avoidance with another agent.

MLP head with 3 layers; ELU is used as the activation function and modest dropout weakly regularizes the networks. For the Overcooked and Highway-Env environments, we tested all our methods using an MLP architecture with four hidden layers and ELU as the activation function.

We comprehensively tuned available hyperparameters (regularization parameters, buffer size, number of expert examples, etc.) and report the average episodic return for each algorithm, each averaged across 1000 trajectories. Specifically, for a trajectory τ of length T in an environment with N agents, we define the episodic return $G(\tau)$ as

$$G(\tau) = \sum_{t=0}^{T-1} \sum_{i=1}^N r_t^i \quad (10)$$

where r_t^i is the reward received by agent i at time step t . The reported value is the mean of $G(\tau)$ over the 1000 test rollouts. All experiments were run on a computing cluster consisting of eight RTX A6000s; however, we note MAMQL can be trained on a single RTX 3060 with 6 GB GDDR6.

VI. RESULTS

Each entry of Table I shows the average return of the agents for the Overcooked and Highway-Env environments across 1000 episodes. The expert dataset size for all algorithms for Gems, Overcooked, and Highway-Env was 20k, 2k, 70k episodes respectively. We additionally plot the ratio of returns, [Agent 1 average return]/[Agent 2 average return], for the Gems environment since it is possible for the agents to acquire considerably different returns there.

TABLE I
AVERAGE REWARD

Algorithm	Gems	Overcooked	Highway-Env
Expert	14.42/14.42	24.3	1.72
BC	4.55/4.35	0.01	-0.14
IQ-Learn Indp	8.65/8.65	0	-1.14
IQ-Learn MA	11.45/11.45	8.48	0.31
MA-AIRL	2.04/2.04	0.01	-1.62
MAMQL	13.05/13.05	13.23	1.71

Comparing MAMQL with the baselines, we found that MAMQL outperformed all baselines approaching or matching the average reward of the expert in the cases of the Gems and Highway-Env environments. Additionally, MAMQL proved to converge roughly 4 \times faster than IQ-Learn MA.

On the Gems environment, MAMQL learns a policy on par with the expert and is qualitatively the only algorithm we observed to consistently balance cooperative and competitive rewards. On average, the policy learned by IQ-Learn MA only targets purple gems while MAMQL agents balance shared rewards with individual rewards, deviating to pick up red or blue gems on their way to the next purple gem. In the Overcooked environment, MAMQL is able to consistently complete a three-stage task, taking approximately 50 steps in the correct order to complete. Imitation learning and inverse reinforcement learning using single-agent methods (behavioral cloning and IQ-Learn) never successfully completed the task. On the multi-agent Highway-Env environment, we tested MAMQL’s ability to scale to several agents. With a continuous state space and discrete action space we trained 4 agents to negotiate right-of-way for a four-way intersection. MAMQL was able to achieve approximately expert-level average rewards with other MAIRL algorithms often unable to avoid collision in this challenging environment.

TABLE II
AVERAGE NUMBER OF EPISODES IN THOUSANDS FOR CONVERGENCE

Algorithm	Gems	Overcooked	Highway-Env
IQ-Learn MA	45	31	43
MA-AIRL	100	100	100
MAMQL	13	4	10

We also measure convergence time across our environments in Table II as the average number of episodes (in thousands) to converge to a good policy. The expert dataset size for all algorithms for Gems, Overcooked, and Highway-Env was 20k, 2k, 70k episodes respectively. MAMQL converges 2-4 \times faster to a near-optimal policy across all environments compared to our baselines. Additionally, while MAMQL

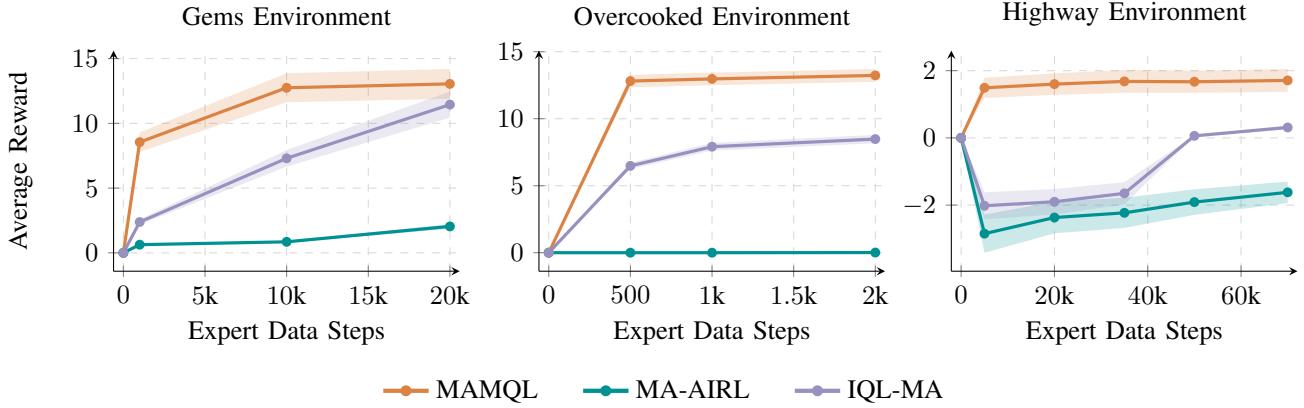


Fig. 2. Reward of recovered policy for MAMQL and baselines, across varying dataset sizes.

at most requires 14.5k episodes to converge to an optimal policy, other MAIRL algorithms generally take up to 100k episodes to train a good policy on environments of similar complexity to Gems, Overcooked, and Highway-Env based on previous work [39].

To distinguish our method as sample efficient instead of quickly overfitting to a dataset that encompasses the majority of the state/action space of the environment, we downscale our dataset size for each environment and observe the average reward over 1000 episodes of the learned policy (Fig. 2). For each environment, MAMQL outperforms our baselines in average reward with a significantly slower performance decay rate as the dataset size decreases.

Furthermore, we compare MAMQL to IQ-Learn MA using recovered rewards, i.e., the reward functions learned by the algorithms. Figure 3 demonstrates the reward recovery of training trajectories of MAMQL and baselines where reward recovery is calculated from the MSE between true and predicted rewards. MAMQL continues the observed rapid convergent behavior, improving reward alignment by 3-300× compared to baselines, where alignment performance is more pronounced on environments of increased complexity.

In addition to reward recovery, we also evaluate behavioral error, which measures the divergence between the actions taken by the learned policy and those taken by the expert policy. Figure 4 illustrates that MAMQL consistently achieves lower behavioral error across all environments, outperforming the baselines significantly. Specifically, MAMQL demonstrates 20-66× lower behavioral error than the baselines, achieving higher fidelity in policy imitation.

We also observed that MAMQL was able to converge to the optimal policy with minimal hyperparameter modifications across environments: we only had to scale the buffer size to approximately 400× the horizon length of the environment. Furthermore, we observe MAMQL converges with a maximum buffer size of 16.6k steps while other MAIRL algorithms in comparable environments, such as MAAC [8] and MA-DAC [9], were reported to necessitate buffer sizes approximately 3-75× larger.

VII. CONCLUSION

We propose MAMQL, a novel multi-agent inverse reinforcement learning algorithm which addresses the complexities of learning rewards in multi-agent general-sum games. Our algorithm jointly learns reward functions and policies for each agent. Our results show the policies learned by MAMQL yield a significant increase in average reward and a decrease in training time across several environments compared to traditional MAIRL and IL algorithms. We demonstrate applications in autonomous driving and long-horizon tasks, though sim-to-real transfer mandates safety considerations and significant testing before deployment. One direction for future work is to explore expressive architectures further to exceed expert performance on higher-complexity tasks.

Another limitation of our work is experimental: we conducted our experiments using data from “simulated” experts. While our results are a proof of concept across different domains, humans may have biases in their decision making that cannot be captured by generalized Boltzmann policies [40], [41]. Although these simplified computational models are common in inverse reinforcement learning, future work may focus on capturing humans’ biases while learning policies or reward functions from their demonstrations, as it was done by Chan et al. [42] for single-agent systems.

ACKNOWLEDGMENTS

The authors would like to acknowledge Supervised Program for Alignment Research (SPAR) that helped form the team for this project.

REFERENCES

- [1] F. Allegra, “Inverse Reinforcement Learning for Autonomous Driving,” 2020.
- [2] Z. Cao, E. Biyik, W. Z. Wang, A. Raventos, A. Gaidon, G. Rosman, and D. Sadigh, “Reinforcement Learning Based Control of Imitative Policies for Near-Accident Driving,” in *Robotics: Science and Systems*, July 2020.
- [3] S. Dey, T. Marzullo, and G. Henze, “Inverse Reinforcement Learning Control for Building Energy Management,” *Energy and Buildings*, vol. 286, p. 112941, 2023.

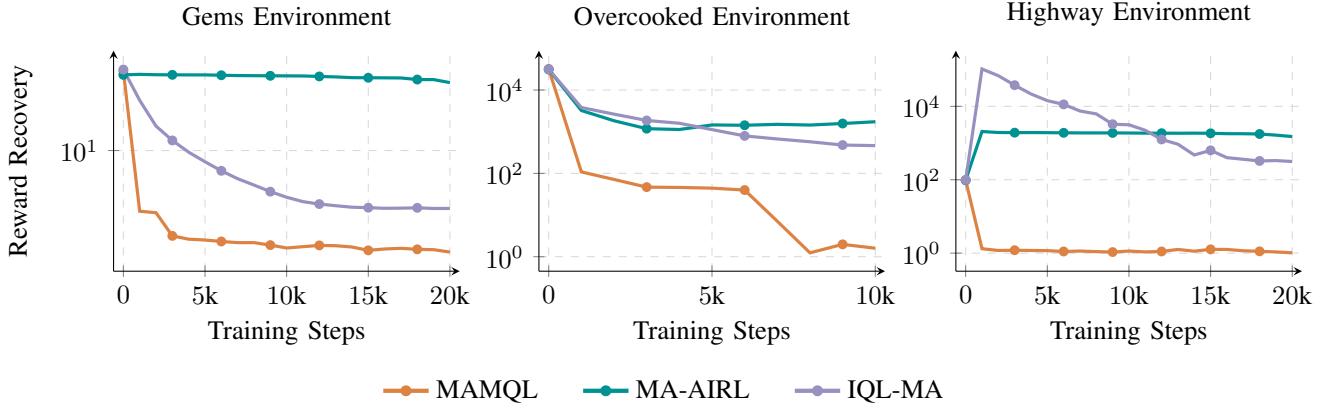


Fig. 3. Reward recovery (MSE between true and predicted rewards) across training trajectories of the MAIRL algorithms.

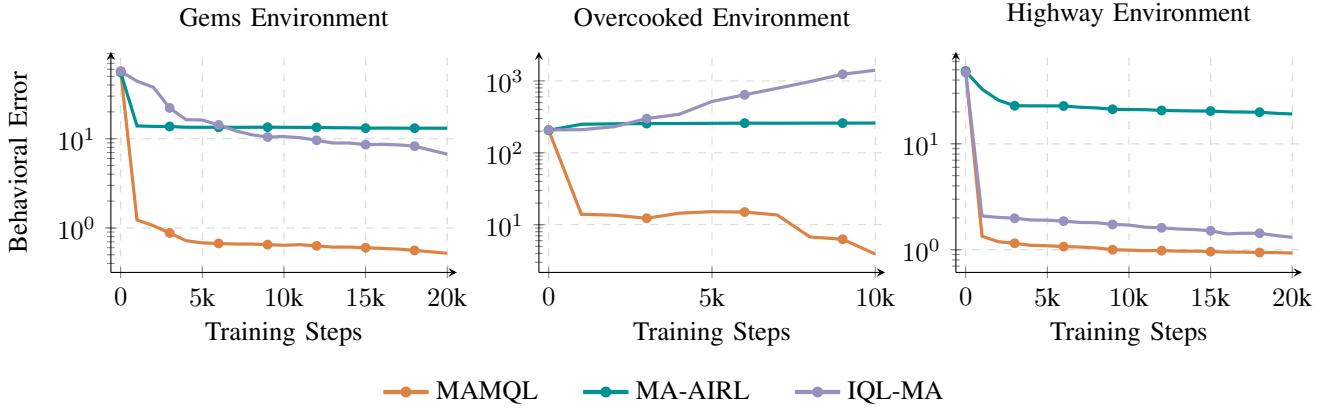


Fig. 4. Behavioral error across trajectories while training the MAIRL algorithms.

- [4] G. Wang, P. Cheng, Z. Chen, W. Xiang, B. Vucetic, and Y. Li, “Inverse Reinforcement Learning With Graph Neural Networks for IoT Resource Allocation,” in *ICASSP 2023-2023 IEEE International Conference on Acoustics*. IEEE, 2023, pp. 1–5.
- [5] X. Chen, J.-Y. Jiang, K. Jin, Y. Zhou, M. Liu, P. J. Brantingham, and W. Wang, “Reliable: Offline Reinforcement Learning for Tactical Strategies in Professional Basketball Games,” in *ACM International Conference on Information & Knowledge Management*, 2022, pp. 3023–3032.
- [6] P. Rahimian and L. Toka, “Inferring the Strategy of Offensive and Defensive Play in Soccer With Inverse Reinforcement Learning,” in *International Workshop on Machine Learning and Data Mining for Sports Analytics*. Springer, 2021, pp. 26–38.
- [7] L. Yu, J. Song, and S. Ermon, “Multi-Agent Adversarial Inverse Reinforcement Learning,” in *International Conference on Machine Learning*. PMLR, May 2019, pp. 7194–7201.
- [8] R. Lowe, YI. WU, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, “Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments,” in *Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.
- [9] K. Xue, J. Xu, L. Yuan, M. Li, C. Qian, Z. Zhang, and Y. Yu, “Multi-Agent Dynamic Algorithm Configuration,” arXiv:2210.06835, 2022.
- [10] M. Carroll, R. Shah, M. K. Ho, T. L. Griffiths, S. A. Seshia, P. Abbeel, and A. Dragan, “On the Utility of Learning About Humans for Human-AI Coordination,” arXiv:1910.05789, 2020.
- [11] E. Leurent, “An Environment for Autonomous Driving Decision-Making,” <https://github.com/eleurent/highway-env>, 2018.
- [12] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, “Maximum Margin Planning,” 2006.
- [13] P. Abbeel and A. Y. Ng, “Apprenticeship Learning via Inverse Reinforcement Learning,” in *International Conference on Machine Learning*, 2004, p. 1.
- [14] A. Y. Ng, S. Russell, et al., “Algorithms for Inverse Reinforcement Learning,” in *ICML*, vol. 1, 2000, p. 2.
- [15] D. Ramachandran and E. Amir, “Bayesian Inverse Reinforcement Learning,” in *IJCAI*, vol. 7, 2007, pp. 2586–2591.
- [16] J. Ho and S. Ermon, “Generative Adversarial Imitation Learning,” in *Neural Information Processing Systems*, vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2016/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html
- [17] K. Waugh, B. D. Ziebart, and J. A. Bagnell, “Computational Rationalization: The Inverse Equilibrium Problem,” arXiv:1308.3506, Aug. 2013.
- [18] E. Biyik, D. P. Losey, M. Palan, N. C. Landolfi, G. Shevchuk, and D. Sadigh, “Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences,” *The International Journal of Robotics Research*, vol. 41, no. 1, pp. 45–67, 2022.
- [19] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, “Maximum Entropy Inverse Reinforcement Learning,” in *AAAI Conference on Artificial Intelligence*, 2008.
- [20] B. Eysenbach and S. Levine, “Maximum Entropy RL (Provably) Solves Some Robust RL Problems,” in *International Conference on Learning Representations*. arXiv, 2022. [Online]. Available: <http://arxiv.org/abs/2103.06257>
- [21] D. Garg, S. Chakraborty, C. Cundy, J. Song, and S. Ermon, “IQ-Learn: Inverse Soft-Q Learning for Imitation,” in *Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 4028–4039.
- [22] D. S. Brown, W. Goo, and S. Niekum, “Better-Than-Demonstrator Imitation Learning via Automatically-Ranked Demonstrations,” in *Conference on Robot Learning*. arXiv, 2019. [Online]. Available: <http://arxiv.org/abs/1907.03976>

- [23] S. Ross, G. Gordon, and D. Bagnell, “A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning,” in *International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.
- [24] S. Sontakke, J. Zhang, S. Arnold, K. Pertsch, E. Büyik, D. Sadigh, C. Finn, and L. Itti, “Roboclip: One demonstration is enough to learn robot policies,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 55\,681–55\,693, 2023.
- [25] X. Zhang, W. Huang, Y. Li, R. Liao, and Z. Zhang, “Imitation Learning From Inconcurrent Multi-Agent Interactions,” in *IEEE Conference on Decision and Control*, 2021, pp. 43–48.
- [26] L. Peters, D. Fridovich-Keil, V. Rubies-Royo, C. J. Tomlin, and C. Stachniss, “Inferring Objectives in Continuous Dynamic Games From Noise-Corrupted Partial State Observations,” arXiv:2106.03611, Aug. 2021.
- [27] A. Shih, A. Sawhney, J. Kondic, S. Ermon, and D. Sadigh, “On the Critical Role of Conventions in Adaptive Human-AI Collaboration,” in *International Conference on Learning Representations*, 2020.
- [28] H. Hu, A. Lerer, A. Peysakhovich, and J. Foerster, ““Other-Play” for Zero-Shot Coordination,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 4399–4410.
- [29] K. Zhang, Z. Yang, and T. Başar, “Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms,” arXiv:1911.10635, 2021.
- [30] F. Kalogiannis, I. Anagnostides, I. Panageas, E.-V. Vlatakis-Gkaragkounis, V. Chatziafratis, and S. Stavroulakis, “Efficiently Computing Nash Equilibria in Adversarial Team Markov Games,” arXiv:2208.02204, Aug. 2022.
- [31] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning With a Stochastic Actor,” arXiv:1801.01290, 2018.
- [32] P. Knott, M. Carroll, S. Devlin, K. Ciosek, K. Hofmann, A. D. Dragan, and R. Shah, “Evaluating the Robustness of Collaborative Agents,” arXiv:2101.05507, 2021.
- [33] J. a. G. Ribeiro, C. Martinho, A. Sardinha, and F. S. Melo, “Assisting Unknown Teammates in Unknown Tasks: Ad Hoc Teamwork Under Partial Observability,” arXiv:2201.03538, 2022.
- [34] R. Zhao, J. Song, Y. Yuan, H. Haifeng, Y. Gao, Y. Wu, Z. Sun, and Y. Wei, “Maximum Entropy Population-Based Training for Zero-Shot Human-AI Coordination,” arXiv:2112.11701, 2022.
- [35] D. Chen, M. Hajidavalloo, Z. Li, K. Chen, Y. Wang, L. Jiang, and Y. Wang, “Deep Multi-Agent Reinforcement Learning for Highway On-Ramp Merging in Mixed Traffic,” arXiv:2105.05701, 2022.
- [36] B. Chen, M. Xu, Z. Liu, L. Li, and D. Zhao, “Delay-Aware Multi-Agent Reinforcement Learning for Cooperative and Competitive Environments,” arXiv:2005.05441, 2020.
- [37] S. N. N, B. Liu, F. Pittaluga, and M. Chandraker, “SMART: Simultaneous Multi-Agent Recurrent Trajectory Prediction,” arXiv:2007.13078, 2020.
- [38] E. Leurent and J. Mercat, “Social Attention for Autonomous Decision-Making in Dense Traffic,” arXiv:1911.12250, 2019.
- [39] W. Jeon, P. Barde, D. Nowrouzezahrai, and J. Pineau, “Scalable Multi-Agent Inverse Reinforcement Learning via Actor-Attention-Critic,” arXiv:2002.10525, 2020.
- [40] C. Laidlaw and A. Dragan, “The Boltzmann Policy Distribution: Accounting for Systematic Suboptimality in Human Models,” in *International Conference on Learning Representations*, 2021.
- [41] M. Kwon, E. Büyik, A. Talati, K. Bhasin, D. P. Losey, and D. Sadigh, “When Humans Aren’t Optimal: Robots That Collaborate With Risk-Aware Humans,” in *ACM/IEEE International Conference on Human-Robot Interaction*, 2020, pp. 43–52.
- [42] L. Chan, A. Critch, and A. Dragan, “Human Irrationality: Both Bad and Good for Reward Inference,” arXiv:2111.06956, 2021.