Inverse Factorized Soft Q-Learning for Cooperative Multi-agent Imitation Learning

The Viet Bui
Singapore Management University, Singapore
theviet.bui.2023@phdcs.smu.edu.sg

Tien Mai
Singapore Management University, Singapore
atmai@smu.edu.sg

Thanh Hong Nguyen
University of Oregon Eugene, Oregon, United States
thanhhng@cs.uoregon.edu

Abstract

This paper concerns imitation learning (IL) in cooperative multi-agent systems. The learning problem under consideration poses several challenges, characterized by high-dimensional state and action spaces and intricate inter-agent dependencies. In a single-agent setting, IL was shown to be done efficiently via an inverse *soft-Q* learning process. However, extending this framework to a multi-agent context introduces the need to simultaneously learn both local value functions to capture local observations and individual actions, and a joint value function for exploiting centralized learning. In this work, we introduce a new multi-agent IL algorithm designed to address these challenges. Our approach enables the centralized learning by leveraging mixing networks to aggregate decentralized Q functions. We further establish conditions for the mixing networks under which the multi-agent IL objective function exhibits convexity within the Q function space. We present extensive experiments conducted on some challenging multi-agent game environments, including an advanced version of the Star-Craft multi-agent challenge (*SMACv2*), which demonstrates the effectiveness of our algorithm.

1 Introduction

Imitation learning (IL) is a powerful approach for sequential decision making in complex environments in which agents can learn desirable behavior by imitating an expert. There are applications of IL in several real-world domains, including healthcare [40, 34] and autonomous driving [43, 17, 23]. A rich body of IL literature focuses on simple single-agent settings [1, 18, 13, 44, 32, 14] while many recent works develop new multi-agent IL methods that are tailored to either cooperative or non-cooperative settings [36, 42, 22, 4, 19, 37]. Leading methods in [36, 42] explore solution concepts for Markov games such as Nash equilibria. Findings on underlying properties of these solution concepts are integrated to extend the single-agent IL models to multi-agent settings. While these methods show promising results, they face difficulties in training, as the underlying adversarial optimization process involves biased and high variance gradient estimators, leading to a highly unstable learning process.

Our work studies IL in cooperative multi-agent settings. We leverage recent advanced findings in both single-agent IL and multi-agent reinforcement learning (MARL) to build a unified multi-agent IL algorithm, named Multi-agent Inverse Factorized Q-learning (MIFQ). MIFQ is built upon inverse soft Q-learning (IQ-Learn) [14], a leading single-agent IL method of which advantage is to learn a single Q-function that implicitly defines both reward and policy functions, thus avoiding adversarial training. To adapt this idea for multi-agent settings, following the well-known paradigm of centralised training with decentralised execution (CTDE) in MARL [26, 21], we develop a centralized learning

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

approach based on the concepts of mixing and hyper-networks [16]. This approach facilitates the integration of individual state-value and reward functions into a unified learning objective, enabling the training of decentralized imitation policies for agents in a centralized manner.

In this work, we formulate the IL problem as a multi-agent inverse soft-Q learning task, where the objective is to match the occupancy distribution of the joint learning policy with that of the expert. To make the learning process practical and facilitate the CTDE paradigm, we propose factorizing both the global Q and V functions using mixing networks with non-negative weights and convex activation functions (such as ReLU and ELU). We further examine the Individual-Global-Max (IGM) principle [29, 35], a core concept in factorization learning, which suggests that the optimal joint actions across agents are equivalent to the collection of individual optimal actions for each agent. However, we argue that this principle does not apply for our problem setting since our objective is to recover soft policies based on max-entropy reinforcement learning. Therefore, we introduce a generalized version of IGM, called Individual-Global-Consistency (IGC), which requires that the distribution of joint actions produced by the optimal joint policy matches the collection of local action distributions produced by the individual optimal policies. We then demonstrate that IGC holds in our approach.

A key advantage of single-agent inverse soft Q-learning (IQ-Learn) is that the original training problem, which is a max-min optimization, can be conveniently converted into a (non-adversarial) concave maximization problem [14]. This conversion ensures that the optimization objective is well-behaved, contributing to the effectiveness of the original IQ-Learn. Interestingly, we provide theoretical results showing that under our factorization approach, the multi-agent training objective can also be converted into a (non-adversarial) concave maximization problem. This advantage holds with any multi-layer feed-forward mixing networks with non-negative weights and convex activations.

Finally, we conduct extensive experiments in three domains: SMACv2 [9], Gold Miner [12], and MPE (Multi Particle Environments) [25]. We show that our MIFQ outperforms other baselines in all these environments. To the best of our knowledge, our experiments with SMACv2 mark the first time IL algorithms are employed and evaluated on such a challenging multi-agent environment.

Concretely, we make the following main contribution:

- (i) We introduce a new multi-agent IL method based on inverse soft-Q learning and factorization.
- (ii) We show that our approach satisfies the IGC, a generation of the IGM principle.
- (iii) We show that the max-min learning objective can be converted into a concave maximization problem, under any mixing networks of non-negative weights and convex activations, which helps avoid adversarial training and ensure well-behaved learning within the Q-space.
- (iv) We empirically show state-of-art results on several challenging multi-agent game tasks, including an advanced version of the Star-Craft multi-agent challenge (*SMACv2*).

2 Related Work

Single-Agent Imitation Learning. There is a rich body of existing works focusing on generating policies that mimic an expert's behavior given data of expert demonstrations in single-agent settings. A classic approach is behavioral cloning which casts IL as a supervised learning problem, attempting to maximize the likelihood of the expert's trajectories [32, 28]. This approach, while simple, requires a large amount of data to work well due to its compounding error issue. An alternative approach is to recover the reward function (either implicitly or explicitly) for which the expert's policy is optimal [13, 18, 30, 20, 10]. Leading methods [18, 13] follow an adversarial optimization process (which is similar to GAN [15]) to train the imitation policy and reward function alternatively. These adversarial training-based methods, however, suffer instability. A more recent work by [14] overcomes this instability issue by introducing an inverse soft-Q learning process. For a comprehensive review of literature on this topic, we refer readers to the survey article in [2].

Multi-Agent Imitation Learning. Most of single-agent IL works, however, do not apply directly to multi-agent settings. Literature on multi-agent IL is rather limited. A few works study multi-agent IL either in cooperative environments [3, 22, 37, 5] or competitive environments [24, 31, 36, 42]. Recent leading methods employ equilibrium solution concepts in Markov games to extend some existing single-agent IL methods to the multi-agent settings [36, 42]. However, these methods still suffer the instability challenge during training as they still rely on adversarial training. Our work focuses on multi-agent IL in a cooperative Dec-POMDP environment. We utilize the idea of inverse soft-Q learning in single-agent IL [14] to avoid adversarial training. We extend the idea to the multi-agent settings under the paradigm of centralized training decentralized execution (CTDE) from MARL, allowing an efficient and stable learning process.

MARL. The literature of MARL encompasses a number of advanced methods; many follow the well-known CTDE paradigm. In [25, 11], they employ actor-critic architectures and train a centralized critic that utilizes global information. Value-decomposition (VD) methods represent the joint Q-function as a function of agents' local Q-functions [38, 29]. QMIX [29] offers a more advanced VD method for consolidating agents' individual local Q-functions through the utilization of mixing and hyper-network [16] concepts. Later value function factorization approaches, such as QTRAN [35] and QPLEX [39], introduce new factorization methods for MARL based on the IGM principle, which necessitates consistency between joint and local action selections. There are also policy gradient based MARL algorithms. Independent PPO (IPPO), a decentralized MARL, can achieve high success rates in several challenging SMAC maps [7]. MAPPO, a PPO version for MARL, achieves SOTA results on several tasks [41]. Our work utilizes MAPPO to train our expert and generate expert demonstrations. We also employ a hyper-network architecture [16] with mixing networks of non-negative weights and convex activations to facilitate our CTDE. While our mixing network techniques are similar to those used in QMIX [29], we show that this configuration offers several *unique advantages* in the context of inverse Q-learning. Specifically, the training objective can be transformed from a max-min problem into a (non-adversarial) concave maximization problem, and it upholds the *IGC principle*. Other methods, such as QTRAN and QPLEX [35, 39], are primarily based on the IGM principle and are designed for different objective structures, making them unsuitable for our context.

3 Preliminaries

A multi-agent cooperative system can be described as a decentralized partially observable Markov decision process (Dec-POMDP), defined by a tuple $\{\mathcal{S}, \mathcal{O}, \mathcal{N}, \mathcal{A}, P, R\}$ [27], where $\mathcal{S}$ is the set of global states, $\mathcal{O}$ is the set of local observations of agents, and $\mathcal{N}$ is all the agents. In addition, $\mathcal{A} = \prod_{i \in \mathcal{N}_{\mathcal{A}}} \mathcal{A}_i$ is the set of joint actions of all the agents, $\mathcal{A}_i$ is the set of actions of an agent $i \in \mathcal{N}$, and P is the transition dynamics of the multi-agent environment. Finally, in Cooperative MARL (Coop-MARL), all agents share the same reward function, R, that can take inputs as global states and actions of all the agents and return the corresponding rewards.

At each step, given a global state S, each ally agent i takes an action $a_i \in \mathcal{A}_i$ based on his policy $\pi_i(a_i \mid o_i)$, where o_i is his local observation. The joint action is defined as $A = \{a_i \mid i \in \mathcal{N}\}$ and the joint policy is defined accordingly: $\Pi(A \mid S) = \prod_{i \in \mathcal{N}} \pi_i(a_i \mid o_i)$. After all the agent actions are executed, the global state is updated to a new state $S' \in \mathcal{S}$ with the transition probability $P(S' \mid A, S)$. The objective of Coop-MARL is to find a joint policy $\Pi(\cdot \mid S) = \prod_i \pi_i(\cdot \mid o_i)$ that maximizes the expected long-term joint reward, formulated as follows:

$$\max_{\Pi} \mathbb{E}_{\Pi}\left[\sum_{t=0}^{\infty} \gamma^t R(A_t, S_t)\right]$$

4 Multi-agent Inverse Q-Learning

In IL, the objective is to recover an expert reward or expert policy from some expert demonstration data. Our new multi-agent IL algorithm is built upon an integration of recent advanced findings in single-agent IL (i.e., the new leading IQ-Learn [14]) and in cooperative MARL. In the following, we first present the main idea of IQ-Learn, with some direct centralized and decentralized adaptations to multi-agent settings. We then discuss key shortcomings of such simple adaptations in the context of a Dec-POMDP. Finally, we present our new algorithm which tackles all those shortcomings.

4.1 Inverse Soft Q-Learning

4.1.1 Centralized Inverse Q-Learning

In general, IQ-Learn can be directly adapted for IL in a *fully-observable* cooperative multi-agent setting. Given expert samples $\mathcal{D}^E = \{\tau = \{(A_t, S_t),\ t = 0, 1...\}\}$, IQ-Learn aims to solve the following maximin problem, which is also the objective of adversarial learning-based IL approaches [18, 13]:

$$\max_R \min_\Pi \Big\{ L(R,\Pi) = \mathbb{E}_{(S,A)\sim\rho^E}\big[R(S,A)\big] - \mathbb{E}_{\rho^\Pi}\big[R(S,A)\big] - \mathbb{E}_{\rho^\Pi}\big[\ln \Pi(S,A)\big] \Big\} \quad (1)$$

where $\rho^\Pi(S,A)$ is the occupancy measure of visiting state S and joint action A, under the policy Π:

$$\rho^\Pi(S,A) = (1-\gamma)\Pi(A \mid S) \prod_{t=0}^{\infty} \gamma^t P(S_t = S \mid \Pi),$$

ρ^E is the occupancy measure of the expert policy π^E and $\mathbb{E}_{\rho^\Pi}\big[\ln \Pi(S,A)\big]$ is the entropy regularizer.

A typical method to solve (1) is to run an adversarial optimization process over rewards and policies, of which idea is similar to generative adversarial networks [18, 13]. However, such an approach suffers instability, a well-known challenge of adversarial training. Thus, IQ-Learn avoids adversarial training by learning a single soft Q-function, as defined in the following.

Definition 4.1 (Soft Q-function [14])**.** *Given a policy* Π*, the soft Bellman operator* $\mathcal{B}^\Pi$ *is defined as:*

$$(\mathcal{B}^\Pi Q)(S,A) = R(S,A) + \gamma \mathbb{E}_{S'\sim P(\cdot|S,A)} V^{\Pi,Q}(S')$$
$$\textit{where } V^{\Pi,Q}(S) = \mathbb{E}_{A\sim\Pi(\cdot|S)}\big[Q(S,A) - \log \Pi(A \mid S)\big] \quad (2)$$

$\mathcal{B}^\Pi$ *is contractive and defines a unique soft Q-function for the reward function* R*, i.e.,* $Q = \mathcal{B}^\Pi Q$*.*

Essentially, in [14], they show that (1) is equivalent to the following single minimization problem which only requires optimizing over the soft Q-function: [1]

$$\max_Q \Big\{ J(Q) = \mathbb{E}_{(S,A)\sim\rho^E}\Big[Q(S,A) - \gamma \mathbb{E}_{S'\sim P(\cdot|S,A)}\big[V^Q(S')\big]\Big] - (1-\gamma)\mathbb{E}_{S^0}\big[V^Q(S^0)\big] \Big\} \quad (3)$$

where $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the soft Q-function, S^0 is the initial state, and $V^Q(S)$ is computed as $V^Q(S) = \log(\sum_{A\in\mathcal{A}} \exp(Q(S,A)))$. A shortcoming of centralized IQ-Learn described above is that the computation of $V^{\Pi,Q}(S)$ or $V^Q(S)$ is **not tractable** as it requires to sample joint actions, an the joint action space grows exponentially in the number of agents. Furthermore, it requires full observations for agents, which is not applicable in a Dec-POMDP with local partial observations.

4.1.2 Independent Inverse Q-Learning

An alternative approach to overcome the shortcomings of centralized IQ-Learn is to consider a separate IL problem for each individual agent, considering its local observations. That is, one can set the objective to recover a local Q function $Q_i(o_i, a_i)$, as a function of a local observation o_i and local action a_i, for each agent $i \in \mathcal{N}$. The local IQ-Learn loss function can be formulated as follows:

$$\max_{Q_i} \min_{\pi_i} \Big\{ J_i(Q_i, \pi_i) = \mathbb{E}_{\mathcal{D}^E}\Big[Q_i(o_i,a_i) - \gamma \mathbb{E}_{o'_i\sim(P(\cdot|)S,A)}\big[V_i^{Q,\pi}(o'_i)\big]\Big] - (1-\gamma)\mathbb{E}_{o_i^0}\big[V_i^{Q,\pi}(o_i^0)\big] \Big\} \quad (4)$$

Here, $o'_i \sim P(\cdot \mid S,A)$ means o'_i is the local observation of agent i corresponding to the new state $S' \sim P(\cdot \mid S,A)$. The value function can be computed as [14]:

$$V_i^{Q,\pi}(o_i) = \mathbb{E}_{\pi_i(a_i|o_i)}\left[Q_i(o_i,a_i) - \log \pi_i(a_i|o_i)\right] \quad (5)$$

This approach is more tractable than the centralized method, making it suitable for a Dec-POMDP environment by enabling decentralized policies for agents. However, it has limitations in addressing the interdependence between agents and the global information available during the training process.

4.2 Inverse Factorized Soft Q-Learning

We now present our new multi-agent IL algorithm designed to recover Q functions in the multi-agent setting, following the CTDE paradigm. A possible straightforward approach is to directly factorize the centralized objective function in 3. This factorization can be achieved by decomposing the global Q-function $Q(S,A)$ into local Q-functions $Q_i(o_i,a_i)$, and computing the global V-functions using the formula $V^Q(S) = \log(\sum_{A\in\mathcal{A}} \exp(Q(S,A)))$. However, this means that the global policy

[1]We convert the maximization formulation in [14] into minimization to facilitate our later theoretical analyses.

has to be computed based on a softmax of the global Q-function as $\Pi(S,A) = \frac{\exp(Q(S,A))}{\sum_{A'}\exp(Q(S,A'))}$. This requirement of a global policy computation violates the decentralized execution under CTDE. According to this analysis, we will instead start from the original *max-min* formulation in (1), keeping in mind the necessity of having π_i to support CTDE.

Our key ideas involve creating (i) agent local Q-value networks that output local V-values $V_i^{Q,\pi}(o_i, a_i)$ and (ii) mixing the networks that utilize global state information to combine local values of agents into joint values that comprise the objective of the inverse soft Q-learning; and (iii) hyper-networks that provide a rich representation for the weights of the mixing networks, allowing us to govern their value ranges. There are two important aspects driving our mixing architecture: the consistency between local and global policies, and whether we can leverage the advantages of single-agent IQ-learn [14], specifically non-adversarial training and convexity within the Q-space. We will first describe our factorization approach, followed by an analysis of these two aspects.

4.2.1 Multi-agent IL Network Architecture

Overall, our network architecture comprises of three different types of networks, described below.

Agent local Q networks. Let $\mathbf{Q}(S,A) = \{Q_i(o_i,a_i)\}_{i=1}^m$ of local soft Q-values of agents where $m = |\mathcal{N}|$ is the number of agents and $(o_i, a_i) \in (S, A)$, $i \in \mathcal{N}$. For an abuse of notations, we denote by $Q_i(o_i, a_i; \theta_i)$ as the local Q-value network of the agent i of which learnable parameter is θ_i.[2] Given (Q, Π), we then use $\mathbf{V}^{Q,\Pi}(S,A)$ to represent the corresponding state-value vector $\mathbf{V}^{Q,\Pi}(S) = \{V_i^{Q,\pi}(o_i)\}_{i=1}^m$ where $V_i^{Q,\pi}(o_i)$ is computed in (5). $\mathbf{Q}$ and $\mathbf{V}^{Q,\Pi}$ will be passed to the corresponding mixing networks to induce the joint values Q^{tot} and $V^{tot}_{Q,\Pi}$. These joint values will be then incorporated into computing the objective of the inverse soft Q-learning. Here, we do not assume any specific model for the policies π_i. As we show later, there exists a closed-form solution to compute optimal policies, which allows us to eliminate the variable π_i from the training problem.

Value mixing networks. We create two mixing networks to combine local Q and V values into the joint values Q^{tot} and V^{tot}, respectively. Let's denote these networks as $\mathcal{M}_{\psi_Q}(\cdot)$ and $\mathcal{M}_{\psi_V}(\cdot)$. Here ψ_Q and ψ_V are corresponding weights of these two networks. In particular, we have:

$$Q^{tot}(S,A) = -\mathcal{M}_{\psi_Q}\big(-\mathbf{Q}(S,A)\big) \qquad V^{tot}_{Q,\Pi}(S) = \mathcal{M}_{\psi_V}\big(\mathbf{V}^{Q,\Pi}(S)\big)$$

Note that, instead of directly mixing $\mathbf{Q}(S)$, we mix the negative of this vector to achieve the **IGC** principle and the convexity. We can now formulate the objective function of our *multi-agent inverse factorized Q-learning* w.r.t the local Q-values and polices, and these mixing networks, as follows:

$$\max_{\mathbf{Q}} \min_{\Pi} \Big\{ J(\mathbf{Q}, \Pi, \psi_Q, \psi_V) = \sum\nolimits_{(S,A)\in\mathcal{D}^E} \Big[Q^{tot}(S,A) - \gamma \mathbb{E}_{S'\sim P(\cdot|S,A)}\big[V^{tot}_{Q,\Pi}(S')\big]\Big] - (1-\gamma)\mathbb{E}_{S^0}\big[V^{tot}_{Q,\Pi}(S_0)\big]\Big\} \tag{6}$$

We further assume that *the two mixing networks are multi-layer feed-forward networks, constructed with non-negative weights and convex activation functions* (e.g., *ReLU*, *ELU*, and *Maxout*). This configuration is widely-used in the literature and, as shown in the next section, is sufficient to ensure consistency between the global and local policies (a form of the IGM property, as shown later) and, importantly, the convexity of the training objective function within the Q-space.

Hyper-networks. Finally, we create two hyper-networks corresponding the two mixing networks. These hyper-networks take the global state S as an input and generate the weights ψ_V and ψ_Q of the mixing networks accordingly. The creation of such hyper-networks allows us to have a rich representation of the weights that can be governed to ensure the convexity of the objective $J(\mathbf{Q}, \Pi, \psi_Q, \psi_V)$ as well as the IGC property (as described below) . We can write $\psi_V = \psi_V(S; \omega_V)$ and $\psi_Q = \psi_Q(S; \omega_R)$ where ω_V and ω_R denote trainable parameters of the state-value and reward hyper-networks. We can alternatively write the objective $J(\mathbf{Q}, \Pi, \psi_Q, \psi_V)$ as $J(\theta, \omega_R, \omega_V)$ when the local soft Q-values $\mathbf{Q}$ are parameterized by $\theta = \{\theta_1, \theta_2, \cdots, \theta_m\}$ and the weights of the mixing networks (ψ_V, ψ_Q) are parameterized by ω_V and ω_R respectively. The optimal policy, as shown later, can be computed as soft-max of $\mathbf{Q}$. We note that we will use either $J(\mathbf{Q}, \Pi, \psi_Q, \psi_V)$ or $J(\theta, \omega_R, \omega_V)$ depending on the context of our analysis.

[2]In practical implementation, we consider a shared local Q-value network for all agents.

4.2.2 Individual-Global-Consistency (IGC)

One of the key aspects of CTDE is ensuring consistency between global learning and local policy execution. In previous MARL approaches based on mixing Q-functions, this is often expressed through the IGM principle [35, 29, 39], implying that the optimal joint actions across agents (obtained by maximizing the joint Q-function) are equivalent to the collection of individual optimal actions of each agent (obtained by maximizing their local Q-functions). In our context, there are significant differences that prevent direct application of this principle. First, our recovered policy is not derived from directly maximizing the Q-function. Instead, it is obtained by solving a distribution-matching objective, which results in soft policies computed as the soft-max of Q-values. To adapt this principle in the context of inverse Q-learning, we introduce the concept of *Individual-Global-Consistency (IGC)*, which is a generalization of the IGM, defined in the following.

Definition 4.2 (Individual-Global-Consistency (IGC)). *A factorized learning approach is said to adhere to the IGC principle if and only if the optimal joint policy (obtained from solving the global training problem) is equivalent to the collection of individual optimal policies for each agent (obtained by solving their respective local training objectives).*

The following result states that our factorized inverse Q-learning approach satisfies the IGC principle.

Theorem 4.3. *Let $\Pi^* = argmin_{\Pi}\{J(\boldsymbol{Q}, \Pi, \psi_Q, \psi_V)\}$, then there are a set of optimal local policies $\{\pi_i^*\}$ such that $\pi_i^* = argmin_{\pi_i} J_i(Q_i, \pi_i),\ \forall i \in \mathcal{N}$, and $\Pi^* = \{\pi_i^*,\ i \in \mathcal{N}\}$.*

The theorem implies that the distribution of joint actions, produced by the optimal joint policy, is the same as the collection of local action distributions produced by the local optimal policies. Moreover, as an additional note, the *IGM principle* with respect to the joint Q-function also holds under our mixing network architecture, i.e., for all $S \in \mathcal{S}$:

$$\text{argmax}_A\{Q^{tot}(S, A)\} = \{\text{argmax}_{a_i}\{Q_i(o_i, a_i)\},\ i \in \mathcal{N}\}.$$

4.2.3 Non-adversarial Training and Convexity

One of the main advantages of the single-agent inverse Q-learning algorithm is that the training problem can be equivalently transformed into a concave maximization problem over the Q-space, making the training process highly stable and well-behaved. We demonstrate below that these features still hold under our mixing network architecture.

Proposition 4.4. *The max-min problem in* (6) *is equivalent to the following maximization problem:*

$$\max_{\boldsymbol{Q}}\Big\{J(\boldsymbol{Q}, \psi_Q, \psi_V) = \sum\nolimits_{(S,A)\in\mathcal{D}^E}\Big[Q^{tot}(S, A) - \gamma\mathbb{E}_{S'\sim P(\cdot|S,A)}\big[V_Q^{tot}(S')\big]\Big]$$
$$- (1-\gamma)\mathbb{E}_{S^0}\big[V_Q^{tot}(S_0)\big]\Big\} \tag{7}$$

where $V_Q^{tot}(S) = \mathcal{M}_{\psi_V}\big(\boldsymbol{V}^Q(S)\big)$, and $\boldsymbol{V}^Q(S) = \Big(V_i^Q(o_i) \overset{def}{=} \log(\sum_{a_i}\exp(Q_i(o_i, a_i))),\ \ i \in \mathcal{N}\Big)$. Moreover, let $\boldsymbol{Q}^*$ be optimal to (7)*, then the global optimal policy can be recovered as follows:*

$$\Pi^* = \left\{\pi_i^*(o_i, a_i) = \frac{\exp(Q_i(o_i, a_i)))}{\sum_{a_i'}\exp(Q_i(o_i, a_i'))}\,\middle|\, i \in \mathcal{N}\right\} \tag{8}$$

Furthermore, it can be shown that the objective in (7) is concave in $\mathbf{Q}$ which is an essential property that make the Q-learning procedure well-behaved and stable [14].

Theorem 4.5. *$J(\boldsymbol{Q}, \psi_Q, \psi_V)$ is (strictly) concave in $\boldsymbol{Q}$. As a result,* (7) *always yield a unique solution within the Q-space.*

Theorem (4.5) indicates that the global training objective $J(\mathbf{Q}, \psi_Q, \psi_V)$ is concave in $\mathbf{Q}$ when using *any multi-layer feed-forward mixing networks with non-negative weights and convex activation functions*. This result is highly general and notably non-trivial due to the nonlinearity and complexity of V_i^Q (as functions of Q_i) and the mixing networks. Prior work often relies on a two-layer mixing structure [29], noting that such a two-layer structure is sufficient for the mixing network to approximate any monotonic function arbitrarily closely in the limit of infinite width [8].

4.3 MIFQ Algorithm

We now present our practical algorithm, MIFQ, and details of our implemented network architecture.

Mixing and Hyper Networks. We employ the following two-layer feed-forward network structure:

$$\mathcal{M}_{\psi_Q}(\mathbf{X}) = \text{ELU}(\mathbf{X} \times |W_1^Q| + b_1^Q) \times |W_2^Q| + b_2^Q \tag{9}$$
$$\mathcal{M}_{\psi_V}(\mathbf{X}) = \text{ELU}(\mathbf{X} \times |W_1^V| + b_1^V) \times |W_2^V| + b_2^V \tag{10}$$

for mixing networks, where $\psi_Q = \{W_1^Q, W_2^Q, b_1^Q, b_2^Q,\}$ and $\psi_V = \{W_1^V, W_2^V, b_1^V, b_2^V\}$ are the weight and bias vectors. The absolute operations $|\cdot|$ ensure that all the weights are non-negative. Moreover, ψ_Q and ψ_V are an output of a *hyper-network* taking the global state S as input. Each hyper-network consists of two fully-connected layers with a ReLU activation. Finally, ELU is employed to mitigate the issue of gradient vanishing and to ensure that negative inputs remain negative. Indeed, ELU is convex and the two mixing networks $\mathcal{M}_{\psi_Q}$ and $\mathcal{M}_{\psi_V}$ have non-negative weights, implying that the loss function $J(\mathbf{Q}, \psi_Q, \psi_V)$ is *concave* in $\mathbf{Q}$ and the ***IGC*** holds.

Practical Implementation. Similar to [14], we use a χ^2-regularizer $\phi(x) = x + \frac{1}{2}x^2$ for the first terms of the loss function in (6). This convex regularizer is useful to ensure that this is lower-bounded even when Q_i, for some $i \in \mathcal{N}$, go to $-\infty$, which is crucial to keep the learning process stable. In addition, instead of directly estimating $\mathbb{E}_{S_0}[V^{tot}(S_0)]$, we utilize the following equation to approximate $\mathbb{E}_{S_0}[V_Q^{tot}(S_0)]$ which can stabilize training:

$$(1-\gamma)\mathbb{E}[V_Q^{tot}(S)] = \mathbb{E}_{(S,A)\sim\rho}\big[V_Q^{tot}(S) - \gamma\mathbb{E}_{S'\sim P(\cdot|S,A)}[V_Q^{tot}(S')]\big]$$

for any value function $V_Q^{tot}(\cdot)$ and occupancy measure ρ [14], we can estimate $\mathbb{E}_{S_0}[V_Q^{tot}(S_0)]$ by sampling (S, A) from replay buffer and estimate $\mathbb{E}_{(S,A,S')\sim\text{replay}}[V_Q^{tot}(S) - \gamma V_Q^{tot}(S')]$ instead. In summary, we will employ the following practical loss function:

$$\min_{\theta,\omega_R,\omega_V} \Big\{ J(\theta, \omega_R, \omega_V) = \sum\nolimits_{(S,A)\in\mathcal{D}^E} \phi\big(Q^{tot}(S,A) - \gamma V_Q^{tot}(S')\big) \\ + \mathbb{E}_{(S,A,S')\in\mathcal{D}^{replay}}\big[V_Q^{tot}(S) - \gamma V_Q^{tot}(S')\big]\Big\}. \tag{11}$$

Fig. 1 shows our network architecture and the details of our MIFQ can be found in the appendix.

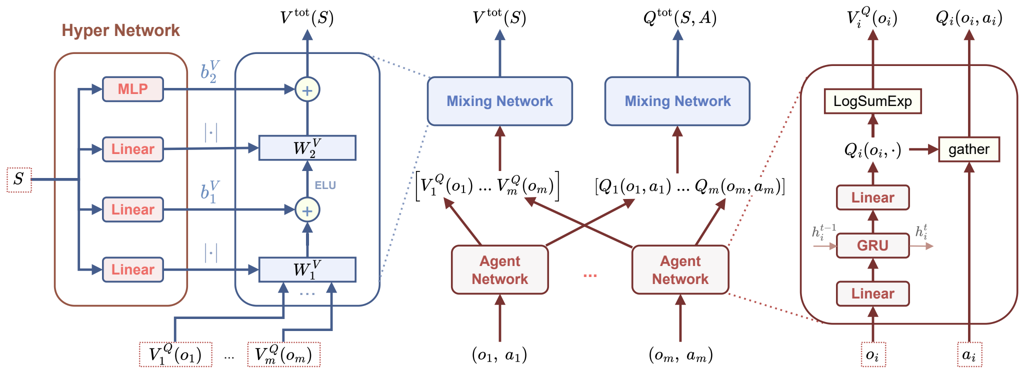


Figure 1: An overview of our network architecture.

5 Experiments

Environments. We use three environments in which enemy agents' policies are fixed and controlled by a simulator. We collect expert trajectories from well-trained ally agents and build IL to mimic them. The resulting imitating agents are evaluated by letting them play against the simulator's enemies.

SMACv2 [9]. SMAC is a well-known multi-agent environment built based on StarCraft II. We employ SMACv2 [9], an enhanced version of SMACv1 [33], that introduces a more formidable environment

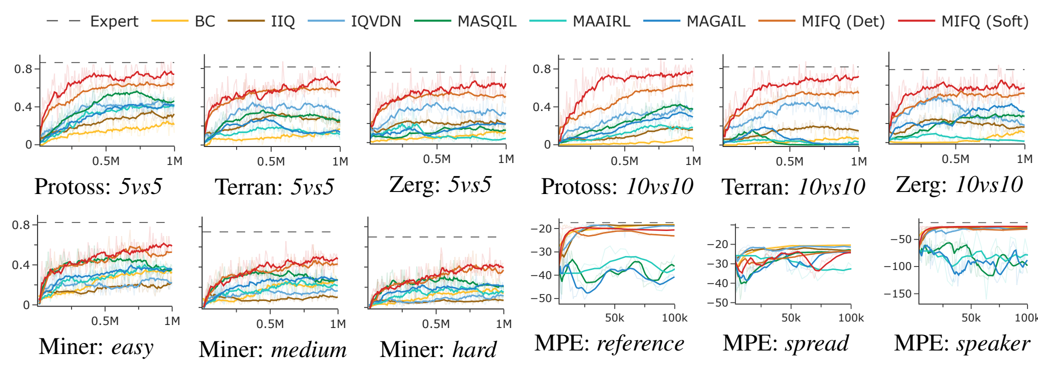


Figure 2: Learning curves

Table 1: Winrate and reward comparisons

Scenarios		Expert	BC	IIQ	IQ-VDN	MA-SQIL	MA-AIRL	MA-GAIL	**MIFQ** (ours) Det	Soft
Protoss	*5vs5*	86.7%	19.5%	33.6%	39.8%	47.7%	42.6%	42.6%	64.8%	**72.7%**
	10vs10	90.3%	5.5%	16.6%	38.3%	36.8%	19.8%	28.3%	61.7%	**77.3%**
Terran	*5vs5*	81.7%	18.0%	19.7%	32.0%	24.6%	10.9%	10.9%	55.9%	**71.9%**
	10vs10	81.7%	6.2%	14.1%	35.9%	0.5%	2.3%	1.0%	53.8%	**72.7%**
Zerg	*5vs5*	73.5%	10.9%	18.6%	33.6%	14.8%	5.3%	18.8%	46.7%	**60.9%**
	10vs10	76.3%	9.4%	16.4%	17.2%	27.1%	1.2%	31.2%	51.0%	**59.4%**
Miner	*easy*	82.4%	31.2%	21.9%	18.8%	36.7%	35.2%	34.6%	52.7%	**60.2%**
	medium	74.9%	28.1%	9.8%	14.8%	28.3%	21.1%	26.0%	43.8%	**49.2%**
	hard	69.8%	12.5%	6.6%	11.7%	19.9%	17.2%	20.9%	39.1%	**39.8%**
MPE	*reference*	-17.3	**-18.3**	-18.6	-19.0	-36.6	-40.7	-40.0	-23.4	-20.5
	spread	-11.1	**-20.5**	-23.6	-21.6	-23.0	-26.5	-24.3	-23.2	-24.2
	speaker	-19.4	-27.5	-29.1	-29.5	-104.3	-125.5	-78.7	-30.8	**-26.3**

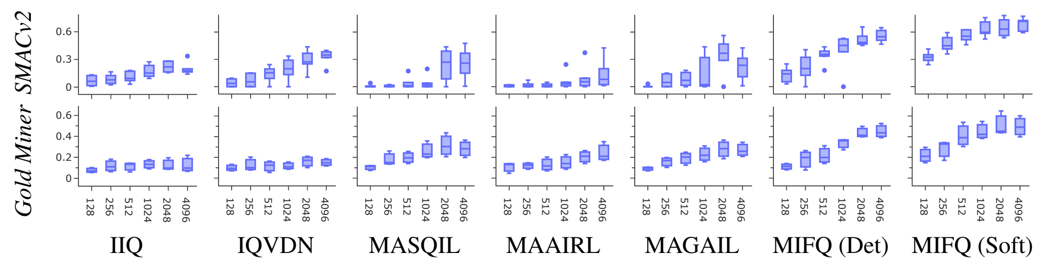


Figure 3: Comparison with different numbers of demonstrations. X-axis: winning rate. Y-axis: number of demonstrations.

for evaluating MARL algorithms. In SMACv2, scenarios are procedurally generated, compelling agents to adapt to previously un-encountered situations. This benchmark has 6 sub-tasks, including *Protoss*, *Terran*, and *Zerg* which feature 5 to 10 agents. These agents have the flexibility to engage with opponents of differing difficulty levels.

Gold Miner [12]. This game originates from a MARL competition, in which two teams, ally and enemy, navigate through a 2D terrain to find gold. A team win if they mined a larger amount of gold than the other. Winning this game is challenging since the allied agents must compete against exceptionally well-developed heuristic-based enemies. We consider three sub-tasks, each involves two ally and two enemy agents: (i) *Easy*: The enemies employ a simple shortest-path strategy to reach the gold deposits; (ii) *Medium*: One enemy agent follows a greedy approach, while the other follows the algorithm developed by the second-ranking team in the competition; and (iii) *Hard*: the enemies consists of the first- and second-ranking teams.

Multi Particle Environments (MPE) [25]. MPE contains multiple communication-oriented deterministic multi-agent environments. We use three cooperative scenarios available in MPE for evaluating, including: (i) *Simple spread:* three agents learn to avoid collisions while covering all of the landmarks; (ii) *Simple reference:* two agents learn to get closer to the target landmarks. Each target landmark is known only by the other agents, so all agents have to communicate to each others; and (iii) *Simple speaker listener:* similar to simple reference, but one agent (speaker) can speak but cannot move, and one agent (listener) can move but cannot speak.

Expert Demonstrations. For each task, we trained an expert policy by MAPPO with large-scale hyper-parameters (e.g., multi layers, higher dimensions, longer training steps, and using recurrent neural network, etc). In term of expert buffer collection, we test each method with different numbers of expert trajectories: up to 128 trajectories for MPEs and up to 4096 trajectories for Miner and SMAC-v2. Note that MPEs are not dynamic environments, so we do not need a large number of expert demonstrations for evaluation. For each collected trajectory, we used a different random seed. For a fair comparison, each method uses the same saved expert demonstrations for the training.

Baselines. We compare our MIFQ against other multi-agent IL algorithms, which either originate from the multi-agent IL literature, or be adapted from SOTA single-agent IL algorithms. These baselines include: (i) *Behavior Cloning (BC)*; (ii) *Independent IQ-Learn (IIQ)* — this is a simple adaption of IQ-Learn for a multi-agent setting, described in Section 4.1.2; (iii) *IQ-Learn with Value Decomposition Network (IQVDN)* — this is another adaptation of IQ-Learn, but instead of using mixing and hyper networks to aggregate agent Q functions, we employ Value Decomposition (VDN) [38]; (iv) *MASQIL* — we adapt SQIL for multi-agent settings [30]. MASQIL shares some similar advantages with our MIFQ, such as being non-adversarial and enabling decentralized learning through centralized learning; (v) *MAGAIL* — a multi-agent IL algorithm introduced by [36]; and finally (vi) *MAAIRL* — this algorithm is proposed by [42]. Moreover, since our MIFQ algorithm relies on soft policies while most previous factorized Q-learning algorithms [29, 35, 39] use deterministic policies, we include a deterministic-policy version of MIFQ for comparison purposes. In this version, the deterministic optimal local actions are determined as $a_i^* = \text{argmax}_{a_i}\{Q_i(o_i, a_i)\}$, and the local V functions are $V_i^Q(o_i) = \max_{a_i} Q_i(o_i, a_i)$. We denote our main MIFQ algorithm as **MIFQ (Soft)**, and the deterministic-policy version as **MIFQ (Det)**.

Comparison Results. We train each task with our algorithms and the different baselines, using 128 trajectories for MPE, and 4096 trajectories for SMACv2 and Miner. As a standard practice, we report winrates for SMACv2 & Gold Miner, and reward scores for MPEs. Figure 2 shows the learning curves and Table 1 reports the final winrates and average scores (the full table with std values can be found in appendix). Our methods MIFQ (*Soft*) significantly outperforms all other baselines on SMACv2 & Gold Miner tasks, and has competitive performance on MPEs, with a note that MPEs is a deterministic environment and its tasks are considerably easier than SMACv2 and Gold Miner. We also observe that MIFQ (*Det*) significantly outperforms other baselines, but remarkably worse than MIFQ (*Soft*), indicating the advantage of using soft-policy learning in our context. Moreover, to evaluate the efficiency of our method with different numbers of expert demonstrations, we compare our MIFQ with the baselines on two dynamic tasks: *SMACv2 & Miner*. Figure 3 shows box and whisker plots of the average winning rate of each method on each task for data summarising analysis. It shows that MIFQ (*Det* and *Soft*) offer higher winrates, and it converges better with smaller standard errors. Moreover, to evaluate how the convexity help enhance our algorithm, we compares MIFQ with those that rely on non-convex mixing networks, which are obtained by replacing the convex activation ELU with non-convex ones such as *sigmoid* or *tanh*. More details can be found in the appendix.

6 Conclusion and Limitations

Conclusion. We developed a multi-agent IL algorithm based on the inversion of soft-Q functions. By employing mixing and hyper-network architectures, our algorithm, MIFQ, is non-adversarial and enables the CTDE approach. We demonstrated that, with some commonly used two-layer mixing network structures, our IL loss function is convex within the Q-function space, making learning convenient. Extensive experiments conducted across several challenging multi-agent tasks demonstrate the superiority of our algorithm compared to existing IL approaches.

Limitations. Some limitations of the work includes: (i) while MIFQ achieves impressive performance across various tasks, it falls short of reaching the expertise levels (increasing the number of expert demonstrations could solve the issue, but it would introduce additional computational challenges, as the replay buffer will become too large and the training process will be very costly); (ii) MIFQ, along with other baseline methods, struggles when confronted with very large-scale tasks, such as some of the largest games in SMACv2; and (iii) our multi-agent IL (and other baselines) still requires a large amount of (clean) expert demonstrations, which would be not always available in practice. These limitations could pave the way for future work.

Broader Impacts. Our research focuses on imitation learning in multi-agent systems, it may have potential applications similar to areas where imitation learning has been impactful, such as autonomous driving, healthcare, and game theory. There are also potential negative impacts. For instance, imitation learning could be used for surveillance purposes, following and monitoring individuals in public spaces, or for developing autonomous weapons.

Acknowledgments

This research/project is supported by the National Research Foundation Singapore and DSO National Laboratories under the AI Singapore Programme (AISG Award No: AISG2-RP-2020-017).

References

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

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

[3] Barrett, S., Rosenfeld, A., Kraus, S., and Stone, P. Making friends on the fly: Cooperating with new teammates. *Artificial Intelligence*, 242:132–171, 2017.

[4] Bhattacharyya, R. P., Phillips, D. J., Wulfe, B., Morton, J., Kuefler, A., and Kochenderfer, M. J. Multi-agent imitation learning for driving simulation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1534–1539. IEEE, 2018.

[5] Bogert, K. and Doshi, P. 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, 2014.

[6] Boyd, S. P. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

[7] de Witt, C. S., Gupta, T., Makoviichuk, D., Makoviychuk, V., Torr, P. H., Sun, M., and Whiteson, S. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

[8] Dugas, C., Bengio, Y., Bélisle, F., Nadeau, C., and Garcia, R. Incorporating functional knowledge in neural networks. *Journal of Machine Learning Research*, 10(6), 2009.

[9] Ellis, B., Moalla, S., Samvelyan, M., Sun, M., Mahajan, A., Foerster, J. N., and Whiteson, S. Smacv2: An improved benchmark for cooperative multi-agent reinforcement learning. *arXiv preprint arXiv:2212.07489*, 2022.

[10] Finn, C., Christiano, P., Abbeel, P., and Levine, S. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*, 2016.

[11] Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[12] FPT. Fpt reinforcement learning competition. `https://github.com/xphongvn/rlcomp2020`, 2020.

[13] Fu, J., Luo, K., and Levine, S. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[14] Garg, D., Chakraborty, S., Cundy, C., Song, J., and Ermon, S. Iq-learn: Inverse soft-q learning for imitation. *Advances in Neural Information Processing Systems*, 34:4028–4039, 2021.

[15] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[16] Ha, D., Dai, A. M., and Le, Q. V. Hypernetworks. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=rkpACe1lx`.

[17] Hawke, J., Shen, R., Gurau, C., Sharma, S., Reda, D., Nikolov, N., Mazur, P., Micklethwaite, S., Griffiths, N., Shah, A., et al. Urban driving with conditional imitation learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 251–257. IEEE, 2020.

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

[19] Jeon, W., Barde, P., Nowrouzezahrai, D., and Pineau, J. Scalable and sample-efficient multi-agent imitation learning. In *Proceedings of the Workshop on Artificial Intelligence Safety, co-located with 34th AAAI Conference on Artificial Intelligence, SafeAI@ AAAI*, 2020.

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

[21] Kraemer, L. and Banerjee, B. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 2016.

[22] Le, H. M., Yue, Y., Carr, P., and Lucey, P. Coordinated multi-agent imitation learning. In *International Conference on Machine Learning*, pp. 1995–2003. PMLR, 2017.

[23] Le Mero, L., Yi, D., Dianati, M., and Mouzakitis, A. A survey on imitation learning techniques for end-to-end autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14128–14147, 2022.

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

[25] Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

[26] Oliehoek, F. A., Spaan, M. T., and Vlassis, N. Optimal and approximate q-value functions for decentralized pomdps. *Journal of Artificial Intelligence Research*, 32:289–353, 2008.

[27] Oliehoek, F. A., Amato, C., et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

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

[29] Rashid, T., Samvelyan, M., De Witt, C. S., Farquhar, G., Foerster, J., and Whiteson, S. Monotonic value function factorisation for deep multi-agent reinforcement learning. *The Journal of Machine Learning Research*, 21(1):7234–7284, 2020.

[30] Reddy, S., Dragan, A. D., and Levine, S. Sqil: Imitation learning via reinforcement learning with sparse rewards. *arXiv preprint arXiv:1905.11108*, 2019.

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

[32] Ross, S. and Bagnell, D. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 661–668. JMLR Workshop and Conference Proceedings, 2010.

[33] Samvelyan, M., Rashid, T., De Witt, C. S., Farquhar, G., Nardelli, N., Rudner, T. G., Hung, C.-M., Torr, P. H., Foerster, J., and Whiteson, S. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*, 2019.

[34] Shah, S. I. H., Coronato, A., Naeem, M., and De Pietro, G. Learning and assessing optimal dynamic treatment regimes through cooperative imitation learning. *IEEE Access*, 10:78148–78158, 2022.

[35] Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, pp. 5887–5896. PMLR, 2019.

[36] Song, J., Ren, H., Sadigh, D., and Ermon, S. Multi-agent generative adversarial imitation learning. *Advances in neural information processing systems*, 31, 2018.

[37] Šošić, A., KhudaBukhsh, W. R., Zoubir, A. M., and Koeppl, H. Inverse reinforcement learning in swarm systems. *arXiv preprint arXiv:1602.05450*, 2016.

[38] Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*, 2017.

[39] Wang, J., Ren, Z., Liu, T., Yu, Y., and Zhang, C. Qplex: Duplex dueling multi-agent q-learning. *arXiv preprint arXiv:2008.01062*, 2020.

[40] Wang, L., Yu, W., He, X., Cheng, W., Ren, M. R., Wang, W., Zong, B., Chen, H., and Zha, H. Adversarial cooperative imitation learning for dynamic treatment regimes. In *Proceedings of The Web Conference 2020*, pp. 1785–1795, 2020.

[41] Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., and Wu, Y. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624, 2022.

[42] Yu, L., Song, J., and Ermon, S. Multi-agent adversarial inverse reinforcement learning. In *International Conference on Machine Learning*, pp. 7194–7201. PMLR, 2019.

[43] Zhou, J., Wang, R., Liu, X., Jiang, Y., Jiang, S., Tao, J., Miao, J., and Song, S. Exploring imitation learning for autonomous driving with feedback synthesizer and differentiable rasterization. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1450–1457. IEEE, 2021.

[44] Ziebart, B. D., Maas, A. L., Bagnell, J. A., Dey, A. K., et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pp. 1433–1438. Chicago, IL, USA, 2008.

Appendices

A Missing Proofs

A.1 Proof of Theorem 4.3

Theorem. *Let $\Pi^* = \text{argmin}_{\Pi}\{J(\boldsymbol{Q}, \Pi, \psi_Q, \psi_V)\}$, then there are a set of optimal local policies $\{\pi_i^*\}$ such that $\pi_i^* = \text{argmin}_{\pi_i} J_i(Q_i, \pi_i),\ \forall i \in \mathcal{N}$, and $\Pi^* = \{\pi_i^*,\ i \in \mathcal{N}\}$.*

Proof. We first consider the local objection function:

$$J_i(Q_i, \pi_i) = \mathbb{E}_{\mathcal{D}^E}\Big[Q_i(o_i, a_i) - \gamma\mathbb{E}_{o_i' \sim P(\cdot|S,A)}\big[V_i^{Q,\pi}(o_i')\big]\Big] - (1-\gamma)\mathbb{E}_{o_i^0}\big[V_i^{Q,\pi}(o_i^0)\big]$$

From this we can see that that $J_i(Q_i, \pi_i)$ is minimized (over π_i) if each value function $V_i^{Q,\pi}(o_i')$ is maximized (over π_i). Moreover, from the definition $V_i^{Q,\pi}(o_i) = \mathbb{E}_{a_i \sim \pi_i(o_i|a_i)}[Q_i(o_i, a_i) - \log \pi_i(a_i \mid o_i)]$, we see that the value of $V_i^{Q,\pi}(o_i)$ is only dependent of $Q_i(o_i, a_i)$ and $\pi_i(a_i|o_i)$ for all $a_i \in \mathcal{A}_i$. It follows that $V_i^{Q,\pi}(o_i)$ achieves its maximization at:

$$\begin{aligned}\pi_i^*(\cdot|o_i) &= \text{argmax}_{\pi_i(\cdot|o_i)}\left\{\mathbb{E}_{a_i \sim \pi_i(o_i|a_i)}[Q_i(o_i, a_i) - \log \pi_i(a_i \mid o_i)]\right\} \\ &= \left\{\pi_i^*(a_i|o_i) = \frac{\exp(Q_i(o_i, a_i))}{\sum_{a_i'} \exp(Q(o_i, a_i'))},\ \forall a_i \in \mathcal{A}_i\right\} \end{aligned} \tag{12}$$

Now let $\Pi^* = \{\pi_i^*,\ i \in \mathcal{N}\}$ be a joint policy (or a set of local policies) defined as in (12). It follows that $V_i^{Q,\pi}(o_i) \le V_i^{Q,\pi^*}(o_i)$ for any $i \in \mathcal{N}$.

We now look at the joint objective function $J(\mathbf{Q}, \Pi, \psi_Q, \psi_V)$, defined as

$$\begin{aligned}J(\mathbf{Q}, \Pi, \psi_Q, \psi_V) = &\sum\nolimits_{(S,A) \in \mathcal{D}^E}\Big[Q^{tot}(S, A) - \gamma\mathbb{E}_{S' \sim P(\cdot|S,A)}\big[V^{tot}_{Q,\Pi}(S')\big]\Big] \\ &- (1-\gamma)\mathbb{E}_{S^0}\big[V^{tot}_{Q,\Pi}(S_0)\big]\end{aligned}$$

Given $V^{tot}_{Q,\Pi}(S) = \mathcal{M}_{\psi_V}(\mathbf{V}^{Q,\Pi}(S))$, our mixing structure ensures that $\mathcal{M}_{\psi_V}(\mathbf{V}^{Q,\Pi}(S))$ is monotonically increasing with respect to each element of $\mathbf{V}^{Q,\Pi}(S)$. Additionally, for any joint policy $\Pi = \{\pi_i,\ i \in \mathcal{N}\}$, we have $V_i^{Q,\pi}(o_i) \le V_i^{Q,\pi^*}(o_i)$ for any $i \in \mathcal{N}$. Combining this with the monotonicity of the mixing network, we obtain:

$$V^{tot}_{Q,\Pi}(S) \le V^{tot}_{Q,\Pi^*}(S),$$

which implies:

$$J(\mathbf{Q}, \Pi, \psi_Q, \psi_V) \ge J(\mathbf{Q}, \Pi^*, \psi_Q, \psi_V)$$

for any joint policy Π. This also implies $\Pi^* = \text{argmin}_{\Pi}\{J(\mathbf{Q}, \Pi, \psi_Q, \psi_V)\}$, as desired. □

A.2 Proof of Proposition 4.4

Proposition. *The max-min problem in (6) is equivalent to the following maximization problem:*

$$\begin{aligned}\max\nolimits_{\boldsymbol{Q}}\Big\{J(\boldsymbol{Q}, \psi_Q, \psi_V) = &\sum\nolimits_{(S,A) \in \mathcal{D}^E}\Big[Q^{tot}(S, A) - \gamma\mathbb{E}_{S' \sim P(\cdot|S,A)}\big[V^{tot}_Q(S')\big]\Big] \\ &- (1-\gamma)\mathbb{E}_{S^0}\big[V^{tot}_Q(S_0)\big]\Big\}\end{aligned}$$

where $V^{tot}_Q(S) = \mathcal{M}_{\psi_V}\big(\boldsymbol{V}^Q(S)\big)$, and $\boldsymbol{V}^Q(S) = \Big(V_i^Q(o_i) \overset{def}{=} \log(\sum_{a_i} \exp(Q_i(o_i, a_i))),\ \ i \in \mathcal{N}\Big)$. Moreover, let $\boldsymbol{Q}^*$ be optimal to (7), then the global optimal policy can be recovered as:

$$\Pi^* = \left\{\pi_i^*(o_i, a_i) = \frac{\exp(Q_i(o_i, a_i)))}{\sum_{a_i'} \exp(Q_i(o_i, a_i'))}\ \middle|\ i \in \mathcal{N}\right\}$$

Proof. We can leverage the result stated in Theorem 4.3 to prove the proposition. That is, Theorem 4.3 tells us that $J(\mathbf{Q}, \Pi, \psi_Q, \psi_V)$ is minimized at $\Pi^* = \{\pi_i^*,\ i \in \mathcal{N}\}$ such that

$$\Pi^* = \left\{ \pi_i^*(o_i, a_i) = \frac{\exp(Q_i(o_i, a_i)))}{\sum_{a_i'} \exp(Q_i(o_i, a_i'))} \middle| \, i \in \mathcal{N} \right\}$$

Therefore, the max-min problem can be converted to a maximization one within the Q-space:

$$\max_{\mathbf{Q}} \min_{\Pi} \{ J(\mathbf{Q}, \Pi, \psi_Q, \psi_V) \} = \max_{\mathbf{Q}} J(\mathbf{Q}, \Pi^*, \psi_Q, \psi_V)$$

Moreover, it is well-known the maximal value $V_i^{Q,\pi^*}(o_i)$ can be computed as a log-sum-exp function of $Q_i(o_i, a_i)$ [14] as:

$$V_i^{Q,\pi^*}(o_i) = \log \left(\sum_{a_i} \exp(Q_i(o_i, a_i)) \right) = V_i^Q(o_i)$$

which confirm our claims. $\square$

A.3 Proof of Theorem 4.5

Theorem *$J(\boldsymbol{Q}, \psi_Q, \psi_V)$ is (strictly) concave in $\boldsymbol{Q}$. As a result,* (7) *always yield a unique solution within the Q-space.*

Proof. To prove the concavity, we will prove that each component of the objective function, i.e. $Q^{tot}(S, A)$ and $V_Q^{tot}(S)$ are concave in $\mathbf{Q}$, keeping in mind that the mixing networks have non-negative weights and convex activations. We will need the following lemmas to validate the main claim.

Lemma A.1. *Any feed-forward mixing networks $\mathcal{M}(\boldsymbol{X})$ constructed with non-negative weights and non-decreasing convex activation functions (such as ReLU or ELU or Maxout) is convex and non-decreasing in $\boldsymbol{X}$.*

Proof. Any N-layer feed-forward network with input $\mathbf{X}$ can be defined recursively as

$$f^0(\mathbf{X}) = \mathbf{X} \tag{13}$$

$$f^n(\mathbf{X}) = \sigma^n\left(f^{n-1}(\mathbf{X})\right) \times W_n + b_n,\ n = 1, \ldots, N \tag{14}$$

where σ^n is a set of activation functions applied to each element of vector $f^{n-1}(\mathbf{X})$, and W_n and b_n are the weights and biases, respectively, at layer n. Therefore, we will prove the result by induction, i.e., $f^n(\mathbf{X})$ is convex and non-decreasing in $\mathbf{X}$ for $n = 0, \ldots$. Here we note that $f^n(\mathbf{X})$ is a vector, so when we say "*$f^n(\boldsymbol{X})$ is convex and non-decreasing in $\boldsymbol{X}$*", it means each element of $f^n(\mathbf{X})$ is convex and non-decreasing in $\mathbf{X}$.

We first see that claim indeed holds for $n = 0$. Now let us assume that, $f^{n-1}(\mathbf{X})$ is convex and non-decreasing in $\mathbf{X}$, we will prove that $f^n(\mathbf{X})$ is also convex and non-decreasing in $\mathbf{X}$. The non-decreasing property can be easily verified as we can see, given two vector $\mathbf{X}$ and $\mathbf{X}'$ such that $\mathbf{X} \geq \mathbf{X}'$ (element-wise comparison), then we have the following chain of inequalities

$$f^{n-1}(\mathbf{X}) \overset{(a)}{\geq} f^{n-1}(\mathbf{X}')$$

$$\sigma^n(f^{n-1}(\mathbf{X})) \overset{(b)}{\geq} \sigma^n(f^{n-1}(\mathbf{X}'))$$

$$\sigma^n(f^{n-1}(\mathbf{X})) \times W_n + b_n \overset{(c)}{\geq} \sigma^n(f^{n-1}(\mathbf{X}')) \times W_n + b_n$$

where (a) is due to the induction assumption that $f^{n-1}(\mathbf{X})$ is non-decreasing in $\mathbf{X}$, (b) is because σ^n are also non-decreasing, and (c) is because the weights W_n is non-negative.

To verify the convexity of $f^n(\mathbf{X})$,we will show that for any $\mathbf{X}, \mathbf{X}'$, and any scalar $\alpha \in (0, 1)$, the following holds

$$\alpha f^n(\mathbf{X}) + (1 - \alpha) f^n(\mathbf{X}) \geq f^n(\alpha \mathbf{X} + (1 - \alpha)\mathbf{X}') \tag{15}$$

To this end, we write

$$\begin{aligned}\alpha f^n(\mathbf{X}) + (1-\alpha) f^n(\mathbf{X}') &= \Big(\alpha\sigma^n(f^{n-1}(\mathbf{X})) + (1-\alpha)\sigma^n(f^{n-1}(\mathbf{X}'))\Big) \times W_n + b_n \\ &\overset{(d)}{\geq} \Big(\sigma^n\Big(\alpha f^{n-1}(\mathbf{X}) + (1-\alpha) f^{n-1}(\mathbf{X}')\Big) \times W_n + b_n \\ &\overset{(e)}{\geq} \Big(\sigma^n\Big(f^{n-1}(\alpha\mathbf{X} + (1-\alpha)\mathbf{X}')\Big) \times W_n + b_n \\ &= f^n(\alpha\mathbf{X} + (1-\alpha)\mathbf{X}')\end{aligned}$$

where (d) is due to the assumption that activation functions σ^n are convex and $W_n \geq 0$, and (e) is because $\alpha f^{n-1}(\mathbf{X}) + (1-\alpha) f^{n-1}(\mathbf{X}') \geq f^{n-1}(\alpha\mathbf{X} + (1-\alpha)\mathbf{X}')$ (because $f^{n-1}(\mathbf{X})$ is convex in $\mathbf{X}$, by the induction assumption) and the activation functions σ^n is non-decreasing and $W_n \geq 0$. So, we have

$$\alpha f^n(\mathbf{X}) + (1-\alpha) f^n(\mathbf{X}') \geq f^n(\alpha\mathbf{X} + (1-\alpha)\mathbf{X}')$$

implying that $f^n(\mathbf{X})$ is convex in $\mathbf{X}$. We then complete the induction proof and conclude that $f^n(\mathbf{X})$ is convex and non-decreasing in $\mathbf{X}$ for any $n = 0, ..., N$. □

Lemma A.2. *For any $i \in \mathcal{N}$,*

$$V_i^Q(o_i) = \log\left(\sum_{a_i} \exp(Q_i(o_i, a_i))\right)$$

are convex in $\boldsymbol{Q}$.

Lemma A.2 can be easily verified, as $V_i^Q(o_i)$ has the log-sum-exp form, so it is convex in $\mathbf{Q}$ [6].

Lemma A.3. *Let $\boldsymbol{X}(\boldsymbol{Q}) = (X_i(\boldsymbol{Q}), \ldots, X_m(\boldsymbol{Q}))$ be a vector of size m where each element is a convex function of $\boldsymbol{Q}$, i.e., $X_i(\boldsymbol{Q})$ is convex in $\boldsymbol{Q}$ for any $i \in \mathcal{N}$, then $\mathcal{M}(\boldsymbol{X}(\boldsymbol{Q}))$ is convex in $\boldsymbol{Q}$, where $\mathcal{M}(\boldsymbol{X}(\boldsymbol{Q}))$ is a mixing network of non-negative weights and convex activations.*

Proof. We will make used general properties of convexity to verify the claim, i.e., we will prove that, for any two vector $\mathbf{Q}$ and $\mathbf{Q}'$, and any scalar $\alpha \in (0, 1)$, the following inequality always holds

$$\alpha\mathcal{M}(\mathbf{X}(\mathbf{Q})) + (1-\alpha)\mathcal{M}(\mathbf{X}(\mathbf{Q}')) \geq \mathcal{M}(\mathbf{X}(\alpha\mathbf{Q} + (1-\alpha)\mathbf{Q}')) \tag{16}$$

According to Lemma A.1, we know that $\mathcal{M}(\mathbf{X})$ is convex in $\mathbf{X}$, thus:

$$\alpha\mathcal{M}(\mathbf{X}(\mathbf{Q})) + (1-\alpha)\mathcal{M}(\mathbf{X}(\mathbf{Q})) \geq \mathcal{M}(\alpha\mathbf{X}(\mathbf{Q}) + (1-\alpha)\mathbf{X}(\mathbf{Q}')) \tag{17}$$

$$= \mathcal{M}\begin{pmatrix}\alpha X_1(\mathbf{Q}) + (1-\alpha)X_1(\mathbf{Q}') \\ \alpha X_2(\mathbf{Q}) + (1-\alpha)X_2(\mathbf{Q}') \\ \ldots \\ \alpha X_m(\mathbf{Q}) + (1-\alpha)X_m(\mathbf{Q}')\end{pmatrix} \tag{18}$$

Moreover, since $X_i(\mathbf{Q})$ is convex in $\mathbf{Q}$ for all $i \in \mathcal{N}$, we have $\alpha X_i(\mathbf{Q}) + (1-\alpha)X_i(\mathbf{Q}') \geq X_i(\alpha\mathbf{Q} + (1-\alpha)\mathbf{Q}')$. In addition, from the monotonicity of $\mathcal{M}(\mathbf{X})$ in each element X_i, we have

$$\mathcal{M}\begin{pmatrix}\alpha X_1(\mathbf{Q}) + (1-\alpha)X_1(\mathbf{Q}') \\ \alpha X_2(\mathbf{Q}) + (1-\alpha)X_2(\mathbf{Q}') \\ \ldots \\ \alpha X_m(\mathbf{Q}) + (1-\alpha)X_m(\mathbf{Q}')\end{pmatrix} \geq \mathcal{M}\begin{pmatrix}X_1(\alpha\mathbf{Q} + (1-\alpha)\mathbf{Q}') \\ X_2(\alpha\mathbf{Q} + (1-\alpha)\mathbf{Q}') \\ \ldots \\ X_m(\alpha\mathbf{Q} + (1-\alpha)\mathbf{Q}')\end{pmatrix} = \mathcal{M}(\mathbf{X}(\alpha\mathbf{Q} + (1-\alpha)\mathbf{Q}')) \tag{19}$$

Combine (18) and (19) we get

$$\alpha\mathcal{M}(\mathbf{X}(\mathbf{Q})) + (1-\alpha)\mathcal{M}(\mathbf{X}(\mathbf{Q})) \geq \mathcal{M}(\mathbf{X}(\alpha\mathbf{Q} + (1-\alpha)\mathbf{Q}'))$$

which validates (16), completing our proof for Lemma A.3. □

We now go back to the main result. First, considering $\mathcal{M}_{\psi_Q}(-\mathbf{Q}(S,A))$, we can see that each element of $-\mathbf{Q}(S,A)$ is linear in $\mathbf{Q}$. Thus, Lemma A.3 tells us that $\mathcal{M}_{\psi_Q}(-\mathbf{Q}(S,A))$ is convex in $\mathbf{Q}$. As a result, $Q^{tot}(S,A)$ is concave in $\mathbf{Q}$. Similarly, $\mathbf{V}^Q(S)$ is also a vector where each element $V_i^Q(o_i)$ is convex in $\mathbf{Q}$ (Lemma A.2), thus $\mathcal{M}_{\psi_V}(\mathbf{V}^Q(S))$ (and $V_Q^{tot}(S)$) is also convex in $\mathbf{Q}$. Therefore, the objective function of our IL, formulated as

$$J(\mathbf{Q},\psi_Q,\psi_V)=\sum\nolimits_{(S,A)\in\mathcal{D}^E}\Big[Q^{tot}(S,A)-\gamma\mathbb{E}_{S'\sim P(\cdot|S,A)}\big[V_Q^{tot}(S')\big]\Big]-(1-\gamma)\mathbb{E}_{S^0}\big[V_Q^{tot}(S_0)\big],$$

should be concave in $\mathbf{Q}$. We further see that $V_i^Q(o_i)$, $\forall o_i$, are strictly convex in $\mathbf{Q}$, implying that $J(\mathbf{Q},\psi_Q,\psi_V)$ is also strictly concave in $\mathbf{Q}$, as desired.

$\square$

It is important to note that the above result only holds if Q_i are independent. It is not the case if Q_i, for some i, share a common network structure. This is also the case of the IQVDN considered in the main paper, i.e., the global reward and value function R^{tot} and V^{tot} are sums of the corresponding local functions, but Q_i share the same neural network structure.

B Additional Details

B.1 MIFQ Algorithm

The detailed steps of our MIFQ algorithm are shown in Algo. 1 below:

Algorithm 1: **M**ulti-agent **I**nverse **F**actorized **Q**-Learning

```
Input: Parameters $\theta=(\theta_1,\ldots,\theta_m)$ and $(\omega_R,\omega_V)$, expert's demonstrations $\mathcal{D}^E$, policy replay
buffer $D^{replay}=\emptyset$, and learning rates $\lambda_\theta$ and $\lambda_\omega$.
repeat
    Compute local policies: $\pi_i(a_i|o_i)=\frac{\exp(Q_i(o_i,a_i;\theta_i))}{\sum_{a'_i\in\mathcal{A}_i}\exp(Q_i(o_i,a'_i;\theta_i))}$,  $\forall i\in\mathcal{N}$;
    # Collect samples for replay buffer
    for a certain number of collecting steps do
        Execute policies $\pi_i$ to collect new transitions $\{(S,A,S')\}$ and add them to $\mathcal{D}^{replay}$;
    # Run gradient descent with mini-batches $\sim\mathcal{D}^{replay}$
    for a certain number of training steps do
        $\theta\leftarrow\theta-\lambda_\theta\nabla_\theta J(\theta,\omega_R,\omega_V)$; $(\omega_R,\omega_V)\leftarrow(\omega_R,\omega_V)-\lambda_\omega\nabla_{\omega_R,\omega_V}J(\theta,\omega_R,\omega_V)$;
until a certain number of loops;
```

B.2 Global versus Local Value Equations

We discuss equations that describe the relationship between the Q and V functions, which we refer to as *Value equations*. Specifically, we consider Eq. 5 for local value functions and Eq. 2 for global ones. So far, we have assumed that the *Value equations* hold for all local agents (i.e., Eq. 5 holds), ensuring that distribution-matching is executed well at the individual level. An alternative approach to the factorized learning is to assume that the *Value equations* hold at the global level, i.e.,

$$V_{Q,\Pi}^{tot}(S)=\mathbb{E}_{A\sim\Pi(A|S)}\left[Q^{tot}(S,A)-\log\Pi(A|S)\right]. \tag{20}$$

Recall that the local value equations can be written as

$$V_i^{Q,\pi}(o_i)=\mathbb{E}_{\pi_i(a_i|o_i)}\left[Q_i(o_i,a_i)-\log\pi_i(a_i|o_i)\right]$$

We then can see that the local and global *Value equations*, generally, cannot hold simultaneously, in particular when V^{tot} and Q^{tot} have different mixing structures. Thus, we can only ensure the validity of either the local *Value equations* in (5) or the global one in (20). It can be seen that, while adhering to (5) is highly tractable due to several advantages shown above (i.e., IGC, non-adversarial training, and convexity), using (20) to compute V^{tot} is highly impractical as due to several reasons. First, it cannot help avoid adversarial training as there is no closed form solution for the minimization over Π, so one needs to learn both $\mathbf{Q}$ and Π adversarially. Second, it requires sampling over joint actions approximate V^{tot}, which shares the same computational issue as the centralized approach discussed in Section 4.1.1, namely, the exponentially large joint action space make the training process highly inefficient.

B.3 Experimental Settings

Each environment has different a observation space, state dimension, and action space. Therefore, we use different hyper-parameters on each task to try to make all algorithms work stably. Moreover, due to limitations in computing resources, especially random access memory, we reduce the buffer size to 5000 on two hardest tasks, Miner and SMACv2, to be more efficient in running time with parallel workers. More details are available in Table 2. We use four High-Performance Computing (HPC) clusters for training and evaluating all tasks. Specifically, each HPC cluster has a workload with an NVIDIA L40 GPU 48 GB GDDR6, 32 Intel-CPU cores, and 100GB RAM. In terms of model architecture, Figure 1 shows our proposed model structure with mixing networks based on QMIX algorithm [29].

Table 2: Hyper-parameters.

Arguments	*MPEs*	*Miner*	*SMACv2*
Max training steps	100000	1000000	
Evaluate times	32		
Buffer size	100000	5000	
Learning rate	2e-5	5e-4	
Batch size	128		
Hidden dim	256		
Gamma	0.99		
Target update frequency	4		
Number of random seeds	4		

B.4 Ablation Study - Non-convex Mixing Networks

Recall that when convex activation functions are used in building mixing networks, our Theorem 4.5 shows that the objective function of the multi-agent inverse soft Q-learning is convex as long as the mixing networks are constructed with non-negative weights and convex activations. This convexity property guarantees the optimization objective is well-behaved and has an unique optimization solution with Q-space, leading to the effectiveness of the IL process. Therefore, in the practical implementation of our algorithm, we chose the ELU activation function. In this section, we empirically demonstrate the impact of convex activation functions by examining the performance of our algorithm when non-convex activation functions such as *sigmoid* or *tanh* are used in mixing networks instead of convex once. Table 3 reports the winrates and rewards, and Figure 4 shows the learning curves of different approaches. These results clearly show that the *ELU* activation significantly outperform those using non-convex activations (i.e., *sigmoid, tanh*), for both versions *MIFQ (Soft)* and *MIFQ (Det)*.

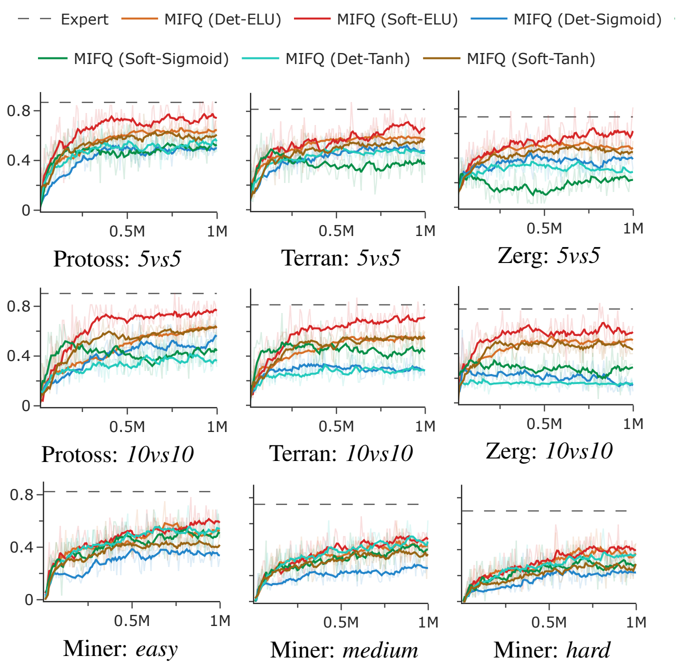
B.5 Ablation Study - Comparison with SAC-based IQ-learn

In this section, we explore the performance of SAC-IQ, an adaptation of the Soft Actor-Critic (SAC) algorithm combined with IQ-Learn, tailored for multi-agent settings. SAC-IQ employs a centralized Q function alongside decentralized critics, maintaining adherence to the original objective of IQ-Learn. This approach is evaluated across 9 distinct tasks within two environments: SMACv2 and Gold Miner. For SMACv2, we consider scenarios with varying team sizes, specifically 5vs5 and 10vs10. In the Gold Miner environment, we assess performance across three difficulty levels: easy, medium, and hard.

Table 3: Winrate and reward comparisons between different activation functions (the values are percentages for SMACv2 and Miner, and rewards for MPE.)

Scenarios		**Expert**	**MIFQ-ELU**		**MIFQ-Sigmoid**		**MIFQ-Tanh**	
			Det	**Soft**	**Det**	**Soft**	**Det**	**Soft**
Protoss	*5vs5*	86.7	64.8±4.1	72.7±10.0	49.2±4.6	53.1±6.6	60.9±10.0	62.5±3.1
	10vs10	90.3	61.7±5.8	77.3±5.6	59.2±8.2	45.3±9.2	36.9±7.2	64.4±1.1
Terran	*5vs5*	81.7	55.9±0.4	71.9±3.8	48.4±2.9	35.2±10.5	45.1±2.7	60.6±6.7
	10vs10	81.7	53.8±6.9	72.7±3.4	26.0±2.2	46.1±9.2	28.6±2.6	53.8±6.7
Zerg	*5vs5*	73.5	46.7±5.2	60.9±14.7	39.8±5.1	21.1±5.1	27.9±2.2	43.8±5.7
	10vs10	76.3	51.0±2.2	59.4±3.1	12.5±4.9	29.7±2.7	15.2±3.9	41.3±2.2
Miner	*easy*	82.4	52.7±0.7	60.2±5.6	32.0±4.6	53.1±7.7	56.4±7.2	43.1±4.5
	medium	74.9	43.8±2.2	49.2±5.1	24.8±2.2	40.6±5.8	45.7±10.1	38.7±12.6
	hard	69.8	39.1±5.2	39.8±4.6	21.7±2.0	25.8±3.4	34.8±3.9	32.5±10.8
MPE	*reference*	-17.2	-23.4±0.1	-20.5±0.0	-40.4±1.2	-23.0±0.1	-22.1±0.1	-21.7±0.1
	spread	-10.7	-23.2±0.1	-24.2±0.1	-46.4±0.1	-24.7±0.7	-27.8±1.1	-26.6±1.5
	speaker	-19.7	-30.8±0.1	-26.3±0.0	-133.4±0.1	-28.9±0.0	-34.7±1.0	-28.2±0.0

Figure 4: Convergence curves of MIFQ with different activation functions



The results, detailed in Table 4, illustrate that SAC-IQ demonstrates commendable performance when compared to other baselines such as Behavior Cloning (BC), Independent IQ-Learn (IIQ), and Multi-Agent Generative Adversarial Imitation Learning (MAGAIL). However, it does not surpass the effectiveness of our proposed methods. Figure 5 provides a visual representation of the learning curves, highlighting the training dynamics of SAC-IQ in comparison to other approaches. This visualization underscores the competitive nature of SAC-IQ while also affirming the superior performance of our methods in these challenging multi-agent environments.

B.6 Experimental Details

In this section, we present the experimental details for the SMACv2, Miner, and MPE tasks, with varying numbers of expert demonstrations. These details complement the box plots shown in Figure 3 in the main paper. Tables 4-9 report win rates (as percentage values) for SMACv2 and Miner tasks with the number of trajectories ranging from 128 to 4096. Table 10 reports final rewards for MPE tasks with the number of trajectories ranging from 1 to 128. We also plot the learning curves of these experiments in Figure 7.

Figure 5: Learning curves of SAC-IQ compared with other baselines and our methods.

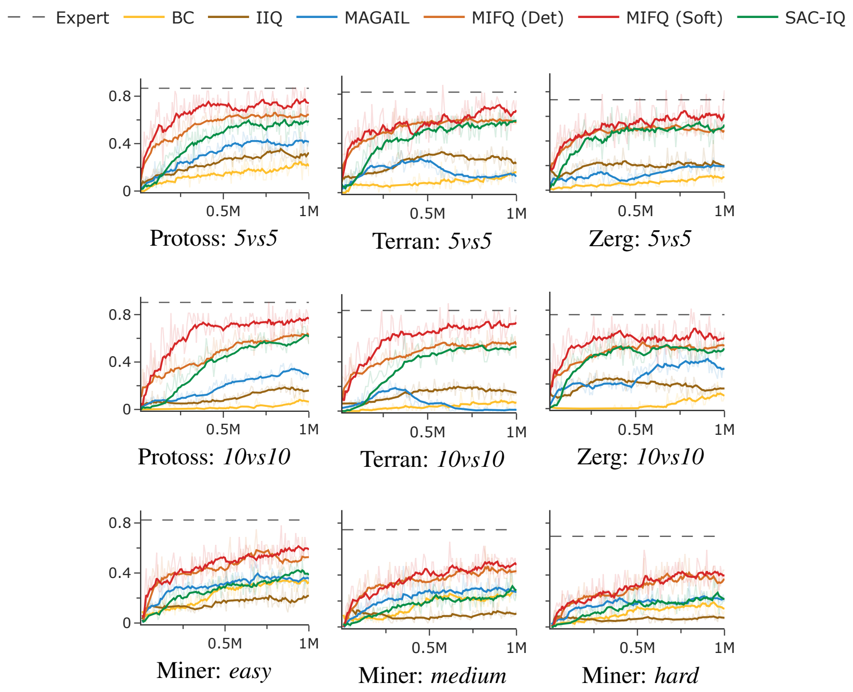


Table 4: Winning rate (percentage) of SAC-IQ compared with other baselines and our methods.

Task		Expert	MAGAIL	MIFQ Det	MIFQ Soft	SAC-IQ
Protoss	5vs5	86.7	42.6±4.6	64.8±4.1	72.7±10.0	60.6±3.2
	10vs10	90.3	28.3±2.3	61.7±5.8	77.3±5.6	64.4±6.2
Terran	5vs5	81.7	10.9±4.5	55.9±0.4	71.9±3.8	59.4±2.1
	10vs10	81.7	1.0±0.7	53.8±6.9	72.7±3.4	53.7±6.3
Zerg	5vs5	73.5	18.8±0.6	46.7±5.2	60.9±14.7	55.6±6.2
	10vs10	76.3	31.2±3.6	51.0±2.2	59.4±3.1	51.3±6.3
Miner	easy	82.4	34.6±3.3	52.7±0.7	60.2±5.6	35.0±5.3
	medium	74.9	26.0±1.5	43.8±2.2	49.2±5.1	26.9±6.7
	hard	69.8	20.9±2.4	39.1±5.2	39.8±4.6	18.8±1.3

Table 5: Results in percentage on SMACv2 & Miner, the number of expert trajectories is: 128

Scenarios		Expert	BC	IIQ	IQ-VDN	MA-SQIL	MA-AIRL	MA-GAIL	MIFQ Det	MIFQ Soft
Protoss	*5vs5*	86.7	17.2±4.7	13.3±2.6	8.6±2.6	0.8±1.4	0.8±0.6	0.0±0.0	18.0±4.6	41.4±4.6
	10vs10	90.3	6.2±4.4	2.3±1.4	3.1±3.8	0.0±0.0	0.0±0.0	0.0±0.0	17.2±5.2	32.8±8.4
Terran	*5vs5*	81.7	9.4±4.9	12.5±3.1	9.4±9.1	0.0±0.0	0.0±0.0	0.0±0.0	25.0±4.4	32.0±3.4
	10vs10	81.7	3.9±3.4	1.6±2.7	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	3.1±2.2	35.2±7.1
Zerg	*5vs5*	73.5	7.0±4.6	10.2±5.6	4.2±4.2	3.9±2.6	2.0±1.2	3.1±3.8	5.5±2.6	24.2±4.1
	10vs10	76.3	10.2±3.4	0.8±1.4	0.0±0.0	0.0±0.0	2.1±2.4	0.0±0.0	10.9±1.6	28.9±9.2
Miner	*easy*	82.4	28.9±8.1	10.2±3.4	13.3±1.4	12.1±2.0	14.1±4.7	10.5±3.7	14.1±8.4	29.7±6.4
	medium	74.9	23.4±3.5	7.0±3.4	9.4±3.8	11.5±3.5	13.3±1.4	10.0±2.4	11.3±1.8	21.1±8.1
	hard	69.8	15.6±5.8	5.5±1.4	7.0±1.4	7.4±1.6	4.7±2.7	7.0±1.5	8.0±3.2	14.8±4.1

Figure 6: Comparison with different numbers of demonstrations. X-axis: winning rate/final score. Y-axis: number of demonstrations.

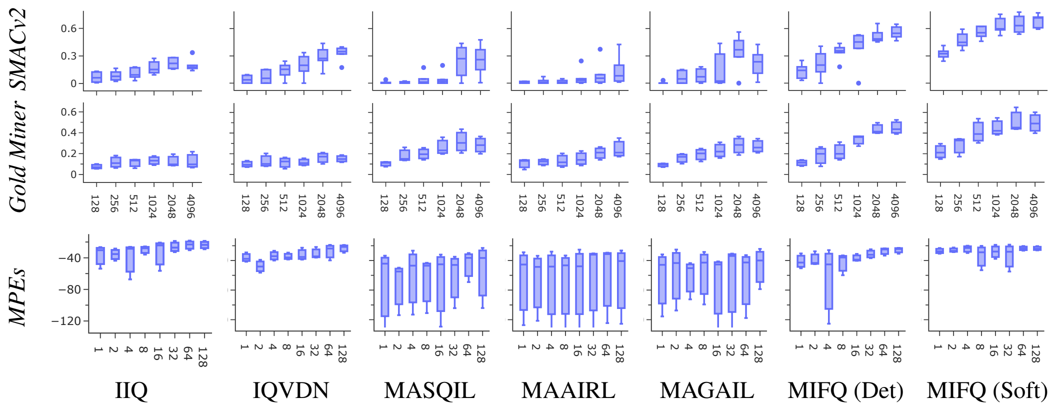


Table 6: Results in percentage on SMACv2 & Miner, the number of expert trajectories is: 256

Scenarios		**Expert**	**BC**	**IIQ**	**IQ-VDN**	**MA-SQIL**	**MA-AIRL**	**MA-GAIL**	**MIFQ Det**	**MIFQ Soft**
Protoss	*5vs5*	86.7	12.5±2.2	16.5±2.4	15.1±1.6	1.6±1.6	7.2±3.2	14.1±4.7	40.6±10.4	53.9±6.8
	10vs10	90.3	6.2±4.9	3.9±2.6	2.9±3.4	0.0±0.0	0.2±0.3	15.0±4.2	32.3±2.9	59.4±8.6
Terran	*5vs5*	81.7	10.9±4.7	12.5±2.2	14.8±2.6	0.8±1.4	0.0±0.0	0.0±0.0	27.3±5.6	46.1±7.5
	10vs10	81.7	4.7±1.6	3.9±4.1	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	43.8±16.7
Zerg	*5vs5*	73.5	8.6±2.6	11.7±2.6	7.8±6.0	1.6±1.6	2.9±2.2	6.2±2.2	12.5±3.8	35.9±5.2
	10vs10	76.3	5.5±2.6	2.3±1.4	0.0±0.0	2.3±2.6	0.4±0.4	3.1±2.2	12.5±3.8	41.4±3.4
Miner	*easy*	82.4	38.3±6.0	18.2±6.2	20.3±5.6	26.2±2.8	14.8±4.6	19.9±2.5	26.6±6.4	34.4±4.4
	medium	74.9	21.9±10.8	10.9±6.4	10.2±2.6	15.0±4.0	12.5±3.8	17.0±3.2	19.9±3.4	32.8±8.4
	hard	69.8	20.3±1.6	6.2±1.9	7.8±3.5	13.5±3.4	8.6±4.1	10.4±2.2	7.8±1.7	17.2±8.1

Table 7: Results in percentage on SMACv2 & Miner, the number of expert trajectories is: 512

Scenarios		**Expert**	**BC**	**IIQ**	**IQ-VDN**	**MA-SQIL**	**MA-AIRL**	**MA-GAIL**	**MIFQ Det**	**MIFQ Soft**
Protoss	*5vs5*	86.7	21.9±6.6	17.2±1.6	19.5±8.1	0.0±0.0	4.9±1.4	10.2±5.1	43.8±3.1	62.5±7.7
	10vs10	90.3	5.5±2.6	7.0±2.6	9.1±2.3	0.0±0.0	0.0±0.0	18.0±7.1	35.9±11.4	62.5±9.4
Terran	*5vs5*	81.7	9.4±4.9	18.0±3.4	24.2±6.8	4.7±3.5	0.8±1.0	3.9±4.1	39.1±10.0	59.4±8.6
	10vs10	81.7	7.0±4.1	7.8±6.4	18.8±18.8	1.6±1.6	0.0±0.0	0.0±0.0	18.0±7.8	51.6±7.8
Zerg	*5vs5*	73.5	6.2±5.8	10.9±3.5	11.7±8.4	17.2±3.5	2.9±0.3	15.6±3.1	33.6±7.8	51.6±9.2
	10vs10	76.3	5.5±4.1	3.1±3.8	0.0±0.0	0.8±1.4	0.2±0.3	1.6±1.6	35.2±4.1	46.1±4.1
Miner	*easy*	82.4	30.5±5.1	13.9±2.9	16.4±6.0	25.8±4.7	20.3±3.5	25.0±4.3	31.2±3.1	53.9±4.1
	medium	74.9	18.8±3.1	14.3±3.4	12.5±2.2	19.5±5.6	11.7±3.4	20.3±4.7	19.9±3.7	39.1±3.5
	hard	69.8	15.6±5.8	6.1±0.9	5.5±2.6	14.1±2.4	7.0±2.6	12.5±1.8	14.1±4.6	30.5±4.1

Table 8: Results in percentage on SMACv2 & Miner, the number of expert trajectories is: 1024

Scenarios		**Expert**	**BC**	**IIQ**	**IQ-VDN**	**MA-SQIL**	**MA-AIRL**	**MA-GAIL**	**MIFQ Det**	**MIFQ Soft**
Protoss	*5vs5*	86.7	16.4±8.9	23.4±3.5	25.8±3.4	3.9±3.4	24.4±2.5	43.8±6.6	49.2±5.6	75.8±2.6
	10vs10	90.3	4.7±4.7	16.4±4.1	29.2±9.3	0.0±0.0	3.9±3.4	3.9±6.8	53.1±5.8	71.1±2.6
Terran	*5vs5*	81.7	12.5±5.4	27.3±3.4	33.6±10.2	0.0±0.0	1.0±0.6	0.8±1.4	52.3±2.6	60.9±6.8
	10vs10	81.7	7.8±8.1	14.1±1.6	13.3±1.4	2.3±1.4	0.0±0.0	0.0±0.0	0.0±0.0	58.6±2.6
Zerg	*5vs5*	73.5	5.5±4.6	9.4±2.2	13.5±1.8	19.5±3.4	4.9±4.4	32.0±10.5	38.3±5.1	58.6±3.4
	10vs10	76.3	9.4±5.4	11.7±3.4	0.0±0.0	1.6±1.6	2.3±2.6	0.8±1.4	41.4±5.6	52.3±7.5
Miner	*easy*	82.4	34.4±8.6	17.8±2.3	15.6±5.4	35.9±4.2	22.7±3.4	31.1±3.0	36.9±3.1	54.7±8.1
	medium	74.9	20.3±4.7	13.3±4.1	10.9±4.7	23.2±1.9	14.1±1.6	22.1±2.4	36.7±8.7	42.2±1.6
	hard	69.8	18.0±2.6	9.0±2.3	8.6±4.1	19.7±4.0	8.6±2.6	15.8±5.6	27.3±6.8	38.3±7.1

Table 9: Results in percentage on SMACv2 & Miner, the number of expert trajectories is: 2048

Scenarios		Expert	BC	IIQ	IQ-VDN	MA-SQIL	MA-AIRL	MA-GAIL	MIFQ	
									Det	Soft
Protoss	*5vs5*	86.7	10.2±5.1	28.1±4.9	28.1±10.4	39.1±10.0	37.3±4.4	56.2±10.6	65.6±11.0	73.4±5.2
	10vs10	90.3	2.3±2.6	16.4±3.4	25.0±5.9	43.8±10.6	9.4±1.5	39.8±6.4	45.3±1.6	78.1±5.8
Terran	*5vs5*	81.7	10.2±6.0	28.1±4.4	36.7±7.1	8.6±3.4	7.0±1.4	28.9±5.1	55.5±5.1	56.2±7.3
	10vs10	81.7	5.5±2.6	24.2±2.6	43.8±11.0	0.0±0.0	0.0±0.0	0.0±0.0	46.9±8.8	69.5±7.1
Zerg	*5vs5*	73.5	7.0±4.6	15.6±3.8	26.3±5.3	28.1±3.8	3.9±1.5	33.6±7.5	48.1±8.0	57.0±10.2
	10vs10	76.3	5.5±2.6	19.5±3.4	10.4±3.6	25.8±8.1	1.2±1.3	46.9±7.3	49.2±2.6	53.9±11.8
Miner	*easy*	82.4	35.9±3.5	19.5±1.2	21.1±1.4	43.8±2.2	26.6±1.6	36.7±7.1	50.0±4.9	64.8±6.0
	medium	74.9	19.5±7.1	9.8±4.4	17.2±3.5	30.5±4.6	21.1±3.4	28.5±2.2	41.4±5.1	43.8±11.5
	hard	69.8	10.2±3.4	8.6±2.0	10.2±2.6	20.7±2.6	14.1±6.8	18.8±3.1	39.8±8.1	45.3±4.7

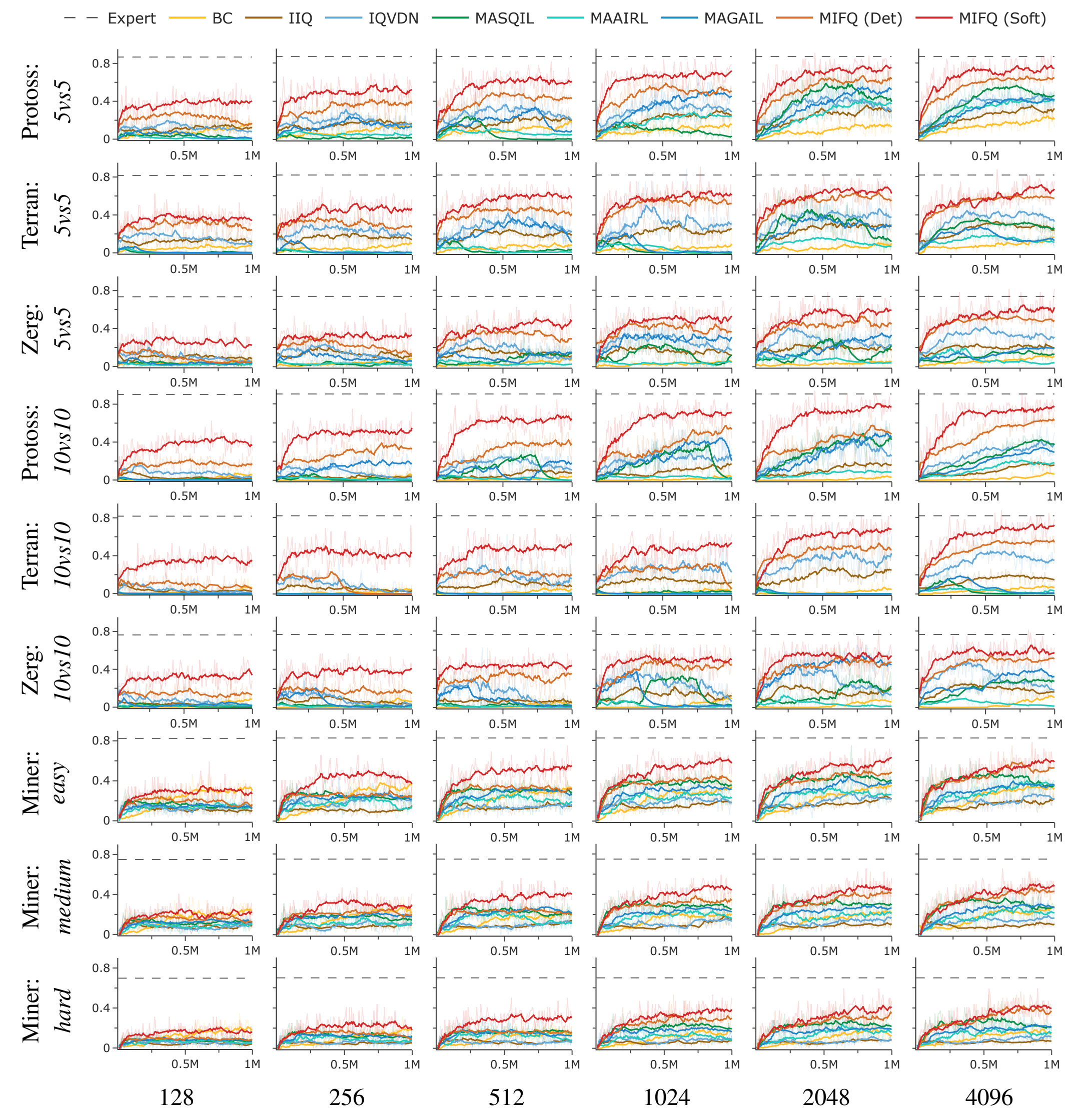
Table 10: Results in percentage on SMACv2 & Miner, the number of expert trajectories is: 4096

Scenarios		Expert	BC	IIQ	IQ-VDN	MA-SQIL	MA-AIRL	MA-GAIL	MIFQ	
									Det	Soft
Protoss	*5vs5*	86.7	19.5±5.6	33.6±8.9	39.8±4.1	47.7±2.5	42.6±3.9	42.6±4.6	64.8±4.1	72.7±10.0
	10vs10	90.3	5.5±1.4	16.6±0.3	38.3±8.9	36.8±2.8	19.8±2.9	28.3±2.3	61.7±5.8	77.3±5.6
Terran	*5vs5*	81.7	18.0±2.6	19.7±5.3	32.0±6.0	24.6±4.2	10.9±4.5	10.9±4.5	55.9±0.4	71.9±3.8
	10vs10	81.7	6.2±2.2	14.1±1.7	35.9±4.7	0.5±0.5	2.3±1.4	1.0±0.7	53.8±6.9	72.7±3.4
Zerg	*5vs5*	73.5	10.9±3.5	18.6±1.4	33.6±7.5	14.8±3.1	5.3±1.2	18.8±0.6	46.7±5.2	60.9±14.7
	10vs10	76.3	9.4±3.8	16.4±0.6	17.2±3.5	27.1±4.8	1.2±1.2	31.2±3.6	51.0±2.2	59.4±3.1
Miner	*easy*	82.4	31.2±5.8	21.9±4.4	18.8±3.8	36.7±2.9	35.2±2.6	34.6±3.3	52.7±0.7	60.2±5.6
	medium	74.9	28.1±3.8	9.8±1.6	14.8±3.4	28.3±1.9	21.1±2.6	26.0±1.5	43.8±2.2	49.2±5.1
	hard	69.8	12.5±2.2	6.6±0.7	11.7±2.6	19.9±2.0	17.2±6.4	20.9±2.4	39.1±5.2	39.8±4.6

Table 11: Results on MPEs, the number of expert trajectories is: [1, 2, 4, 8, 16, 32, 64, 128]

Expert Traj.	Scenarios	Expert	BC	IIQ	IQ-VDN	MA-SQIL	MA-AIRL	MA-GAIL	MIFQ	
									Det	Soft
1	*reference*	-17.2	-20.5±0.0	-26.5±0.1	-29.1±0.1	-33.5±0.9	-45.4±2.4	-45.8±0.3	-42.8±0.3	-23.1±0.0
	spread	-10.7	-21.7±0.0	-30.3±0.2	-41.6±0.4	-44.2±0.1	-28.9±0.6	-31.8±1.5	-30.4±0.1	-30.8±0.4
	speaker	-19.7	-28.5±0.0	-53.8±0.4	-37.7±0.2	-138.9±0.4	-127.4±0.1	-115.6±1.6	-50.5±0.3	-27.6±0.0
2	*reference*	-17.5	-24.1±0.2	-28.4±0.1	-39.0±1.2	-49.7±0.1	-48.6±0.1	-43.4±0.1	-44.6±0.6	-22.4±0.1
	spread	-11.4	-24.3±0.0	-35.3±0.2	-48.1±1.9	-54.8±0.2	-33.6±2.2	-25.2±0.1	-27.5±0.6	-28.4±0.4
	speaker	-19.2	-27.6±0.0	-43.7±0.4	-56.4±4.6	-113.9±3.7	-121.5±6.9	-107.6±0.2	-44.2±0.0	-27.3±0.0
4	*reference*	-17.5	-22.5±0.1	-26.2±0.4	-26.8±0.7	-26.7±0.0	-48.2±0.3	-50.3±0.2	-45.2±0.2	-20.0±0.0
	spread	-11.4	-23.8±0.0	-28.5±4.8	-34.2±1.1	-46.9±0.2	-28.0±0.2	-43.5±0.2	-25.9±0.2	-23.1±0.4
	speaker	-19.2	-29.6±0.0	-67.3±1.3	-40.0±1.9	-113.1±0.6	-134.0±0.0	-92.5±1.2	-125.4±0.2	-29.9±0.0
8	*reference*	-17.5	-19.0±0.1	-25.6±0.1	-35.9±1.6	-43.1±0.6	-46.5±1.0	-42.8±1.1	-33.1±0.3	-20.2±0.1
	spread	-11.4	-21.7±0.0	-27.9±0.1	-29.3±0.2	-47.0±1.2	-33.2±1.9	-28.4±0.7	-35.8±0.1	-53.1±0.9
	speaker	-19.2	-32.9±0.0	-35.9±0.1	-38.6±0.3	-111.1±0.2	-134.0±0.0	-99.4±0.5	-59.9±0.2	-28.9±0.1
16	*reference*	-17.5	-18.2±0.1	-20.8±0.2	-22.7±0.2	-32.0±0.5	-47.8±2.1	-45.4±0.6	-38.9±0.2	-18.8±0.1
	spread	-11.4	-21.3±0.0	-24.2±0.2	-38.7±1.1	-45.1±0.2	-25.3±0.5	-41.2±1.0	-31.0±0.3	-37.7±0.6
	speaker	-19.2	-31.9±0.0	-56.5±0.1	-35.2±0.5	-129.4±0.4	-134.0±0.0	-134.0±0.0	-40.4±0.5	-27.9±0.0
32	*reference*	-17.5	-18.3±0.0	-18.7±0.2	-21.7±0.2	-46.0±0.2	-30.4±1.5	-30.7±1.3	-32.2±0.1	-18.6±0.1
	spread	-11.4	-20.6±0.0	-27.1±0.1	-34.7±0.5	-34.9±0.3	-32.1±0.7	-33.8±0.4	-24.8±0.2	-54.9±0.4
	speaker	-19.2	-29.0±0.0	-32.8±0.2	-35.9±0.1	-104.4±0.2	-132.5±2.1	-134.8±0.4	-36.8±1.0	-28.1±0.0
64	*reference*	-17.5	-18.1±0.0	-18.3±0.2	-18.9±0.2	-36.4±0.7	-31.3±1.0	-42.6±0.2	-24.6±0.1	-20.4±0.0
	spread	-11.4	-20.3±0.0	-23.0±0.1	-23.8±0.2	-29.9±0.2	-29.4±0.0	-32.7±0.1	-23.5±0.2	-23.7±0.2
	speaker	-19.2	-28.4±0.0	-30.3±0.3	-39.8±1.1	-69.2±1.7	-124.6±0.3	-117.7±0.1	-33.4±0.0	-26.7±0.0
128	*reference*	-17.2	-18.3±0.0	-18.6±0.2	-19.0±0.1	-36.6±1.5	-40.7±3.2	-40.0±0.2	-23.4±0.1	-20.5±0.0
	spread	-10.7	-20.5±0.0	-23.6±0.2	-21.6±0.2	-23.0±0.4	-26.5±0.3	-24.3±0.1	-23.2±0.1	-24.2±0.1
	speaker	-19.7	-27.5±0.0	-29.1±0.1	-29.5±0.5	-104.3±5.0	-125.5±0.7	-78.7±4.9	-30.8±0.1	-26.3±0.0

Figure 7: Learning curves with different numbers of demonstrations.



NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: *Our abstract includes our main claims reflecting our main contributions and finding.*

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: *We created a subsection in the Conclusion to discuss limitations.*

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: *All the proofs of the theorems and propositions stated in the main paper are provided in the appendix with clear references.*

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: *We provide details on the environments and hyper-parameter settings in the appendix. We also uploaded our source code for re-productivity purposes.*

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: *The data we used, along with our source code, has been uploaded with the main paper. We have also provided sufficient instructions for their use.*

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: *All the details are provided in the main paper and appendix.*

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: *Following standard practices, we reported mean scores and standard deviations, computed by running several random seeds, for comparison.*

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: *We provides all the details in the appendix.*

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: *The paper develops a general imitation learning algorithm for multi-agent games, which we have tested only in simulated environments. As such, we do not foresee any direct societal impact.*

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: **[TODO]**

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: *We have provided clear citations to the source code and data we used in the paper.*

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: *Our source code is submitted alongside the paper, accompanied by sufficient instructions. We will share the code publicly for re-producibility or benchmarking purposes.*

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: **[TODO]**

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: **[TODO]**

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

https://doi.org/10.52202/079017-0854