Modeling the Interaction between Agents in Cooperative Multi-Agent Reinforcement Learning

Xiaoteng Ma[*]
Tsinghua University
Beijing, China
ma-xt17@mails.tsinghua.edu.cn

Yiqin Yang[*]
Tsinghua University
Beijing, China
yangyiqi19@mails.tsinghua.edu.cn

Chenghao Li[*]
Tsinghua University
Beijing, China
lich18@mails.tsinghua.edu.cn

Yiwen Lu
Tsinghua University
Beijing, China
luyw20@mails.tsinghua.edu.cn

Qianchuan Zhao
Tsinghua University
Beijing, China
zhaoqc@tsinghua.edu.cn

Jun Yang
Tsinghua University
Beijing, China
yangjun603@tsinghua.edu.cn

ABSTRACT

Value-based methods of multi-agent reinforcement learning (MARL), especially the value decomposition methods, have been demonstrated on a range of challenging cooperative tasks. However, current methods pay little attention to the interaction between agents, which is essential to teamwork in games or real life. This limits the efficiency of value-based MARL algorithms in the two aspects: collaborative exploration and value function estimation. In this paper, we propose a novel cooperative MARL algorithm named as interactive actor-critic (IAC), which models the interaction of agents from the perspectives of policy and value function. On the policy side, a multi-agent joint stochastic policy is introduced by adopting a collaborative exploration module, which is trained by maximizing the entropy-regularized expected return. On the value side, we use the shared attention mechanism to estimate the value function of each agent, which takes the impact of the teammates into consideration. At the implementation level, we extend the value decomposition methods to continuous control tasks and evaluate IAC on benchmark tasks including classic control and multi-agent particle environments. Experimental results indicate that our method outperforms the state-of-the-art approaches and achieves better performance in terms of cooperation.

KEYWORDS

Collaborative Exploration; Multi-Agent Reinforcement Learning; Maximum Entropy Learning

1 INTRODUCTION

In recent years, cooperative multi-agent reinforcement learning is an active research area [4, 5, 14, 40] and has made exciting

---

[*]Equal contribution.

progress in many domains, including traffic signal network optimization [7, 31, 42] and network packet routing [22]. The full spectrum of cooperative MARL algorithms is a hybrid of *centralized training and decentralized execution* (CTDE) [25]. In the CTDE framework, value-based methods receive the most attention and they can be divided into two categories: centralized value function methods and value decomposition methods [38].

The centralized value function methods allow agents to learn approximate models of other agents online and further effectively use estimated models in their policy learning procedure. For example, MADDPG employs the deterministic policies of other agents to learn the centralized value function [20]. Counterfactual multi-agent policy gradient (COMA) adopts a counterfactual baseline to marginalize out a single agent's action while keeping the other agents' actions fixed [9]. MADDPG and COMA introduce respectively the deterministic policy gradient and the policy gradient methods into MARL. Unfortunately, the suboptimality of one agent's policy can propagate through the centralized value function and negatively affect policy learning of other agents, which is named *centralized-decentralized mismatch* (CDM) [36]. To alleviate the problem, Multi Actor Attention Critic (MAAC) adopts a shared attention mechanism, a successful approach applied in various domains [6, 23, 34, 35], to select relevant value function information of agents to calculate the centralized value function [17]. However, MAAC has a limited performance improvement compared with MADDPG and cannot get rid of the CDM problem.

Different from centralized value function methods, current value decomposition methods adopt structural constraints to meet that local maxima on the per-agent action-values amount to the global maximum on the joint action-value, which is named Individual-Global-Max (IGM) assumption. Value-Decomposition Networks (VDN) proposes a learned additive value-decomposition approach over individual agents, which aims to learn an optimal linear value decomposition from the team reward signal [30]. Another example is QMIX, which enforces a monotonic structure between the joint action-value function and the per-agent action-value functions. This structure allows tractable maximization of the joint action-value function in off-policy learning and guarantees consistency between the centralized and decentralized policies [28]. Moreover, QTRAN proposes a new factorization method, which can transform the original joint action-value function into an easily factorizable one, with the same optimal actions [29]. These value decomposition

methods can effectively estimate the joint action-value function by calculating individual action-value functions, and achieve better performance than centralized value function methods.

Compared with single-agent RL, the growth of the agents' number in MARL causes the exponential growth of the state-action space. This phenomenon seriously affects the estimation accuracy of the value function, which is conspicuous for centralized value function methods. The variance of multi-agent policy gradient would grow exponentially as the number of agents increases [36]. Value decomposition methods avoid the curse of dimensionality in estimating joint action-value function by assuming that the value function of each agent is independent, while the IGM assumption limits the representation power of joint action-value function [16]. For example, in the traffic system at the crossroads, if the running car only maximizes self expected reward, the transportation system may fall into chaos [2]. Therefore, the optimization objective of agents in cooperative tasks should contain cumulative self expected reward and cooperative expected reward. We name the newly defined expected objective as *mutual value function information* (MVFI).

From the policy view, another problem is the collaborative exploration, which is neglected in current MARL algorithms. When directly applying the existing MARL algorithms to complex games with a large number of agents, agents may fail to learn good strategies and end up with little interaction with other agents even when collaboration is significantly beneficial [19]. Back to single-agent RL, maximum entropy policy achieves conspicuous success in exploration by acquiring diverse behaviors. Some works have revealed that the objective with entropy regularization enjoys the smoother optimization landscape and the faster convergence rate, which builds connections between RL with probabilistic graphical models and convex optimization [1]. Therefore, directly introducing the maximum entropy policy into MARL is a natural idea, which is already explored [12]. However, this method just aims to impose the maximum entropy policy on the individual policies rather than the joint stochastic policy. Similar work has done to generalize maximum entropy policy into the stochastic games [11], such as the two-player soft q-learning (TPSQL). Although this work shows that the stochastic game with soft Q-learning would exhibit an optimal state value, TPSQL is restricted to the two-player zero-sum stochastic games. For this reason, designing an effective multi-agent maximum entropy algorithm remains an open research problem.

Contributions. In this paper, we propose an novel off-policy cooperative MARL algorithm, named IAC. The full architecture is discussed in detail in Section 4. Our method has three following key technical and experimental contributions. We analyze the necessity and challenge of modeling the interaction between agents. To deal with the problems, we propose a multi-agent joint stochastic policy by adopting a collaborative exploration module on the policy side, which is trained by maximizing the entropy-regularized expected return. On the value side, we propose a new optimization objective named mutual value function information, which is estimated by the shared attention mechanism. As for the implementation issue, we extend the value decomposition methods to continuous control tasks and evaluate our algorithm on benchmark tasks including classic control and multi-agent cooperative tasks. Experimental results are shown in Section 5, which demonstrates that IAC not

only outperforms the state-of-the-art approaches but also achieves better performance in terms of cooperation, robustness, and scalability. Moreover, the positive performances of multi-agent joint stochastic policy and MVFI are confirmed respectively in experiments. Our work appears to be the first study of modeling the interaction between agents under CTDE framework in cooperative MARL.

2 BACKGROUND

In this section, we give the introduction to the problem formulation of multi-agent cooperation and the basic building blocks for our approach.

2.1 Notation

We consider Dec-POMDP [24] as a standard model consisting of a tuple $G = \langle S, U, P, R, Z, O, \gamma \rangle$ for cooperative multi-agent tasks. Within G, $s \in S$ denotes the global state of the environment. Each agent $i \in N := 1, ..., n$ chooses an action $u_i \in U$ at each time, forming a joint action $\mathbf{u} \in U^n$. Let $P(s' \mid s, \mathbf{u}) : S \times U^n \times S \to [0, 1]$ denotes the state transition function. The reward function $R(s, \mathbf{u}) : S \times U^n \to \mathbb{R}$ is shared among all agents and $\gamma \in [0, 1)$ is the discount factor.

In a partially observable scenario, each agent has individual observations $z \in Z$ according to the observation function $Z(s, u) : S \times U \to O$, which conditions on a stochastic policy $\pi_i(u_i \mid o_i)$ parameterized by $\theta_i : O \times U \to [0, 1]$. Each agent has its observation history $\tau_i \in T := (O \times U)^t$. The joint action-value function is defined as

$$Q_{\text{tot}}^{\boldsymbol{\pi}}(s, \boldsymbol{u}) = \mathbb{E}_{s_t, \boldsymbol{u}_t \sim \tau_{\boldsymbol{\pi}}} \left[\sum_{t=0}^{T} \gamma^t r_t \mid s_t = s, \boldsymbol{u}_t = \boldsymbol{u} \right], \quad (1)$$

where $\boldsymbol{\pi}$ is the joint policy with parameters $\theta = \langle \theta_1, \ldots, \theta_n \rangle$.

2.2 Value Decomposition Methods

Value decomposition methods are widely used in value-based MARL algorithms [16, 27]. These methods estimate the joint action-value function by learning individual action-value functions. For a joint action-value function $Q_{\text{tot}}^{\boldsymbol{\pi}}(\boldsymbol{o}, \mathbf{u})$, if the following holds

$$\arg\max_{\mathbf{u}} Q_{\text{tot}}^{\boldsymbol{\pi}}(\boldsymbol{o}, \mathbf{u}) = \left(\begin{array}{c} \arg\max_{u_1} Q_1(o_1, u_1) \\ \vdots \\ \arg\max_{u_n} Q_n(o_n, u_n) \end{array} \right), \quad (2)$$

Q_i satisfies **IGM Condition** and the $Q_{\text{tot}}^{\boldsymbol{\pi}}(\boldsymbol{o}, \mathbf{u})$ can be factorized by $Q_i(o_i, u_i)$. Equation 2 demonstrates that local maxima on the individual action-values amount to the global maximum on the joint action-value.

Building on purely independent DQN-style agents, VDN [30] assumes that the joint action-value function can be additively decomposed into action-value functions across agents. Differently, QMIX [28] applies a structural constraint between $Q_{\text{tot}}^{\boldsymbol{\pi}}(\boldsymbol{o}, \mathbf{u})$ and $Q_i(o_i, u_i)$ to meet the monotonic assumption

$$\frac{\partial Q_{\text{tot}}^{\boldsymbol{\pi}}(\boldsymbol{o}, \mathbf{u})}{\partial Q_i(o_i, u_i)} \geq 0, \forall i \in N. \quad (3)$$

The monotonic condition is guaranteed by a mixing network with nonnegative weights, which in turn guarantees Equation 2. Massive

challenging MARL tasks, such as StarCraft II, demonstrate the monotonic decomposition form is simple but effective as it can be performed in $O(n|\mathcal{U}|)$ time as opposed to $O(|\mathcal{U}|^n)$.

2.3 Maximum Entropy RL

Maximum Entropy Learning is an active research area in recent years and it has been applied in many aspects [33, 43]. Different with the standard reinforcement learning, which maximizes the expected sum of rewards $\sum_{t=0}^{T}\mathbb{E}_{(o_t,u_t)\sim\tau_\pi}[r(o_t,u_t)]$, the maximum entropy policy favors stochastic policies by augmenting the objective with the expected entropy of the policy

$$J(\pi) = \sum_{t=0}^{T}\mathbb{E}_{(o_t,u_t)\sim\tau_\pi}\left[r(o_t,u_t) + \alpha\mathcal{H}(\pi(\cdot\mid o_t))\right], \quad (4)$$

where $\mathcal{H}(\pi(\cdot\mid o_t))$ denotes the entropy of the policy, which implies the randomness of the policy. The temperature parameter α determines the relative importance of the entropy term against the reward. Maximum entropy policy calculates the soft Q-value iteratively with the entropy-regularized expected return objective as follows:

$$Q_{\text{soft}}(o_t,u_t) = r(o_t,u_t) + \gamma\mathbb{E}_{o_{t+1}\sim p}\left[V_{\text{soft}}(o_{t+1})\right], \quad (5)$$

$$V_{\text{soft}}(o_t) = \mathbb{E}_{u_t\sim\pi}\left[Q_{\text{soft}}(o_t,u_t) - \alpha\log\pi(u_t\mid o_t)\right]. \quad (6)$$

According to the learned Q-function in Equation 5, maximum entropy policy is directly updated for each state by minimizing the following objective

$$J(\pi) = \mathbb{E}_{o_t\sim\tau_\pi}\left[D_{KL}\left(\pi(\cdot\mid o_t)\parallel\frac{\exp\left(\frac{1}{\alpha}Q_{\text{soft}}(o_t,\cdot)\right)}{Z_{\text{soft}}(o_t)}\right)\right]$$

$$= \mathbb{E}_{o_t\sim\tau_\pi,\epsilon_t\sim\mathcal{N}}\left[\log\pi\left(f(\epsilon_t;o_t)\mid o_t\right) - \frac{1}{\alpha}Q_{\text{soft}}\left(o_t,f(\epsilon_t;o_t)\right)\right],$$

where $Z_{\text{soft}}(o_t)$ denotes the partition function and ϵ_t represents the noise vector sampled in Gaussian distribution $\mathcal{N}$. The policy is reparameterized [37] by a neural network transformation $u_t = f(\epsilon_t;o_t)$. The above iteration process leads to an improved policy in terms of the soft value function [13].

2.4 Collaborative Exploration in MARL

Exploration is necessary due to the long horizons and limited or delayed reward signals in complex tasks. In general, current exploration methods in single-agent RL mainly include heuristics methods [15], such as ϵ-greedy and count-based methods [3, 32]. However, most algorithms for statistically efficient RL are not computationally tractable in complex environments [26]. To deal with this problem, MAVEN introduces a latent space for hierarchical control, which would be fixed over an entire episode. Each joint action-value function can be thought of as a mode of joint exploratory behavior [21]. Unfortunately, it is extremely hard to guarantee the validness and diversity of the learned latent space.

3 ANALYSIS OF INTERACTION IN MARL

Current MARL algorithms pay little attention to the interaction between agents. For example, individual exploration and individual action-value function estimation in QMIX. In this section, we

analyze the case that the policy learned by QMIX cannot represent the true optimal action-value function as well as cannot be the optimal policy. A simple example is given by the payoff matrix of the two-player three-action matrix game, which is shown in Table 1.

u_2 \ u_1	**A**	B	C
A	**8**	-12	-12
B	-12	0	0
C	-12	0	0

Table 1: Payoff of matrix game

Q_2 \ Q_1	-8.9(A)	-0.6(B)	**-0.5(C)**
-8.0(A)	-8.06	-8.05	-8.05
0.1(B)	-8.05	-0.01	-0.01
0.2(C)	-8.05	-0.02	**-0.01**

Table 2: Individual and joint action-value of QMIX

According to the learned values in Table 2, the Boltzmann policy of QMIX is shown in Table 3 according to

$$P(u_{i,j}) = \frac{\exp(Q_{i,j}/\alpha)}{\sum_{i,j}\exp(Q_{i,j}/\alpha)}. \quad (7)$$

However, experimental result indicates that QMIX learns a suboptimal policy under uniform visitation. To solve this problem, several methods choose to communicate self-state information with other agents [8, 41]. But this does not work in the grid task as there is no state-information. Meanwhile, we find out that if each agent knows the actions of other agents in advance and use them as part of the state, we can accurately get the optimal solution, which is shown in Table 4. For the convenience of expression, we name this method as QMIXs. For details of it, please refer to the Appendix ??. Unfortunately, interacting action-information violates the Dec-POMDP framework where all agents choose actions simultaneously. For this reason, we propose an novel MARL algorithm modeling the interaction between agents under the Dec-POMDP framework in Section 4.

u_2 \ u_1	A	B	C
A	0.01	0.02	0.02
B	0.02	0.23	0.23
C	0.01	0.23	0.23

Table 3: Policy of QMIX

u_2 \ u_1	**A**	B	C
A	**0.76**	0.01	0.01
B	0.01	0.05	0.05
C	0.01	0.05	0.05

Table 4: Policy of QMIXs

4 AN INTERACTIVE APPROACH FOR MARL

In this section, we analyze the necessity for modeling the interaction between agents and the key challenges of the maximum entropy MARL. Based on these challenges, we introduce the novel cooperative MARL algorithm, **Interactive Actor-Critic**, which is shown in Figure 1. The complete algorithm is illustrated in Algorithm 1.
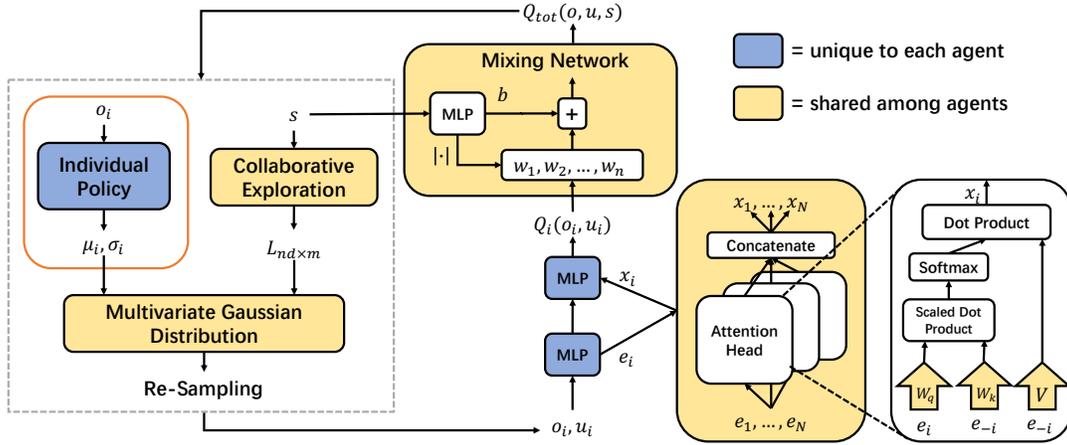
Figure 1: The structure of IAC. The leftmost box denotes the collaborative exploration module. Attention-QMIX Structure is shown on the right-hand side. During the centralized training, all the boxes are activated with shared information. During the decentralized execution, agents observe local information and only the orange boxes are activated.

4.1 Collaborative Exploration Module

Individual exploration in single-agent RL, such as increasing per-agent exploration rate ϵ, can help exploration. However, individual exploration may lead to provably poor exploration and suboptimality in MARL [21], which is illustrated by the simple example in Section 3. As the superior exploration ability of the maximum entropy policy discussed in the Section 1, a natural idea is extending the maximum entropy policy to MARL algorithms. The modified objective with the expected entropy of the joint stochastic policy is defined as

$$\boldsymbol{\pi}^* = \arg\max_{\boldsymbol{\pi}} \sum_{t=0}^{T} \mathbb{E}_{(\boldsymbol{o}_t, \boldsymbol{u}_t) \sim \tau_{\boldsymbol{\pi}}} \left[r(\boldsymbol{o}_t, \boldsymbol{u}_t) + \alpha \mathcal{H}(\boldsymbol{\pi}(\cdot \mid s_t)) \right]. \quad (8)$$

With the joint action-value function, the joint stochastic policy is updated as follows:

$$\boldsymbol{\pi}_{\text{new}} = \arg\min_{\boldsymbol{\pi}'} D_{KL} \left(\boldsymbol{\pi}'(\cdot \mid \boldsymbol{o}_t) \; \middle\| \; \frac{\exp\left(\frac{1}{\alpha} Q_{\text{tot}}^{\boldsymbol{\pi}_{\text{old}}}(\boldsymbol{o}_t, \cdot) \right)}{Z_{\text{tot}}^{\boldsymbol{\pi}_{\text{old}}}(\boldsymbol{o}_t)} \right). \quad (9)$$

The above iteration is the standard multi-agent maximum entropy policy, and it mainly follows along the standard Soft Q-Learning theorem [12]. Following the proof of soft policy iteration in SAC [13], this standard maximum entropy policy in MARL inherits the convergence property for the joint stochastic policy iteration, which is formally stated in Theorem 1.

THEOREM 1. *Let $\boldsymbol{\pi}_{\text{old}}$ and $\boldsymbol{\pi}_{\text{new}}$ denote the joint stochastic policy, which is the optimizer of the minimization problem in Equation 9. The joint action-value function has a joint stochastic policy improvement $Q_{\text{tot}}^{\boldsymbol{\pi}_{\text{new}}}(\boldsymbol{o}_t, \boldsymbol{u}_t) \geq Q_{\text{tot}}^{\boldsymbol{\pi}_{\text{old}}}(\boldsymbol{o}_t, \boldsymbol{u}_t)$ for all $(\boldsymbol{o}_t, \boldsymbol{u}_t) \in O \times \mathcal{U}$ with $|\mathcal{U}| < \infty$. The joint stochastic policy $\boldsymbol{\pi}$ could converge to $\boldsymbol{\pi}^*$ such that $Q_{\text{tot}}^{\boldsymbol{\pi}^*}(\boldsymbol{o}_t, \boldsymbol{u}_t) \geq Q_{\text{tot}}^{\boldsymbol{\pi}}(\boldsymbol{o}_t, \boldsymbol{u}_t)$ for all $\boldsymbol{\pi} \in \Pi$, where Π denotes the set of policies.*

The standard maximum entropy MARL method learns the joint stochastic policy $\boldsymbol{\pi}(\boldsymbol{u}_t \mid \boldsymbol{o}_t)$ while the individual policy $\pi_i(u_i \mid o_i)$ is unavailable, which violates the CTDE framework. To deal with the

issue, we attempt to use the multivariate Gaussian distribution $\mathcal{N}_M$ to model the interaction between individual policies because the behavioral strategies reflect agents' cooperation relationship [18]. Let d denote the dimension of the action space for each agent and Σ denote the covariance matrix of the multivariate Gaussian distribution. The probability density function of the distribution $\mathcal{N}_M$ is calculated as follows:

$$f(x) = \frac{1}{\left(\sqrt{2\pi} \right)^{nd} |\Sigma|^{\frac{1}{2}}} \exp \left\{ -\frac{(x - \mu_x)^T (\Sigma)^{-1} (x - \mu_x)}{2} \right\}, \quad (10)$$

where x is a random variable that represents actions sampled from the multivariate Gaussian distribution. μ_x indicates the mean of x, which represents the output of individual policies. When the individual policies are independent, the off-diagonal elements of the covariance matrix Σ are $\boldsymbol{0}$. The joint action-value function can be simplified to

$$Q_{\text{tot}} = \sum_{t=0}^{T} \mathbb{E}_{(\boldsymbol{o}_t, \boldsymbol{u}_t) \sim \tau_{\boldsymbol{\pi}}} \left[r(\boldsymbol{o}_t, \boldsymbol{u}_t) - \alpha \sum_{i=1}^{n} \log \pi_i(\cdot \mid o_t^i) \right]. \quad (11)$$

Otherwise, the covariance matrix Σ is a real symmetric matrix, which reflects the relationship between behavioral strategies of agents. When the agent number increases largely, it is hard to approximate the value of Σ due to the exponential growth of interactions and dimensionality. Therefore, we replace $\mathcal{N}_M$ with the *low rank multivariate Gaussian distribution* $\mathcal{N}_{LM}$ and decompose the covariance matrix $\Sigma_{nd \times nd}$ as follows:

$$\Sigma_{nd \times nd} = L_{nd \times m} L_{nd \times m}^T + D_{nd \times nd}, \quad (12)$$

where n is the number of agents and $L_{nd \times m}$ denotes the covariance factor. $D_{nd \times nd}$ denotes the covariance diagonal matrix. $L_{nd \times m}$ is named the *collaborative exploration matrix*, which is approximated with a shared neural network

$$L_{nd \times m} = f_\psi(s_t; \psi), \quad (13)$$

where $f_\psi(s)$ is parameterized by ψ and the input of the shared neural network is global state information s. The joint stochastic policy is reparameterized as follows

$$
\begin{aligned}
\boldsymbol{u}_t &= p_{\theta,\psi}(\boldsymbol{\epsilon}_N, \boldsymbol{\epsilon}_M; \boldsymbol{o}_t, s_t) \\
&= f_\theta(\boldsymbol{\epsilon}_N; \boldsymbol{o}_t) + f_\psi(\boldsymbol{\epsilon}_M; s_t) \\
&= \boldsymbol{\mu}_t + \boldsymbol{\sigma}_t \boldsymbol{\epsilon}_N + L_{nd \times m} \boldsymbol{\epsilon}_M,
\end{aligned}
\tag{14}
$$

where $p_{\theta,\psi}$ denotes the joint stochastic policy network composed of the individual policy network f_θ the collaborative exploration network f_ψ. $\boldsymbol{u} = [\mu_1, \ldots, \mu_n]$, $\boldsymbol{\sigma} = [\sigma_1, \ldots, \sigma_n]$ are the vectors, where μ_i, σ_i are the outputs of the individual policies $\pi_i(o_i; \theta_i)$. $\boldsymbol{\epsilon}_N$ denotes the individual Gaussian noise and $\boldsymbol{\epsilon}_M$ denotes the collaborative Gaussian noise.

While executing, we set the individual Gaussian noise $\boldsymbol{\epsilon}_N$ and the collaborative Gaussian noise to $\boldsymbol{0}$. Each agent only applies their individual information to make decisions. The collaborative exploration policy only works in the training phase, which meets the requirement of the CTDE framework. Sine the joint stochastic action $\boldsymbol{u}$ is resampled from the low rank multivariate Gaussian distribution $\mathcal{N}_{LM}$, the entropy of the joint stochastic policy $\mathcal{H}(\boldsymbol{\pi}(\cdot \mid s_t))$ can be calculated according to the Equation 10. The policy parameters are learned by minimizing the following equation

$$
\begin{aligned}
J_{\boldsymbol{\pi}}(\theta, \psi) &= \mathbb{E}_{\tau \sim \boldsymbol{\pi}}\left[D_{KL}\left(\boldsymbol{\pi}_{\theta,\psi}(\cdot \mid \boldsymbol{o}_t, s_t) \;\middle\|\; \frac{\exp\left(\frac{1}{\alpha}Q^{\boldsymbol{\pi}_{\text{old}}}(\boldsymbol{o}_t, \cdot)\right)}{Z^{\boldsymbol{\pi}_{\text{old}}}(\boldsymbol{o}_t)}\right)\right] \\
&= \mathbb{E}_{(s_t,\boldsymbol{o}_t)\sim\tau_{\boldsymbol{\pi}}, \boldsymbol{\epsilon}_N, \boldsymbol{\epsilon}_M \sim \mathcal{N}}[\log \boldsymbol{\pi}_{\theta,\psi}(p_{\theta,\psi}(\boldsymbol{\epsilon}_N, \boldsymbol{\epsilon}_M; \boldsymbol{o}_t, s_t) \mid \boldsymbol{o}_t, s_t) - \\
&\qquad \frac{1}{\alpha}Q_{\text{tot}}(\boldsymbol{o}_t, p_{\theta,\psi}(\boldsymbol{\epsilon}_N, \boldsymbol{\epsilon}_M; \boldsymbol{o}_t, s_t))].
\end{aligned}
\tag{15}
$$

where Q_{tot} is the joint action-value function, which is discussed in detail in Subsection 4.2.

4.2 Mutual Value Function Information

As discussed in Subsection 2.2, the joint action-value function Q_{tot} can be factorized by $Q_i(o_i, u_i)$ according to the IGM condition, while the decomposition methods have been criticized for the limited representative ability of the joint action-value function. Therefore, we propose a new optimization objective, which maximizes the joint action-value function $Q_{\text{tot}}(\boldsymbol{o}_t, \boldsymbol{u}_t)$ by maximizing the individual action-value function $Q_i(o_t^i, u_t^i)$ and the cooperative action-value function $Q_{ij}(o_t^i, o_t^j, u_t^i, u_t^j)$

$$
Q_{\text{tot}}(\boldsymbol{o}_t, \boldsymbol{u}_t) = \sum_{i=0}^{n} Q_i(o_t^i, u_t^i) + \sum_{i=0}^{n}\sum_{j \neq i} Q_{ij}(o_t^i, o_t^j, u_t^i, u_t^j),
\tag{16}
$$

$$
\pi_i(\boldsymbol{o}_t) = \arg\max_{u_i}\left\{Q_i(o_t^i, u_t^i) + \sum_{j \neq i} \mathbb{E}_{u_j \sim \bar{\pi}_j}\left[Q_{ij}(o_t^i, o_t^j, u_t^i, u_t^j)\right]\right\},
\tag{17}
$$

where $\bar{\pi}_j$ denotes the fixed policy of agent j. We name the new optimization objective as mutual value function information. Based on the new objective, the IGM condition in Subsection 2.2 is modified

Algorithm 1 Interactive Actor-Critic (IAC)

1: Initialize parameter vectors $\theta, \psi, \vartheta, \phi$
2: Initialize replay buffer $\mathcal{D}$, parameters update rate T, training times t
3: **for** each episode **do**
4:     Reset environment and receive s, $\boldsymbol{o}$
5:     **for** each step in episode **do**
6:        Select actions $\boldsymbol{u} \sim \boldsymbol{\pi}(\cdot \mid \boldsymbol{o}, s)$ according to (14)
7:        Send $\boldsymbol{u}$ to environment and receive $\boldsymbol{o}', s', r$
8:        Store transitions in replay buffer $\mathcal{D}$
9:        **if** $|\mathcal{D}| > N_{\text{Batch}}$ **then**
10:          Sample minibatch B
11:          Calculate $\boldsymbol{u} \sim \boldsymbol{\pi}_\theta(\boldsymbol{o}_B, s_B)$ according to (14)
12:          Calculate $Q_{\text{tot}}(\boldsymbol{o}_B, \boldsymbol{u}, s_B; \vartheta, \phi)$ according to (22)
13:          Update the joint-policy according to (15)
14:          Calculate $Q_{\text{tot}}(\boldsymbol{o}_B, \boldsymbol{u}_B, s_B; \vartheta, \phi)$ according to (22)
15:          Calculate $\boldsymbol{u}' \sim \boldsymbol{\pi}_{\theta^-}(\boldsymbol{o}'_B, s_B)$ according to (14)
16:          Calculate $Q_{\text{tot}}(\boldsymbol{o}'_B, \boldsymbol{u}', s'_B; \vartheta^-, \phi^-)$ according to (22)
17:          Update critics using $\nabla L_Q(\vartheta, \phi)$ according to (23)
18:        **end if**
19:        **if** $t \% T = 0$ **then**
20:          Update target parameters:
          $\theta^- \leftarrow \theta, \psi^- \leftarrow \psi, \vartheta^- \leftarrow \vartheta, \phi^- \leftarrow \phi$
21:        **end if**
22:     **end for**
23: **end for**

as

$$
\arg\max_{\boldsymbol{u}} Q_{\text{tot}}^{\boldsymbol{\pi}}(\boldsymbol{o}, \boldsymbol{u}) = \left(
\begin{array}{c}
\arg\max_{u_1}\left\{Q_1(o_1, u_1) + \sum_{j \neq 1} \mathbb{E}\left[Q_{1j}\right]\right\} \\
\vdots \\
\arg\max_{u_n}\left\{Q_n(o_n, u_n) + \sum_{j \neq n} \mathbb{E}\left[Q_{nj}\right]\right\}
\end{array}
\right).
\tag{18}
$$

However, accurately calculating the mutual value function is extremely hard due to the uncertainty of the cooperative relationship between agents, which is the core problem of MARL. We adopt the shared attention mechanism to approximately meet the condition in Equation 18. The individual action-value function $Q_i^\phi(\boldsymbol{o}_t, \boldsymbol{u}_t)$ is calculated according to all observation-action information of agents

$$
Q_i^\phi(\boldsymbol{o}_t, \boldsymbol{u}_t) = f_i(g_i(o_t^i, u_t^i), x_t^i),
\tag{19}
$$

$$
x_t^i = \sum_{j \neq i} \alpha_t^j h(V g_j(o_t^j, u_t^j)),
\tag{20}
$$

where f_i, g_i, h denote multi-layer perceptions and x_i denotes the weighted sum of per-agent action-values. The matrix V is shared across agents. Individual action-value functions are parameterized by $\phi = \langle \phi_1, \phi_2, \ldots, \phi_n \rangle$. We apply the query-key system to compute the attention weight α_j by comparing the embedding e_j

$$
\alpha_j \propto \exp\left(e_j^T W_k^T W_q e_i\right),
\tag{21}
$$

where $e_i = g_i(o_i, u_i)$, W_k transforms e_j into a key and W_q transforms e_i into a query. As for the implementation issue, we adopt a multi-head attention mechanism to select the different weighted mixture of per-agent action-values. The same mixing network as

(a) DDPG-MIX. **(b) Attn-QMIX.** **(c) IAC.**
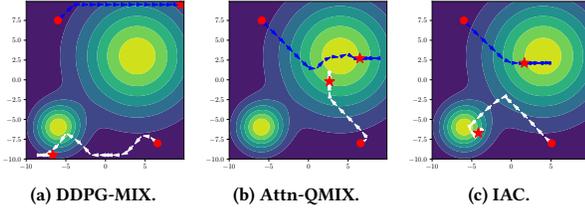
Figure 2: Visualization of value distributions and trajectories of various methods. Dotted lines denote trajectories generated by different algorithms. The red circle is the starting point and the red star is the ending point.

QMIX is adopted to calculate the joint action-value function Q_{tot}

$$Q_{\text{tot}}(\boldsymbol{o}_t, \boldsymbol{u}_t, s_t; \vartheta, \phi) = \text{MIX}\left(Q_1^\phi(\boldsymbol{o}_t, \boldsymbol{u}_t), ..., Q_n^\phi(\boldsymbol{o}_t, \boldsymbol{u}_t), s_t; \vartheta\right), \quad (22)$$

where MIX denotes the mixing network parameterized by ϑ. The attention-based joint action-value network is trained end-to-end by minimizing the following loss

$$L_Q(\vartheta, \phi) = \sum_{i=1}^{|b|}\left[\left(y_i^{\text{tot}} - Q_{\text{tot}}(\boldsymbol{o}_t, \boldsymbol{u}_t, s_t; \vartheta, \phi)\right)^2\right], \quad (23)$$

$$y^{\text{tot}} = r + \gamma Q_{\text{tot}}(\boldsymbol{o}_{t+1}, \boldsymbol{u}_{t+1}, s_{t+1}; \vartheta^-, \phi^-)\mid_{\boldsymbol{u}_{t+1}=\boldsymbol{\pi}^-(\boldsymbol{o}_{t+1}, s_{t+1})}, \quad (24)$$

where b is the batch sampled from the replay buffer and $\boldsymbol{\pi}^-$ denotes the target joint stochastic policy. The target mixing network, individual action-value network are parameterized respectively by ϑ^-, ϕ^-.

5 EXPERIMENTS

To evaluate our algorithm, we design a series of multi-agent cooperative experiments to compare IAC with the state-of-the-art MARL algorithms. Hyper-parameters and implementation details are listed in Subsection 5.1 and Appendix **??**. The state-of-the-art MARL algorithms are represented in Subsection 5.2. The multi-agent cooperative experiments are shown in Appendix **??**. The performance comparison between different methods and detailed analysis are discussed in Subsection 5.3.

5.1 Setting

Simple World. We design a task named *Simple World* to evaluate the exploration ability of the collaborative exploration module and representative ability of Attention-based QMIX. Simple World is a two-player one-action one-step task, which has no state and will return the reward according to taken actions. For convenience, we adopt two independent Gaussian distribution to represent the value distributions. The means are $[4, 3]$ and $[-6, -6]$. The variances are 5 and 2 respectively. There two value distributions denote the reward distributions in the task.

Multi-Pendulum. In this subsection, we extend the pendulum task from single-agent to multi-agent tasks. Single-pendulum in the gym is a classic benchmark. In detail, we aim at swinging up a pendulum starting from a random position and keeping it upright. Compared with the single-pendulum task, the multi-agent classic control task has multiple pendulums to be controlled. Each pendulum only

(a) Independent policy distribution. **(b) Collaborative policy distribution.**
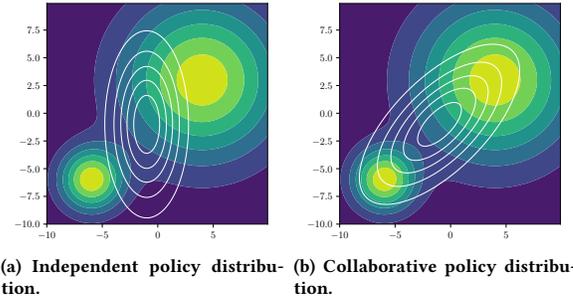
Figure 3: Visualization of value distributions and policy distributions. Colored area denotes independent value distributions. Bright color area indicates high reward. White circle on (a) represents the independent policy distribution. White circle on (b) represents the collaborative policy distribution.

receives the self-observation and the shared reward, which indicates the multi-pendulum environment is a Dec-POMDP task. Moreover, the reward information is sparse in the environment. When all angles of pendulums are less than 0.2, the immediate reward is +5. Otherwise, all agents receive reward 0.

Predator-prey is a classic multi-agent cooperative benchmark. In the predator-prey game, N slower cooperating agents chase a faster adversary around a randomly generated environment with L large landmarks. We set three cooperating agents, two landmarks and one adversary. The adversary moves randomly. When the cooperative agents collide with the adversary, these cooperative agents receive a shared reward +10. Otherwise, the immediate reward is 0. Predator-prey is a Dec-MDP task as agents can observe the relative positions and velocities of other agents and landmarks at all times.

PO-Predator-prey is an enhanced version of the predator-prey. Different from the predator-prey task, agents in the PO-Predator-prey task can only receive the information within the distance of 0.8. Other settings, such as the number of agents and the reward function, are the same as the predator-prey task.

5.2 Baselines

We compare our algorithms with the state-of-the-art approaches: MADDPG and MAAC, which are decentralized policies of centralized training. To ensure the accuracy of the reproduction, we apply the official code[1].

Since improvements of IAC come from both actor and critic modules, we set different baselines to evaluate their performances respectively. We combine the value decomposition methods and the multi-agent deterministic policy. This basic approach is named **DDPG-MIX**. We replace respectively the exploration policy and attention module in IAC to generate the corresponding baselines: Soft-QMIX (**SQ**) and Attention-QMIX (**Attn-QMIX**).

As a comparison for maximum entropy policy, a naive method to solve the problem in Subsection 4.1 is defining the soft individual

---

[1]https://github.com/openai/maddpg, https://github.com/shariqiqbal2810/MAAC

(a) Average rewards in multi-pendulums environment.

(b) Average rewards in Predator-prey task.

(c) Average rewards in PO-Predator-prey task.
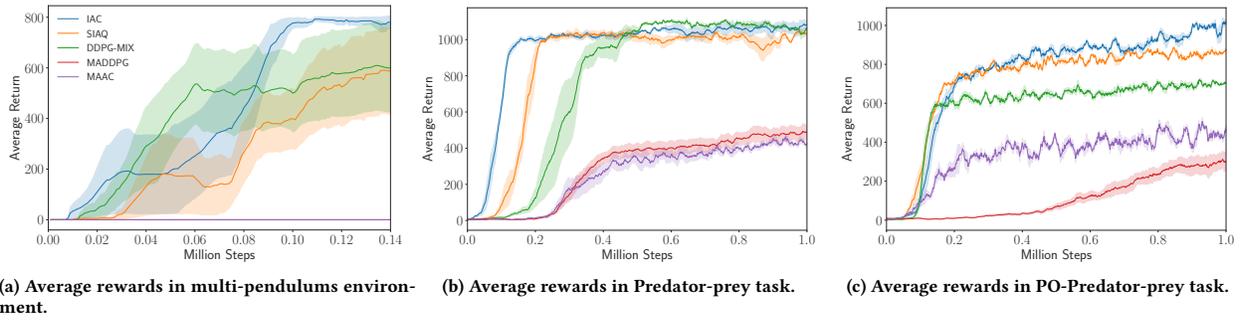
Figure 4: Evaluation of various methods in Classic Control, Predator-prey and POMDP-predator-prey tasks. IAC achieves better performance in all tasks compared with other baselines.

action-value function:

$$Q_i = \sum_{t=0}^{T} \mathbb{E}_{(o_t^i, u_t^i) \sim \tau_{\pi_i}} \left[r(o_t^i, u_t^i) + \alpha \mathcal{H}(\pi_i(\cdot \mid o_t^i)) \right]. \quad (25)$$

Mixing Network in Section 4.2 calculates the joint action-value function Q_{tot} by approximating weight ϑ in neurual network

$$Q_{\text{tot}} = \sum_{i=1}^{n} \text{MIX} \left(Q_i(o_t^i, u_t^i); \vartheta \right)$$

$$= \sum_{i=1}^{n} \text{MIX} \left(\sum_{t=0}^{T} \mathbb{E}_{(o_t^i, u_t^i) \sim \tau_{\pi_i}} \left[r(o_t^i, u_t^i) + \alpha \mathcal{H}(\pi_i(\cdot \mid o_t^i)) \right]; \vartheta \right), \quad (26)$$

where MIX denotes the Mixing Network, such as the QMIX Network. Based on individual policies, the entropy of the joint stochastic policy is $\sum_{i=1}^{n} \text{MIX} \left(\sum_{t=0}^{T} \mathbb{E}_{(o_t^i, u_t^i) \sim \tau_{\pi_i}} \left[\mathcal{H}(\pi_i(\cdot \mid o_t^i)) \right]; \vartheta \right)$. We name this method as Soft Individual QMIX (**SIQ**), which has the similar critic structure as IAC but different exploration module. Moreover, we adopt the same attention-based critic module in SIQ to generate the corresponding baseline: Soft-Individual-Attention-QMIX (**SIAQ**). All the above methods use the same hyperparameters and networks as IAC.

5.3 Results

Simple World. We evaluate DDPG-MIX, Attn-QMIX and IAC in the task, which is illustrated in the Figure 2. Experimental results indicate that the Attn-QMIX has a stronger representative ability compared with QMIX. Moreover, compared with the individual exploration, the collaborative exploration module in IAC promotes to explore the more meaningful area. The experimental result can be explained visually in Figure 3, where we consider two different exploration policies: the independent policy and the collaborative policy. Both the independent policy and collaborative policy are denoted by a multivariate Gaussian distribution with the same mean $[-1, -1]$. The covariance matrix of the independent policy is $\Sigma = [4, 0; 0, 20]$. The covariance matrix of the collaborative policy is $\Sigma = [15, 10; 10, 15]$. The independent policy distribution has some meaningless areas, which induces useless exploration. On the contrary, the collaborative policy distribution prefers to explore high

(a) Visualization at the 1st step (b) Visualization at the 20th step

**(c) Visualization at the 40th step (d) Visualization at the 60th step**
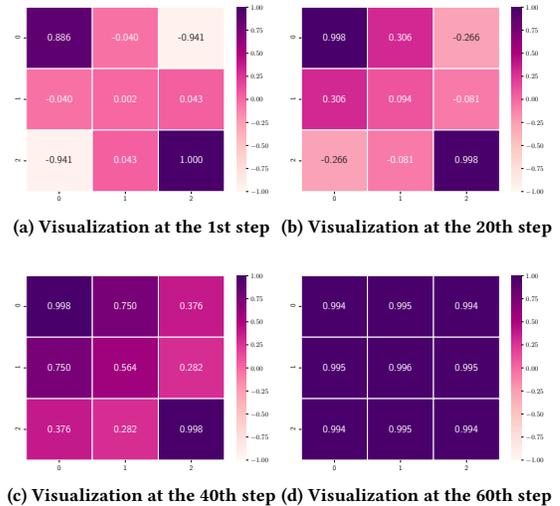
Figure 5: Visualization of the covariance matrix in IAC at different steps in the multi-pendulum task. We set the number of the pendulums is 3. The covariance matrix indicates the relationship of behavior strategies. Values in the covariance matrix change from irregular to regular, which indicates that there is a clear relationship between agents.

reward areas, which indicates that the collaborative exploration is more effective compared with the independent exploration.

Multi-Pendulum. The experimental results are shown in Figure 4. The improvements of IAC, SIAQ, and DDPG-MIX are apparent compared with MADDPG and MAAC, which indicates the superiority of value decomposition methods. We visualize the covariance matrix Σ in IAC at different training steps, which is shown in Figure 5. At the 1st step, the covariance matrix consists of irregular values. The policy of the first agent is positively related to itself and negatively related to the third agent. Gradually, the relationship of agent policies is positive. Finally, the relationship of agent policies is stable. In the multi-pendulum task, agents have the same objective and
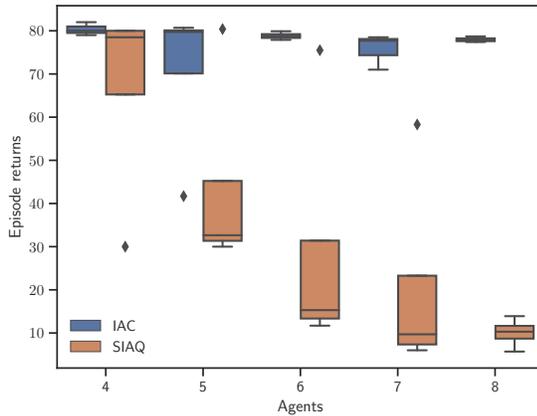
Figure 6: Scalability performance in the multi-pendulum task. When all angles of the pendulums are less than 0.2, the immediate reward is +0.5. When there are one or three pendulums meeting the angle requirement, the immediate rewards are +0.1 and +0.2 respectively.

(a) Saved Steps.   (b) Diff-Return.

Figure 7: The performance comparison between IAC, SIAQ and baselines. Assume agents controlled by DDPG-MIX capture prey cooperatively in the Predator-prey task at time t_1. Other methods achieve the same objective at time t_i. The saved steps is $\Delta t = t_1 - t_i$, and the unit is one thousand step. The diff-return is the episode return difference between IAC, SIAQ and DDPG-MIX in the PO-Predator-prey task.

their policies should be positively related, which has been satisfied by the visualization result.

Moreover, we evaluate the scalability of IAC and SIAQ in multi-pendulum task. As the number of agents grows, rewards in this task are extremely sparse. To reduce the exploration difficulty, agents receive few rewards when a few pendulums meet the angle requirement, which is illutrated in Figure 6. Experimental results illustrate that the performance of IAC does not deteriorate when pendulums are increased, while the performance of SIAQ is not scalable.

Predator-prey and PO-Predator-prey. Various algorithms are evaluated in Predator-prey and PO-Predator-prey tasks, whose results are illustrated in the Figure 4. Experimental results indicate that value-based methods achieve better performance than MAD-DPG and MAAC. Moreover, collaborative exploration module has superior performance than the deterministic policy. Compared with other methods, our proposed algorithm (IAC) is competitive in both environments. As the reward setting in the Predator-prey task is similar to the multi-pendulum task, MADDPG and MAAC have limited capacity to find the optimal individual action-value function for each agent. We set the average reward threshold to 1000, where three predators can succeed to capture prey cooperatively. Compared with DDPG-MIX and SIAQ, IAC saves 0.295 and 0.06 million steps respectively to complete the predator-task.

In the PO-Predator-prey task, agents only receive the local information within 0.8 distance. Experimental results indicate that the performance of MADDPG, DDPG-MIX and SIAQ have decreased. As the MAAC applies the attention mechanism to calculate the individual action-value function, the performance of MAAC is maintained. Compared with these methods, IAC still achieves better robustness and succeeds to capture prey cooperatively. Since IAC and SIAQ have been modified on the policy and value-function estimation sides, the respect performance analysis is necessary, which is represented in Figure 7. Compared with SQ and SIQ, IAC and SIAQ achieve better performance in terms of the saved time

and episode reward. Therefore, both the collaborative exploration policy and shared attention mechanism have a positive influence on the performance.

6 CONCLUSION

In this paper, we propose a novel cooperative MARL algorithm named IAC to model the interaction between agents. IAC adopts the collaborative exploration module for the policy to couple the individual policies and the joint-policy of agents, which is trained by maximizing the entropy-regularized expected return. For the value estimation, IAC utilizes a shared attention mechanism to estimate the mutual value function, which considers the impact of the teammates. Moreover, IAC extends the value decomposition methods to continuous control tasks. Taking the advantages for both sides of collaborative exploration and mutual value function, IAC not only outperforms existing baselines but also achieves superior cooperation, robustness and scalability.

In the future, we will extend our work to large-scale multi-agent tasks liking MAgents [10, 39], because AI multi-agent system gradually contains hundreds of agents, such as the Warehouse Robot System and Bee colony system. These large-scale multi-agent systems have more complex interaction and tasks. We believe that solving modeling interaction in multi-agent systems would be one-step toward Multi-Agent AI.

REFERENCES

[1] Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. 2019. Understanding the impact of entropy on policy optimization. In *International Conference on Machine Learning*. PMLR, 151–160.

[2] Ranwa Al Mallah, Alejandro Quintero, and Bilal Farooq. 2019. Cooperative evaluation of the cause of urban traffic congestion via connected vehicles. *IEEE Transactions on Intelligent Transportation Systems* 21, 1 (2019), 59–67.

[3] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. 2016. Unifying count-based exploration and intrinsic motivation. In *Advances in neural information processing systems*. 1471–1479.

[4] Dimitri Bertsekas. 2020. Multiagent Value Iteration Algorithms in Dynamic Programming and Reinforcement Learning. *arXiv preprint arXiv:2005.01627* (2020).

[5] Lucas Cassano, Kun Yuan, and Ali H Sayed. 2018. Multi-Agent Fully Decentralized Value Function Learning with Linear Convergence Rates. *arXiv preprint arXiv:1810.07792* (2018).

[6] Yong Cheng. 2019. Semi-supervised learning for neural machine translation. In *Joint Training for Neural Machine Translation*. Springer, 25–40.

[7] Tianshu Chu, Sandeep Chinchali, and Sachin Katti. 2020. Multi-agent Reinforcement Learning for Networked System Control. *arXiv preprint arXiv:2004.01339* (2020).

[8] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. 2019. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning*. 1538–1546.

[9] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. 2017. Counterfactual multi-agent policy gradients. *arXiv preprint arXiv:1705.08926* (2017).

[10] Sriram Ganapathi Subramanian, Pascal Poupart, Matthew E Taylor, and Nidhi Hegde. 2020. Multi Type Mean Field Reinforcement Learning. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*. 411–419.

[11] Jordi Grau-Moya, Felix Leibfried, and Haitham Bou-Ammar. 2018. Balancing two-player stochastic games with soft Q-learning. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 268–274.

[12] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. 2017. Reinforcement Learning with Deep Energy-Based Policies. In *International Conference on Machine Learning*. 1352–1361.

[13] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *International Conference on Machine Learning*. 1861–1870.

[14] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. 2019. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems* 6, 33 (2019), 750–797.

[15] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. 2016. Vime: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems*. 1109–1117.

[16] Shariq Iqbal, Christian A Schroeder de Witt, Bei Peng, Wendelin Böhmer, Shimon Whiteson, and Fei Sha. 2020. AI-QMIX: Attention and Imagination for Dynamic Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:2006.04222* (2020).

[17] Shariq Iqbal and Fei Sha. 2019. Actor-attention-critic for multi-agent reinforcement learning. In *International Conference on Machine Learning*. 2961–2970.

[18] Minghuan Liu, Ming Zhou, Weinan Zhang, Yuzheng Zhuang, Jun Wang, Wulong Liu, and Yong Yu. 2020. Multi-Agent Interactions Modeling with Correlated Policies. *arXiv preprint arXiv:2001.03415* (2020).

[19] Qian Long, Zihan Zhou, Abhibav Gupta, Fei Fang, Yi Wu, and Xiaolong Wang. 2020. Evolutionary Population Curriculum for Scaling Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:2003.10423* (2020).

[20] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*. 6379–6390.

[21] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. 2019. Maven: Multi-agent variational exploration. In *Advances in Neural Information Processing Systems*. 7613–7624.

[22] Hangyu Mao, Zhengchao Zhang, Zhen Xiao, and Zhibo Gong. 2019. Modelling the Dynamic Joint Policy of Teammates with Attention Multi-agent DDPG. In *Proceedings of the 18th International Conference on Autonomous Agents and Multi-Agent Systems*. 1108–1116.

[23] Junhyuk Oh, Valliappa Chockalingam, Honglak Lee, et al. 2016. Control of Memory, Active Perception, and Action in Minecraft. In *International Conference on Machine Learning*. 2790–2799.

[24] Frans A Oliehoek, Christopher Amato, et al. 2016. *A concise introduction to decentralized POMDPs*. Vol. 1. Springer.

[25] Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. 2008. Optimal and Approximate Q-value Functions for Decentralized POMDPs. *Journal of Artificial Intelligence Research* 32 (2008), 289–353.

[26] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. 2016. Deep exploration via bootstrapped DQN. In *Advances in neural information processing systems*. 4026–4034.

[27] Tabish Rashid, Gregory Farquhar, Bei Peng, and Shimon Whiteson. 2020. Weighted QMIX: Expanding Monotonic Value Function Factorisation. *arXiv preprint arXiv:2006.10800* (2020).

[28] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. 4295–4304.

[29] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. 2019. QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. 5887–5896.

[30] Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z Leibo, Karl Tuyls, et al. 2017. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296* (2017).

[31] Quinlan Sykora, Mengye Ren, and Raquel Urtasun. 2020. Multi-agent routing value iteration network. *arXiv preprint arXiv:2007.05096* (2020).

[32] Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. 2017. # exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in neural information processing systems*. 2753–2762.

[33] Emanuel Todorov. 2008. General duality between optimal control and estimation. In *2008 47th IEEE Conference on Decision and Control*. IEEE, 4286–4292.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[35] Dirk B Walther and Christof Koch. 2007. Attention in hierarchical models of object recognition. *Progress in brain research* 165 (2007), 57–78.

[36] Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. 2020. Off-Policy Multi-Agent Decomposed Policy Gradients. *arXiv preprint arXiv:2007.12322* (2020).

[37] Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8, 3-4 (1992), 229–256.

[38] Yaodong Yang, Jianye Hao, Guangyong Chen, Hongyao Tang, Yingfeng Chen, Yujing Hu, Changjie Fan, and Zhongyu Wei. 2020. Q-value Path Decomposition for Deep Multiagent Reinforcement Learning. *arXiv preprint arXiv:2002.03950* (2020).

[39] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean Field Multi-Agent Reinforcement Learning. In *International Conference on Machine Learning*. 5571–5580.

[40] Kaiqing Zhang, Zhuoran Yang, Han Liu, Tong Zhang, and Tamer Basar. 2018. Fully Decentralized Multi-Agent Reinforcement Learning with Networked Agents. In *International Conference on Machine Learning*. 5872–5881.

[41] Sai Qian Zhang, Qi Zhang, and Jieyu Lin. 2019. Efficient communication in multi-agent reinforcement learning via variance based control. In *Advances in Neural Information Processing Systems*. 3235–3244.

[42] Zhi Zhang, Jiachen Yang, and Hongyuan Zha. 2020. Integrating Independent and Centralized Multi-agent Reinforcement Learning for Traffic Signal Network Optimization. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*. 2083–2085.

[43] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. 2008. Maximum entropy inverse reinforcement learning. In *AAAI*, Vol. 8. Chicago, IL, USA, 1433–1438.