EasyChair Preprint

№ 11117

MRRC: Multi-Agent Reinforcement Learning with Rectification Capability in Cooperative Tasks

Sheng Yu, Wei Zhu, Shuhong Liu, Zhengwen Gong and Haoran Chen

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

October 23, 2023

MRRC: Multi-Agent Reinforcement Learning with Rectification Capability in Cooperative Tasks

Sheng Yu, Wei Zhu(✉), Shuhong Liu, Zhengwen Gong, and Haoran Chen

School of Information and Communication, National University of Defense Technology, Wuhan 430014, China
yusheng17@nudt.edu.cn,zhuwei929@hotmail.com

Abstract. Motivated by the centralised training with decentralised execution (CTDE) paradigm, multi-agent reinforcement learning (MARL) algorithms have made significant strides in addressing cooperative tasks. However, the challenges of sparse environmental rewards and limited scalability have impeded further advancements in MARL. In response, MRRC, a novel actor-critic-based approach is proposed. MRRC tackles the sparse reward problem by equipping each agent with both an individual policy and a cooperative policy, harnessing the benefits of the individual policy's rapid convergence and the cooperative policy's global optimality. To enhance scalability, MRRC employs a monotonic mix network to rectify the state-action value function Q for each agent, yielding the joint value function Q_{tot} to facilitate global updates of the entire critic network. Additionally, the Gumbel-Softmax technique is introduced to rectify discrete actions, enabling MRRC to handle discrete tasks effectively. By comparing MRRC with advanced baseline algorithms in the "Predator-Prey" and challenging "SMAC" environments, as well as conducting ablation experiments, the superior performance of MRRC is demonstrated in this study. The experimental results reveal the efficacy of MRRC in reward-sparse environments and its ability to scale well with increasing numbers of agents.

Keywords: Multi-agent reinforcement learning · Cooperative task · Individual reward rectification · Monotonic mix function.

1 Introduction

In recent years, artificial intelligence (AI) technologies have been extensively applied in various domains such as medical services [3,15], multiplayer games [13,34], and image processing [4,25]. Within these domains, effectively handling multi-agent cooperative tasks has emerged as a crucial challenge. The advent of the centralised training with decentralised execution (CTDE) [11] paradigm has significantly advanced the field of multi-agent reinforcement learning (MARL) in addressing such tasks through centralized agent training. However, challenges arise due to the conditions under which the environment provides rewards to agents [19]. For instance, in games like Go, it is difficult to determine the winner until the game's completes, making it arduous for agents to obtain rewards during gameplay. Consequently, this hampers the convergence of the algorithm, increases the cost of sample acquisition, and diminishes the utilization of sample [1].

To tackle the sparse reward problem, researchers have proposed various methods, including reward function reconstruction [10,20] and multi-objective learning [7,23]. While reconstructing the reward function can partially alleviate the slow update issue caused by sparse rewards, it typically applies exclusively to specific tasks. Moreover, migrating to a new environment necessitates designing a novel reward function. Additionally, reconstructed reward functions occasionally misguide agents and require constant debugging to provide meaningful guidance, which reduces efficiency [5]. On the other hand, multi-objective learning involves augmenting the original learning goal, enabling agents to receive rewards upon reaching certain milestones. This approach expedites training in the initial stages, and the agent's policy improves through accumulated successful experiences. However, these additional rewards introduce significant errors that accumulate over time, ultimately impeding the attainment of a globally optimal solution [16].

As the number of agents participating in cooperative tasks grows, several MARL algorithms, including FOP [35], QPLEX [28], and MADDPG [12], encounter challenges related to dimensionality. These challenges, commonly referred to as dimensional catastrophes, hinder the efficient operation of the algorithms. To enhance the scalability of algorithms within the CTDE framework, researchers have proposed various methodologies, including value-based approaches [21,26,31] and actor-critic methods [32,36]. The value-based approach typically revolves around Q-values updated by agents. It often employs a mix network that combines the Q-values of each agent resulting in a joint value function Q_{tot} with global nature. This approach improves scalability to some extent. Nonetheless, in this approach, the agent needs to recalculate the value function and select the next action at each moment t, leading to a significant reduction in operational efficiency. In contrast, algorithms based on the actor-critic framework utilize policy gradients for updates. By incorporating a global state value function, these algorithms can achieve greater scalability. However, policy gradients are normally employed in tasks with continuous action spaces, which imposes certain limitations [2].

A novel method, $\underline{\text{M}}$ulti-agent $\underline{\text{R}}$einforcement learning with $\underline{\text{R}}$ectification $\underline{\text{C}}$apability in cooperative tasks (MRRC), is proposed in this study to address the challenges of sparse reward and scalability in multi-agent cooperative tasks. The MRRC method incorporates both individual and cooperative policies for each agent, with the individual policy aiming for individual rewards and the cooperative policy targeting cooperative rewards. By employing an actor network, the individual reward rectifies the policy and produces the action a, while the critic network generates Q-values based on this action. These Q-values are further adjusted by the mix network to form a global value function Q_{tot}. The loss value is calculated using Q_{tot}, and the MRRC's network is updated accordingly. MRRC incorporates a discrete action corrector to handle discrete actions and estimate the policy gradient. The key contributions of this study are as follows:

- A novel MARL method based on the actor-critic framework is proposed, specifically designed to tackle multi-agent cooperative tasks.
- An individual reward rectification module is introduced which equips each agent with two policies: a individual policy and a cooperative policy. By combining the benefits of rapid convergence from the individual policy and the global optimal solution from the cooperative policy, the actor network efficiently outputs the optimal next action, effectively addressing the challenges posed by sparse rewards.
- The mix network rectification module and the discrete action rectification module are introduced in this study, which significantly enhance the scalability of the algorithm. The linear combination of the value functions Q from all agents forms the global joint value function Q_{tot}, which proves to be effective, even in situations with changing environments.

2 Background

2.1 Dec-POMDP

In the pursuit of solving fully cooperative multi-agent tasks, it is often necessary to decompose the task and model it as a decentralized partially observable Markov decision process (Dec-POMDP) [17]. This modeling approach is normally represented by a one-tuple, denoted as $G = <S, A, P, r, \gamma, N, \Omega, O>$. Among the tuple $N \equiv \{1, 2, ..., n\}$ represents the set of all agents, S represents the set of environment states, and the observation $o_i \in \Omega$ is obtained from the observation kernel $O(s, i)$. The discount factor is denoted as $\gamma \in [0, 1)$. At time step t, each agent i selects an action $a_i \in A$ based on its observation o_i and the collective actions of all agents form a joint action denoted as a. This joint action influences the environment, leading to a change in the state and the generation of a reward value $r = R(s, \boldsymbol{a})$. Each agent then transitions to the next state s' through the state transition function $P(s'|s, \boldsymbol{a}) : S \times A \to [0, 1]$. Based on their respective observation values, each agent maintains its own action-observation history, denoted as $\tau_i \in T \equiv \{\Omega \times A\}$. The overarching objective throughout this process is to discover a joint policy that maximizes the joint action-value function $Q(s_t, \boldsymbol{a}_t) = \mathbb{E}[R_t|s_t, \boldsymbol{a}_t]$. In tasks where actions are continuous, agents utilize a continuous policy μ, while discrete actions correspond to a discrete policy π.

2.2 MERL and IRAT

MERL [14] and IRAT [30] are both approaches aimed at addressing the sparse reward problem in multi-agent tasks. They utilize gradient-based optimizers to train agents with higher reward potential, ultimately maximizing the overall reward.

MERL takes a hierarchical approach and employs two optimization processes to handle different objectives. Initially, an evolutionary algorithm is utilized to neuroevolve the entire population of agents, focusing on maximizing the collective reward in scenarios with sparse rewards. Subsequently, policy gradients are incorporated into the agent population for training. This involves information transfer between agents and the population, and culminates in the optimization of the overall policy using the acquired rewards.

In contrast, IRAT utilizes a team policy to guide the optimization of agent-specific policies. For a specific agent i, the optimization goal is to maximize its individual rewards, denoted as $J(\theta_i) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_i^t\right]$. Since the team policy in IRAT represents a joint policy, the actions outputted by each agent i are executed using the respective individual policies π_i. The objective for each agent i is to maximize the following function:

$$J(\theta_i) = \begin{cases} \mathbb{E}[\max(J^{CLIP}(\theta_i), J^{IRAT}(\theta_i))],\ \sigma_i^t \leq 1 \\ \mathbb{E}[\min(J^{CLIP}(\theta_i), J^{IRAT}(\theta_i))],\ \sigma_i^t > 1 \end{cases}, \tag{1}$$

where σ_i^t is the coefficient of similarity. $J^{IRAT}(\theta_i) = \mathbb{E}\left[\text{clip}\left(\sigma_i^t(\theta_i), 1-\xi, 1+\xi\right) A_i^t\right]$ is a cooperation-oriented goal and $J^{CLIP}(\theta_i) = \mathbb{E}[\min(\eta_i^t A_i^t, \text{clip}(\eta_i^t, 1-\epsilon, 1+\epsilon) A_i^t)]$ is a goal reward of a single agent.

2.3 MADDPG, QMIX and FACMAC

MADDPG [12], QMIX [21], and FACMAC [18] are all algorithms employed in the CTDE paradigm to address cooperative tasks involving multiple agents. QMIX is a value-based algorithm, while MADDPG and FACMAC are actor-critic framework-based algorithms. In MADDPG, each agent possesses its own independent actor and critic, enabling autonomous learning. For agent i, a joint action value function $Q_i^{\boldsymbol{\mu}}(s, a_1, ..., a_n; \phi_i)$ is maintained, and the critic network is updated by minimizing the following loss function:

$$LOSS(\phi_i) = \mathbb{E}_D[(y^i - Q_i^{\mu}(s, a_1, ..., a_n; \phi_i))^2], \tag{2}$$

where $y^i = r_i + \gamma Q_i^{\mu}(s', a_1', ..., a_n'; \phi_i^-)$,$a_i' = \mu_i(\tau_i'; \theta_i^-)$ and r_i represent the reward for agent i, $\{a_1', ..., a_n'\}$ denotes the set of actions taken by other agents obtained from the replay buffer D, and ϕ_i^- represents the parameter of the target critic for agent i. In contrast, the QMIX algorithm builds upon Q-learning and utilizes a linear, monotonic mix function to compose the global value function Q_{tot}, expressed as:

$$Q_{tot}(\boldsymbol{\tau}, \boldsymbol{a}, s; \boldsymbol{\phi}, \omega) = f_\omega\left(s, Q_1\left(\tau_1, a_1; \phi_1\right), \ldots, Q_n\left(\tau_n, a_n; \phi_n\right)\right), \tag{3}$$

where ω represents the parameter of the monotonic mixture function f.

Taking advantage of both MADDPG and QMIX, the FACMAC algorithm is introduced. It incorporates mix functions within the actor-critic framework. Specifically, the Q-values of each agent is combined with nonlinear functions to obtain the joint action value function $Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}, \boldsymbol{a}, s; \boldsymbol{\phi}, \omega) = g_\omega(s, \{Q_i^{\mu_i}(\tau_i, a_i; \phi_i)\}_{i=1}^n)$, parameters $\boldsymbol{\phi}$ and ϕ_i represent the parameters of the joint value function Q and $Q_i^{\mu_i}$ for agent i, respectively. The parameter of the nonlinear function is represented by ω. The critic network is updated by minimizing the following loss function:

$$LOSS(\boldsymbol{\phi}, \omega) = \mathbb{E}_D[(y^{tot} - Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}, \boldsymbol{a}, s; \boldsymbol{\phi}, \omega))^2], \tag{4}$$

where $y^{tot} = r + \gamma Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}', \boldsymbol{\mu}(\boldsymbol{\tau}'; \boldsymbol{\theta}^-), s'; \boldsymbol{\phi}^-, \omega^-)$, ω^-, $\boldsymbol{\theta}^-$, and $\boldsymbol{\phi}^-$ denote the parameters of the mix function, the target actor network, and the critic network, respectively.

3 Method

This section describes the components and processes of MRRC, and the overall framework of MRRC is shown in Fig. 1.

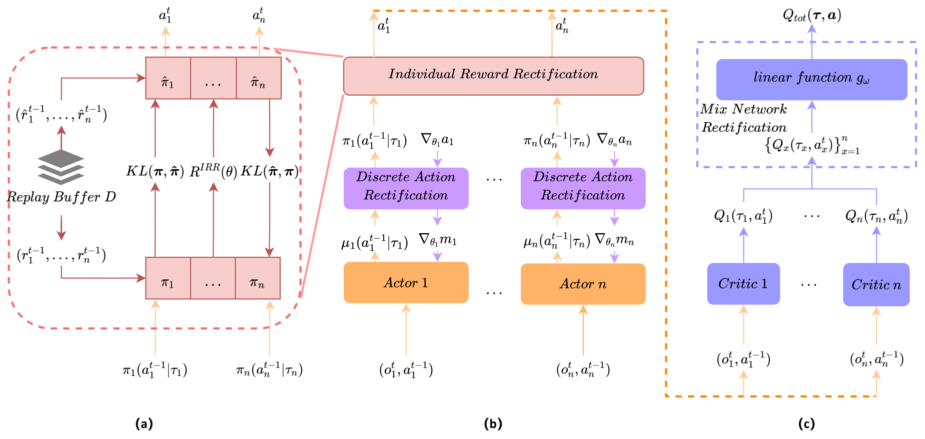

a_1^t
a_n^t
$(\hat{r}_1^{t-1}, \dots, \hat{r}_n^{t-1})$
$\hat{\pi}_1$ … $\hat{\pi}_n$
Replay Buffer D
$KL(\boldsymbol{\pi}, \hat{\boldsymbol{\pi}})$ $R^{IRR}(\theta)$ $KL(\hat{\boldsymbol{\pi}}, \boldsymbol{\pi})$
$(r_1^{t-1}, \dots, r_n^{t-1})$
π_1 … π_n
$\pi_1(a_1^{t-1}|\tau_1)$
$\pi_n(a_n^{t-1}|\tau_n)$
Individual Reward Rectification
$\nabla_{\theta_1} a_1$
$\nabla_{\theta_n} a_n$
Discrete Action Rectification
$\mu_1(a_1^{t-1}|\tau_1)$ $\nabla_{\theta_1} m_1$
$\mu_n(a_n^{t-1}|\tau_n)$ $\nabla_{\theta_n} m_n$
Actor 1
Actor n
(o_1^t, a_1^{t-1})
(o_n^t, a_n^{t-1})
$Q_{tot}(\boldsymbol{\tau}, \boldsymbol{a})$
linear function g_ω
Mix Network Rectification
$\{Q_x(\tau_x, a_x^t)\}_{x=1}^n$
$Q_1(\tau_1, a_1^t)$
$Q_n(\tau_n, a_n^t)$
Critic 1
Critic n
(a)
(b)
(c)

Fig. 1. Overview of the proposed method MRRC.**(a)** The overall process of individual reward rectification. **(b)** Sampling of continuous policy and policy gradients using discrete action rectification. **(c)** Computation of the joint value function Q_{tot} using mix network

3.1 Individual Reward Rectification

By leveraging the widespread implementation of the CTDE paradigm in MARL algorithms, researchers have achieved remarkable advancements in addressing collaborative multi-agent tasks, including QMIX [21] and MADDPG [12]. Nevertheless, such tasks often provide cooperative rewards exclusively upon accomplishing specific objectives. As the complexity of the environment escalates, agents encounter difficulties in attaining rewards within a designated timeframe, leading to sparse reward predicaments which yield sluggish policy convergence [33]. To tackle this issue, the integration of individual reward rectification into the actor-critic framework is proposed in this study. Individual reward rectification involves utilizing each agent's personal reward to rectify cooperative reward, thereby mitigating the sparsity inherent in the reward environment. Unlike IRAT, this approach necessitates the acquisition of two policies per agent—an individual policy and a cooperative policy—rendering it more compatible with the actor-critic framework and significantly enhancing the agent's learning efficiency. The exact process of Individual Reward Rectification is shown in Fig.2.

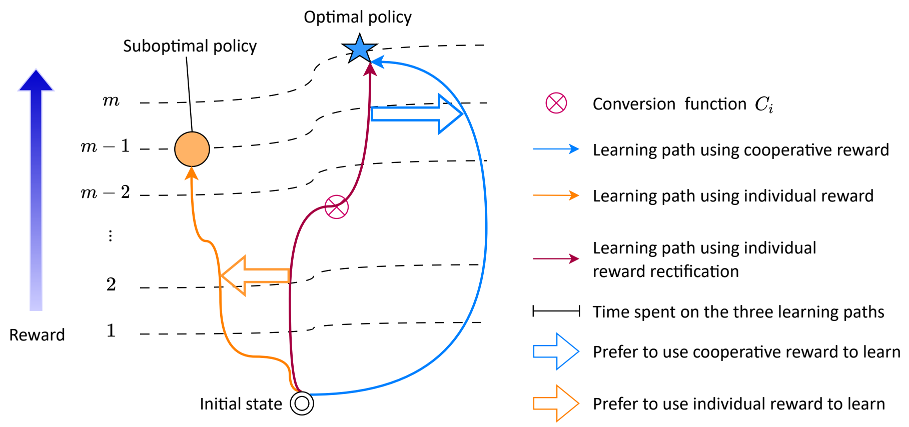

Optimal policy
Suboptimal policy
m
$m-1$
$m-2$
⋮
2
1
Reward
Initial state
Conversion function C_i
Learning path using cooperative reward
Learning path using individual reward
Learning path using individual reward rectification
Time spent on the three learning paths
Prefer to use cooperative reward to learn
Prefer to use individual reward to learn

Fig. 2. Example of individual reward rectification

In Fig.2, the far-right blue line represents the cooperative policy, which relies on sparse cooperative rewards for learning, resulting in a slow learning process. Conversely, the far-left orange line represents the individual policy, which utilizes dense individual rewards for learning, leading to a faster learning process but with a tendency to converge towards local optima. In order to leverage the strengths of both approaches while mitigating their drawbacks, we introduce individual reward rectification. This approach combines the benefits of dense individual rewards for rapid learning in the initial stage, and subsequently relies on the cooperative policy to attain the optimal reward position in the later stage.

Regarding the concept of "two policies per agent", specifically, agent i is required to acquire an individual policy $\hat{\pi}_i$ with parameter $\hat{\theta}_i$ and a cooperative policy π_i with parameter θ_i. Their objectives encompass maximizing the cumulative individual reward $\hat{R}(\hat{\theta}_i) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \hat{r}_i^t\right]$ and the cumulative cooperative reward $R(\theta_i) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_i^t\right]$, respectively. It is noteworthy that these two policies have mutually influencing effects. The process of individual reward rectification employs cumulative increasing KL regularization [27] and cumulative decreasing KL regularization to strike a balance between the two policies, ultimately leading to the optimal action produced by the individual policy. The cumulative decreasing KL regularization is employed to learn the actions performed by the individual policy, while the cumulative increasing KL regularization refines and samples the actions of the cooperative policy.

The cumulative increasing KL regularization does not exert its influence during the initial stages of learning for agent i. Consequently, the actions acquired by agent i tend to learn individual policies. This accelerates the learning efficiency and assists in escaping the sparse reward state in the early stages of learning. Subsequently, the role of the regularizer employed to learn cooperative policies gains prominence in the later stages of learning. Agent i becomes inclined to acquire cooperative policies associated with substantial reward values, thus eliminating the drawback of suboptimal solutions yielded by solely learning individual policies.

Based on the aforementioned analysis, the current challenge focuses on determining the optimal balance between the influence of the individual and the cooperative policies, and the difference between the two should not be sufficient to generate conflicts between the policies. Similar to IRAT, a similarity coefficient is utilized to ascertain the equilibrium point, denoted as δ. The similarity coefficient is defined as follows:

$$\delta_i^t(\theta_i, \hat{\theta}_i) = \frac{\hat{\pi}(a_i^t|\tau_i^t, \hat{\theta}_i)}{\pi(a_i^t|\tau_i^t, \theta_i)}, \tag{5}$$

where δ_i^t represents the similarity coefficient of agent i at time t and $\delta_i^t \in [1-x, 1+x]$, therein x serving as the limiting factor. Examining the expression of the similarity coefficient reveals a distinction from IRAT. In this research, each agent corresponds to an individual policy $\hat{\pi}$ and a cooperative policy π, which facilitates the computation of the similarity coefficient, enhances operational efficiency, and adapts more effectively to the actor-critic framework within the CTDE paradigm.

In the initial execution of individual reward rectification, significant emphasis is placed on the individual reward, necessitating the utilization of a decreasing KL regularizer to amplify the effectiveness of the individual policy. The target reward for the individual policy $\hat{R}(\hat{\theta}_i)$ is represented as follows:

$$\begin{aligned} \hat{R}(\hat{\theta}_i) = \mathbb{E}[&\mathbb{I}_{\delta \geq 1} \max(\delta_i^t(\theta_i, \hat{\theta}_i), \hat{\delta}_i^t(\hat{\theta}_i, \theta_i))\hat{\boldsymbol{A}}_i \\ &+ \mathbb{I}_{\delta < 1} \min(\delta_i^t(\theta_i, \hat{\theta}_i), \hat{\delta}_i^t(\hat{\theta}_i, \theta_i))\hat{\boldsymbol{A}}_i \\ &+ \alpha KL(\pi_i, \hat{\pi}_i)], \end{aligned} \tag{6}$$

where $\mathbb{I}$ denotes the conditional selection function, α indicates the decreasing coefficient, and $\hat{\boldsymbol{A}}_i$ denotes the set of actions taken by agent i using the individual policy. Subsequently, a progressively increasing KL regularizer is introduced to enhance its influence when a cooperative policy is required, and the target reward for the cooperative policy $R(\theta_i)$ is expressed as:

$$\begin{aligned} R(\theta_i) = \mathbb{E}[&\mathbb{I}_{\delta \geq 1} \min(\delta_i^t(\theta_i, \hat{\theta}_i), \hat{\delta}_i^t(\hat{\theta}_i, \theta_i))\boldsymbol{A}_i \\ &+ \mathbb{I}_{\delta < 1} \max(\delta_i^t(\theta_i, \hat{\theta}_i), \hat{\delta}_i^t(\hat{\theta}_i, \theta_i))\boldsymbol{A}_i \\ &+ \beta KL(\hat{\pi}_i, \pi_i)], \end{aligned} \tag{7}$$

where β represents the incremental coefficient, and $\boldsymbol{A}_i$ denotes the set of actions taken by agent i using the cooperative policy. Following the computation of the individual and cooperative rewards, which necessarily translate into the target reward for individual reward rectification $R^{IRR}(\theta)$, which can be described as:

$$R^{IRR}(\theta_i, \hat{\theta}_i) = \begin{cases} C_i^t \hat{R}(\hat{\theta}_i) - (C_i^t - C_i^{t-1})R(\theta_i), & C_i^t \geq 0 \\ -C_i^t R(\theta_i) + (C_i^{t-1} - C_i^t)\hat{R}(\hat{\theta}_i), & C_i^t < 0 \end{cases}, \tag{8}$$

where C_i^t denotes the conversion function and $C_i^t = r_i^t + \gamma V(s_t) - V(s_{t-1})$, $V(s_t) = \mathbb{E}_{s_{t+1}:\infty, a_t:\infty}[\sum_{l=0}^{\infty} r_{t+l}]$ [22]. When $C_i^t \geq 0$, the individual reward holds dominates, whereas when $C_i^t < 0$, the cooperative reward takes precedence. The output action set $\boldsymbol{A}^{IRR}$ is calculated from the rectified individual policy and $\boldsymbol{A}^{IRR} = \{a_1, a_2, ..., a_n\}$.

3.2 Mix Network Rectification

As a prominent representative of the CTDE paradigm, MADDPG excels at swiftly attaining the global optimal solution through centralized critic training. However, as the number of agents increases, obtaining a single global state-value function through centralized training for all agents becomes challenging [24]. To address this issue, mix network rectification is proposed, wherein a mix network [6] is employed to factorize the Q-value of each agent and derive the global value, denoted as Q_{tot}. Distinct from FACMAC, the factorization function employed is linear, which promotes scalability.

Specifically, in mix network rectification, the critic undergoes centralized training using the value functions of all agents to acquire Q_{tot}, as depicted by:

$$\begin{aligned} Q_{tot}^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \boldsymbol{a}, s; \boldsymbol{\phi}, \omega) =& g_\omega(s, Q_1^{\pi_1}(\tau_1, a_1; \phi_1), \\ & ..., Q_n^{\pi_n}(\tau_n, a_n; \phi_n)), \end{aligned} \tag{9}$$

where Q_{tot}, the joint action value function, depends on the parameters $\boldsymbol{\phi}$. On the other hand, the value function for a single agent, Q_n, depends on ϕ_n. The mix network parameter is denoted as ψ, and the mixing function, g_ψ, is a linear function. During the training process, the critic network is updated by minimizing the loss function, which is given by:

$$LOSS(\boldsymbol{\phi}, \omega) = \mathbb{E}_D[(y^{tot} - Q_{tot}^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \boldsymbol{a}, s; \boldsymbol{\phi}, \omega))^2], \tag{10}$$

where $y^{tot} = r + \gamma Q_{tot}^{\boldsymbol{\pi}}(\boldsymbol{\tau}', \boldsymbol{\pi}(\boldsymbol{a}, \boldsymbol{\tau}'; \boldsymbol{\psi}^-), s'; \boldsymbol{\phi}^-, \omega^-)$, and D represents the replay buffer. The parameters $\boldsymbol{\psi}^-$, $\boldsymbol{\phi}^-$, and ω^- represent the target actor, target critic, and target mix network, respectively.

Once the global state value function Q_{tot} is computed, the actor network requires updating. Taking MADDPG as an example, each agent's actor parameters are updated through its own policy gradient, where the policy gradient of each agent is affected by the actions of all agents, typically obtained from the replay buffer D. Nevertheless, this approach typically leads to reduced computational efficiency. To overcome this challenge, a mix network is adopted to introduce a novel policy gradient calculator. This innovation effectively accelerates the efficiency of global policy updates for actor networks, thereby enhancing the algorithm's scalability.

Similar to FACMAC, during actor network updates, the action set is denoted as $\boldsymbol{A} = \{a_1, a_2, ..., a_n\}$, is extracted from the current policies of all agents rather than solely relying on the replay buffer D. However, unlike FACMAC, the mix network in this study utilizes linearity, which facilitates the implementation of policy gradient estimation and reduces computational complexity. The actor network's policy gradient can be expressed as follows:

$$\nabla_{\psi} J(\boldsymbol{\pi}) = \mathbb{E}_D[\nabla_{\psi}\boldsymbol{\pi}\nabla_{\boldsymbol{\pi}} Q_{tot}^{\boldsymbol{\pi}}(\boldsymbol{\tau}, \pi_1(\tau_1, a_1), ..., \pi_n(\tau_n, a_n), s)], \tag{11}$$

where $\boldsymbol{\pi} = \{\pi_1(\tau_1, a_1|\psi_1), ..., \pi_n(\tau_n, a_n|\psi_n)\}$ denotes the set of current policies for all agents, and ψ indicates the actor network's parameter.

3.3 Discrete Action Rectification

Conventionally, policy gradients are computed based on continuous policies, denoted as μ. However, many cooperative tasks inherently involve discrete actions. This poses a challenge for algorithms like MADDPG, as they struggle to handle tasks with discrete actions effectively. The discrete action rectification is introduced to settle this issue. Specifically, the Gumbel-Softmax [9] technique is utilized to sample from the continuous policy, resulting in the output of the sampled agent being $\pi(\boldsymbol{a}^{t-1}, \boldsymbol{\tau})$. Moreover, Gumbel-Softmax enables the computation of gradients for discrete samples, thereby approximating the policy gradient. The process of gradient approximation is described as follows:

$$\begin{aligned} \nabla_{\psi} J(\boldsymbol{\mu}) &= \mathbb{E}_D[\nabla_{\psi}\boldsymbol{\mu}\nabla_{\boldsymbol{\mu}} Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}, \mu_1(\tau_1), ..., \mu_n(\tau_n), s)] \\ &\approx \mathbb{E}_D[\nabla_{\psi}\boldsymbol{v}\nabla_{\boldsymbol{v}} Q_{tot}^{\boldsymbol{v}}(\boldsymbol{\tau}, v_1, ..., v_n, s)] = \nabla_{\psi} J(\boldsymbol{\pi}), \end{aligned} \tag{12}$$

where $\boldsymbol{v} = \boldsymbol{\pi}(\boldsymbol{a}|\boldsymbol{\tau})$, and $\boldsymbol{v} = \{v_1, ..., v_n\}$ demonstrates the actions performed during the continuous sampling process.

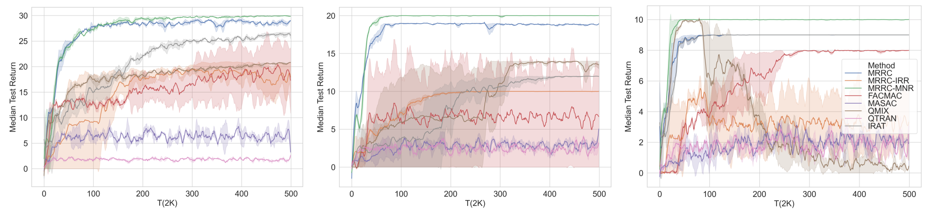
4 Experiments and Results

This section presents the experimental results of the MRRC algorithm conducted in two cooperative environments: "Predator-Prey" and "StarCraft Multi-Agent Challenge (SMAC)". To assess the effectiveness of MRRC, it was compared against several state-of-the-art actor-critic algorithms, namely FACMAC [18] and MASAC [8], as well as the value-based algorithms QMIX [21] and QTRAN

[26], and the policy-based algorithm IRAT [30]. Additionally, ablation experiments were performed to evaluate the impact of the "Individual Reward Rectification" and "Mix Network Rectification" modules in MRRC. Specifically, two new algorithms were created in this study, "MRRC-IRR" by removing individual reward rectification, and "MRRC-MNR" by excluding mix network rectification. In the "Predator-Prey" environment, all experiments were conducted with a training duration of 1 million steps, while for the more challenging "SMAC" environment, the training duration was set to 2 million steps. Each experiment was executed with seven random seeds, and the 95% confidence interval is shown shaded.

4.1 Predator-Prey

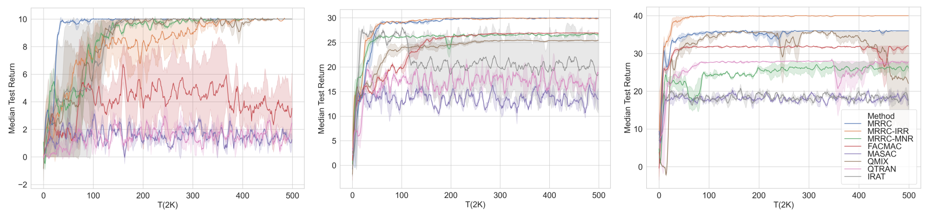
Experiments were conducted in the "Predator-Prey" environment, utilizing a 12×12 grid world. The preys were controlled by built-in AI, and the objective of the MARL algorithm was to manipulate the predators to capture these preys. The observation range of the predators was limited to a 2×2 grid. The aim was to verify the effectiveness of the MRRC algorithm in environments with sparse rewards and scalability when the number of agents increases. Two experimental scenarios were established for this purpose: Predator-Prey with decreasing preys and Predator-Prey with increasing agents. Three experiments were conducted in the Predator-Prey with decreasing preys. The number of predators was fixed at 6, while the number of preys varied, specifically 3, 2, and 1. The experimental results are presented in Fig.3. Similarly, three experiments were conducted in the Predator-Prey with increasing agents. The number of predators was set to 3, 6, and 9, respectively. The corresponding results are illustrated in Fig.4.



(a) 6 predators and 3 preys **(b)** 6 predators and 2 preys **(c)** 6 predators and 1 prey

Fig. 3. Median test returns of Predator-Prey with decreasing preys

As depicted in Fig.3, the MRRC algorithm exhibits superior performance compared to all other baseline algorithms in Predator-Prey with decreasing preys. When the number of preys is high, as illustrated in Fig.3(a), most algorithms manage to maintain relatively better performance. However, as the number of preys decreases, resulting in increasingly sparse rewards in the environment, only MRRC, MRRC-MNR, and IRAT demonstrate significant per-



(a) 3 predators and 3 preys **(b)** 6 predators and 6 preys **(c)** 9 predators and 9 preys

Fig. 4. Median test returns of Predator-Prey with increasing agents

formance, particularly when only one prey remaining, as shown in Fig.3(c). Notably, the reward value of QMIX algorithm decline over time, indicating its poor performance in an environment with extremely sparse rewards. Among all the ablation experiments conducted in the sparse reward scenario, MRRC-MNR achieves the best performance, suggesting that individual reward rectification effectively addresses the challenges posed by sparse rewards in the environment. Remarkably, the performance of both MRRC and MRRC-IRR is lower than that of MRRC-MNR, suggesting that mix network rectification may introduce additional complexity to the algorithm, which potentially leads to negative effects.

Fig.4 demonstrates that MRRC surpasses all other baseline algorithms in the Predator-Prey with increasing agents. When the number of agents is small, as illustrated in Fig.4(a)and4(b), most algorithms exhibit better performance. Nevertheless, as the number of agents increases, as shown in Fig.4(c), the performance of the baseline algorithms QMIX and FACMAC declines significantly, highlighting their poor scalability. In contrast, the MRRC algorithm which incorporates the monotonic mix function performs better, emphasizing the crucial role of mix network rectification in enhancing scalability. The performance of MRRC is noticeably lower than that of MRRC-IRR in Fig.4(c), indicating that individual reward rectification has a negative impact on scalability.

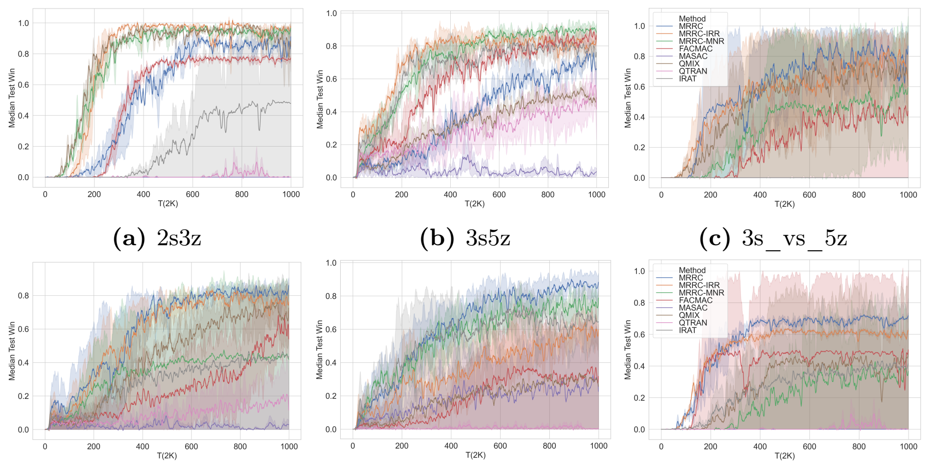
4.2 More Challenging SMAC

To assess the adaptability of the MRRC algorithm in complex tasks, six distinct experiments were conducted in the more challenging SMAC environment. The details of the experimental maps and parameters can be found in Table 1, and accroding to the research of Wang et al. [29] the other parameters were set. The results of these experiments were illustrated in Fig.5.

As presented in Fig.5, the MRRC algorithm demonstrates strong performance across SMAC environments with varying maps and difficulty levels. In simpler tasks, as shown in Fig.5(a)and5(b), MRRC, MRRC-IRR, MRRC-MNR, FACMAC, and QMIX all exhibit excellent performance. As task complexity increases, as indicated in Fig.5(c),5(d),5(e)and5(f), the MRRC algorithm consistently outperforms the other baseline algorithms, demonstrating its superiority and adaptability in diverse cooperative environments. Notably, the QTRAN and

Table 1. The maps and parameters of the SMAC in experiment

Map name	Difficulty	Ally	Opponent
2s3z	Easy	2 Stalkers & 3 Zealots	–
3s5z	Easy	3 Stalkers & 5 Zealots	–
3s_vs_5z	Hard	3 Stalkers	5 Zealots
5m_vs_6m	Hard	5 Marines	6 Marines
3s5z_vs_3s6z	Super hard	3 Stalkers & 5 Zealots	3 Stalkers & 6 Zealots
27m_vs_30m	Super hard	27 Marines	30 Marines



(a) 2s3z **(b)** 3s5z **(c)** 3s_vs_5z

(d) 5m_vs_6m **(e)** 3s5z_vs_3s6z **(f)** 27m_vs_30m

Fig. 5. Median test win for easy(a)(b), hard(c)(d) and super-hard (e)(f) maps of SMAC

MASAC algorithms consistently underperform across all tasks, indicating their limited adaptability in different environments.

The results were analyzed in detail in the ablation experiments. In simpler tasks, as shown in Fig.5(a)and5(b), both the MRRC-IRR and MRRC-MNR algorithms outperform the MRRC algorithm. This suggests that in simple tasks, excessive optimization may be unnecessary, as solving such tasks does not require the full range of modules. Alternatively, incorporating an excessive number of modules increases computational burden and degrades algorithm performance. As task complexity increases, the MRRC algorithm outperforms the two module-removed algorithms, indicating the significance of both individual reward rectification and mix network rectification in challenging environments. Fig.5(e) shows that MRRC-MNR outperforms MRRC-IRR in the "3s5z_vs_3s6z" task, which features sparse rewards. This highlights the effectiveness of individual reward rectification in addressing the sparse reward problem. Notably, in an environment with a large number of agents, as depicted in Fig.5(f), MRRC-IRR outperforms MRRC-MNR, indicating that mix network rectification excels at enhancing algorithm scalability.

5 Conclusion

In this study, we propose a novel actor-critic based method called MRRC for addressing multi-agent cooperation tasks. MRRC leverages the rapid convergence of the individual policy to rectify agent behaviors and effectively tackle the issue of sparse rewards. To enhance the scalability of the algorithm, a monotonic mix network is introduced to rectify the agents' value functions and constructs global state value functions. Additionally, Gumbel-Softmax is combined to handle discrete actions. Through extensive experiments conducted in the "Predator-Prey" and "StarCraft Multi-Agent Challenge (SMAC)" environments, we demonstrate that MRRC effectively guides the actions of agents, enabling them to overcome the slow convergence caused by sparse rewards and improving the scalability of the algorithm. Moreover, the results indicate that a superior performance of MRRC is achieved compared to other baseline algorithms. In future work, we aim to further optimize the algorithm to mitigate the negative impacts of individual reward rectification and mix network rectification on the overall algorithm's performance.

Acknowledgements This work is sponsored by Equipment Advance Research Fund (NO.61406190118).

References

1. Aleardi, M., Vinciguerra, A., Stucchi, E., Hojat, A.: Machine learning-accelerated gradient-based markov chain monte carlo inversion applied to electrical resistivity tomography. Near Surface Geophysics **20**(4), 440–461 (2022)
2. Barth-Maron, G., Hoffman, M.W., Budden, D., Dabney, W., Horgan, D., Tb, D., Muldal, A., Heess, N., Lillicrap, T.: Distributed distributional deterministic policy gradients. arXiv preprint arXiv:1804.08617 (2018)
3. Castiglioni, I., Rundo, L., Codari, M., Di Leo, G., Salvatore, C., Interlenghi, M., Gallivanone, F., Cozzi, A., D'Amico, N.C., Sardanelli, F.: Ai applications to medical images: From machine learning to deep learning. Physica Medica **83**, 9–24 (2021)
4. Cetinic, E., She, J.: Understanding and creating art with ai: review and outlook. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM) **18**(2), 1–22 (2022)
5. Dalal, G., Hallak, A., Dalton, S., Mannor, S., Chechik, G., et al.: Improve agents without retraining: Parallel tree search with off-policy correction. Advances in Neural Information Processing Systems **34**, 5518–5530 (2021)
6. Ha, D., Dai, A., Le, Q.V.: Hypernetworks. arXiv preprint arXiv:1609.09106 (2016)
7. Huang, S., Abdolmaleki, A., Vezzani, G., Brakel, P., Mankowitz, D.J., Neunert, M., Bohez, S., Tassa, Y., Heess, N., Riedmiller, M., et al.: A constrained multi-objective reinforcement learning framework. In: Conference on Robot Learning. pp. 883–893. PMLR (2022)

8. Iqbal, S., Sha, F.: Actor-attention-critic for multi-agent reinforcement learning. In: International conference on machine learning. pp. 2961–2970. PMLR (2019)
9. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)
10. Jin, L., Qian, S., Owens, A., Fouhey, D.F.: Planar surface reconstruction from sparse views. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12991–13000 (2021)
11. Kraemer, L., Banerjee, B.: Multi-agent reinforcement learning as a rehearsal for decentralized planning. Neurocomputing **190**, 82–94 (2016)
12. Lowe, R., Wu, Y.I., Tamar, A., Harb, J., Pieter Abbeel, O., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. Advances in neural information processing systems **30** (2017)
13. Lu, Y., Li, W.: Techniques and paradigms in modern game ai systems. Algorithms **15**(8), 282 (2022)
14. Majumdar, S., Khadka, S., Miret, S., McAleer, S., Tumer, K.: Evolutionary reinforcement learning for sample-efficient multiagent coordination. In: International Conference on Machine Learning. pp. 6651–6660. PMLR (2020)
15. Mansour, R.F., El Amraoui, A., Nouaouri, I., Díaz, V.G., Gupta, D., Kumar, S.: Artificial intelligence and internet of things enabled disease diagnosis model for smart healthcare systems. IEEE Access **9**, 45137–45146 (2021)
16. Nian, R., Liu, J., Huang, B.: A review on reinforcement learning: Introduction and applications in industrial process control. Computers & Chemical Engineering **139**, 106886 (2020)
17. Oliehoek, F.A., Amato, C.: A concise introduction to decentralized pomdps (2015)
18. Peng, B., Rashid, T., Schroeder de Witt, C., Kamienny, P.A., Torr, P., Böhmer, W., Whiteson, S.: Facmac: Factored multi-agent centralised policy gradients. Advances in Neural Information Processing Systems **34**, 12208–12221 (2021)
19. Qin, Z., Zhang, K., Chen, Y., Chen, J., Fan, C.: Learning safe multi-agent control with decentralized neural barrier certificates. arXiv preprint arXiv:2101.05436 (2021)
20. Rajeswar, S., Ibrahim, C., Surya, N., Golemo, F., Vazquez, D., Courville, A., Pinheiro, P.O.: Haptics-based curiosity for sparse-reward tasks. In: Conference on Robot Learning. pp. 395–405. PMLR (2022)
21. Rashid, T., Samvelyan, M., De Witt, C.S., Farquhar, G., Foerster, J., Whiteson, S.: Monotonic value function factorisation for deep multi-agent reinforcement learning. The Journal of Machine Learning Research **21**(1), 7234–7284 (2020)
22. Schulman, J., Moritz, P., Levine, S., Jordan, M., Abbeel, P.: High-dimensional continuous control using generalized advantage estimation. arXiv preprint arXiv:1506.02438 (2015)
23. Shao, Y., Lin, J.C.W., Srivastava, G., Guo, D., Zhang, H., Yi, H., Jolfaei, A.: Multi-objective neural evolutionary algorithm for combinatorial optimization problems. IEEE transactions on neural networks and learning systems (2021)
24. Sharma, P.K., Fernandez, R., Zaroukian, E., Dorothy, M., Basak, A., Asher, D.E.: Survey of recent multi-agent reinforcement learning algorithms utilizing centralized training. In: Artificial intelligence and machine learning for multi-domain operations applications III. vol. 11746, pp. 665–676. SPIE (2021)
25. Shen, Y., Song, K., Tan, X., Li, D., Lu, W., Zhuang, Y.: Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface. arXiv preprint arXiv:2303.17580 (2023)

26. Son, K., Kim, D., Kang, W.J., Hostallero, D.E., 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)
27. Vieillard, N., Kozuno, T., Scherrer, B., Pietquin, O., Munos, R., Geist, M.: Leverage the average: an analysis of kl regularization in reinforcement learning. Advances in Neural Information Processing Systems **33**, 12163–12174 (2020)
28. Wang, J., Ren, Z., Liu, T., Yu, Y., Zhang, C.: Qplex: Duplex dueling multi-agent q-learning. arXiv preprint arXiv:2008.01062 (2020)
29. Wang, J., Zhang, Y., Gu, Y., Kim, T.K.: Shaq: Incorporating shapley value theory into multi-agent q-learning. Advances in Neural Information Processing Systems **35**, 5941–5954 (2022)
30. Wang, L., Zhang, Y., Hu, Y., Wang, W., Zhang, C., Gao, Y., Hao, J., Lv, T., Fan, C.: Individual reward assisted multi-agent reinforcement learning. In: International Conference on Machine Learning. pp. 23417–23432. PMLR (2022)
31. Wang, T., Wang, J., Zheng, C., Zhang, C.: Learning nearly decomposable value functions via communication minimization. arXiv preprint arXiv:1910.05366 (2019)
32. Wang, Y., Han, B., Wang, T., Dong, H., Zhang, C.: Off-policy multi-agent decomposed policy gradients. arXiv preprint arXiv:2007.12322 (2020)
33. Yan, Y., Chow, A.H., Ho, C.P., Kuo, Y.H., Wu, Q., Ying, C.: Reinforcement learning for logistics and supply chain management: Methodologies, state of the art, and future opportunities. Transportation Research Part E: Logistics and Transportation Review **162**, 102712 (2022)
34. Zhang, R., McNeese, N.J., Freeman, G., Musick, G.: " an ideal human" expectations of ai teammates in human-ai teaming. Proceedings of the ACM on Human-Computer Interaction **4**(CSCW3), 1–25 (2021)
35. Zhang, T., Li, Y., Wang, C., Xie, G., Lu, Z.: Fop: Factorizing optimal joint policy of maximum-entropy multi-agent reinforcement learning. In: International Conference on Machine Learning. pp. 12491–12500. PMLR (2021)
36. Zhou, H., Lan, T., Aggarwal, V.: Pac: Assisted value factorisation with counterfactual predictions in multi-agent reinforcement learning. arXiv preprint arXiv:2206.11420 (2022)