ECAI 2023
K. Gal et al. (Eds.)

doi:10.3233/FAIA230586

IPERS: Individual Prioritized Experience Replay with Subgoals for Sparse Reward Multi-Agent Reinforcement Learning

Zaipeng Xie[a,b;*], **Yufeng Zhang**[b], **Chentai Qiao**[b] **and Sitong Shen**[b]

[a]Key Laboratory of Water Big Data Technology of Ministry of Water Resources, Hohai University, Nanjing, China
[b]College of Computer and Information, Hohai University, Nanjing, China

Abstract. Multi-agent reinforcement learning commonly uses a global team reward signal to represent overall collaborative performance. Value decomposition breaks this global reward into estimated individual value functions per agent, enabling efficient training. However, in sparse reward environments, agents struggle to assess if their actions achieve the team goal, slowing convergence. This impedes the algorithm's convergence rate and overall efficacy. We present IPERS, an Individual Prioritized Experience Replay algorithm with Subgoals for Sparse Reward Multi-Agent Reinforcement Learning. IPERS integrates joint action decomposition and prioritized experience replay, maintaining invariance between global and individual loss gradients. Subgoals serve as intermediate goals that break down complex tasks into simpler steps with dense feedback and provide helpful intrinsic rewards that guide agents. This facilitates learning coordinated policies in challenging collaborative environments with sparse rewards. Experimental evaluations of IPERS in both the SMAC and GRF environments demonstrate rapid adaptation to diverse multi-agent tasks and significant improvements in win rate and convergence performance relative to state-of-the-art algorithms.

1 Introduction

Multi-agent reinforcement learning (MARL) constitutes a transformative development in artificial intelligence, involving multiple autonomous agents who cooperate and learn within a shared environment to fulfill a common goal while simultaneously receiving collective rewards. It has captivated the interest of myriad researchers [8]. However, integrating reinforcement learning principles into multi-agent systems (MAS) presents several challenges. In a cooperative MAS [20], agents collaborate to realize a common goal, emphasizing group rewards over individual ones. Individual rewards gauge an agent's solitary contribution to the environment, while group rewards assess the entire team's success in accomplishing the given task. In MARL, centralized training with decentralized execution (CTDE) [19] has been the most prevalent paradigm. However, due to the lack of a central controller during the execution phase, an agent cannot ascertain its contribution to global task completion. During training, agents may struggle to learn and acquire optimal strategies effectively when reward signals are deferred until the event's conclusion and sparse rewards are granted upon task completion.

[*] Corresponding Author. Email: zaipengxie@hhu.edu.cn.

Reward sparsity is often a pervasive challenge [18] in multi-agent systems. Consequently, investigating techniques to ensure the stability or enhancement of algorithmic performance in such environments becomes essential for addressing real-world tasks [4, 33, 30]. The trial-and-error reward model in reinforcement learning typically involves substantial overhead costs during strategy revisions. As a result, optimizing learning in practical MARL implementations relies on augmenting the training convergence, bolstering experience sampling efficiency, and effectively capitalizing on experience. Intrinsic rewards [26, 5, 32, 14] based on goal-oriented value functions have been used to address sparse rewards in reinforcement learning. These intrinsic rewards can be utilized to train value networks by sampling experiences from a replay buffer. Furthermore, in multi-agent reinforcement learning, the global team reward represents the overall performance of the entire team of agents. Individual rewards for each agent can be estimated from a shared team reward signal [12]. However, there is a trade-off between optimizing the global team reward and optimizing individual rewards, representing individual agents' performance. In addition to intrinsic rewards, past experiences may hold significant value. Agents can leverage their past experiences to enhance the state transition of the current model. Even in environments where rewards are sparse, agents can acquire effective behavioral strategies. However, not all prior experiences hold equal significance for agents [25, 21]. Thus, it is beneficial to develop non-uniform sampling methods in lieu of uniform sampling.

This study proposes a novel multi-agent reinforcement learning approach, named Individual Prioritized Experience Replay with Subgoals for Sparse Reward Multi-Agent Reinforcement Learning (IPERS), to address the problem of diminished training efficiency in multi-agent systems due to sparse rewards. IPERS utilizes prioritized experience replay to select valuable individual agent transitions from shared experiences. It identifies subgoals for each agent by considering both local and global Q-values to guide exploration. Intrinsic rewards are incorporated alongside sparse environmental rewards to mitigate the impact of sparse external signals. Our contributions can be summarized as follows:

- We propose decoupling the joint observation and action data into individual transitions for each agent, allowing for the isolation of their respective attributes. Furthermore, we identify state transitions characterized by substantial TD-errors using prioritized experience replay to select the most useful experiences for training each agent's policy. This helps to speed up the learning process and concurrently enhance the convergence.

- We introduce decentralized training on individual experiences that employs intrinsic rewards based on a learned goal-conditioned value function. This facilitates off-policy learning tailored to each agent using their own experiences. Subgoals are utilized to shape the intrinsic rewards for individual experiences that can promote exploration and accelerate learning.
- We evaluate the proposed IPERS algorithm on different MAS tasks, including both dense and sparse reward settings, using the SMAC and Google research football environments. Sparse-reward settings are particularly challenging, requiring agents to coordinate actions for extended periods before receiving any rewards. Experimental results show that the IPERS algorithm consistently outperforms four state-of-the-art methods in terms of convergence speed and success rate across all scenarios.

2 Related work

Multi-agent reinforcement learning is a dynamic field dedicated to developing agents capable of interacting and collaborating in complex environments. Value decomposition methods [29] play a crucial role in this pursuit by breaking down the global value function into individual value functions tailored to each agent. Sunehag et al. [29] propose the Value Decomposition Network (VDN) algorithm that decomposes the joint action value function into the sum of each agent's action value function. However, the VDN algorithm assumes that all agents contribute equally to the global value, making the linear decomposition method impractical for complex tasks. To overcome this limitation, Rashid et al. [24] proposed QMIX, extending to nonlinear integrals and allowing varied agent contributions, despite potential limitations in fitting the value function accurately. Notable progress has been made with other algorithms like QTRAN [27] and Qatten [34]. However, value decomposition algorithms may struggle with sparse rewards and delayed reward signals, resulting in credit assignment problems and action-reward association difficulties. Therefore, it is crucial to exercise caution in using value decomposition algorithms to ensure optimal decision-making.

The adoption of experience replay [35] in reinforcement learning can offer a solution to the issues of convergence speed and poor performance that stem from the absence of reward signals. Experience replay involves reusing past transformation data to update the present strategy. It aims to enhance data utilization, accelerate learning efficiency, and avert long-term exploration by agents in the environment, all of which can significantly compromise algorithmic performance. Uniform sampling was the preferred sampling method in early experience replay, but it fails to account for the variations in each transformation. Intuitively, transitions with higher learning values should be sampled with greater frequency. Several replay strategies have been developed in single-agent reinforcement learning [28, 13, 25], yet prioritized experience replay (PER) [25] remains the most successful approach. PER uses TD-error to measure state transitions' significance, calculates priorities, and replays more critical state transitions at higher frequencies. However, in MAS, agents receive joint rewards from the environment, and it can be difficult to discern their contributions to these joint rewards. Overcoming the credit assignment problem and comprehending individual agents' contributions to joint rewards are crucial. Distributed prioritized experience replay [10] extends PER to allow decentralized agents to independently collect and prioritize transitions based on TD error for focused sampling. Importance sampling weights, which correct bias in mini-batch training, are used in this process. However, this approach requires increased communication for global experience sharing and may lack flexibility for global optimization. Remember and forget for experience replay [22] utilizes the PER method, tailored explicitly for multi-agent RL algorithms. Nevertheless, the results suggest that these approaches could be suboptimal, indicating that further improvements can be useful for experience replay in multi-agent reinforcement learning.

On the other hand, the efficacy of an agent's exploration can be significantly enhanced by allocating suitable subgoals [31, 17, 3, 2, 14]. Wang et al. [31] adopted a hierarchical approach, classifying the learning strategies of agents into high-level and low-level types. The former assigns pertinent subgoals to the agent, while the latter guides the agent in executing goal-related actions autonomously. Chen et al. [3] view the allocation of subgoals as a task allocation problem with limited resources in linear programming. The resulting solution assigns agents to subgoals, after which a universal multi-agent reinforcement learning algorithm completes the subgoals. Therefore, appropriate and effective subgoal allocation can significantly improve algorithm performance and accelerate convergence. Jeon et al. [14] address sparse reward by utilizing intrinsic rewards based on a learned goal-conditioned value function to train individual value networks by sampling experiences from a shared replay buffer. Liang et al. [17] introduced a two-layer decision-making scheme, where the high-level strategy solves the problem of goal allocation, and the low-level strategy facilitates agent collaboration in the pursuit of completing tasks. Specifically, agents are assigned to modules with akin subgoals, optimizing subgoal completion through collaboration.

To improve convergence performance in the absence of reward signals, the prioritized experience replay method with subgoals can be employed. Notably, prioritized experience replay and subgoals have shown promise [13, 25, 1, 31, 3] in enhancing the performance of reinforcement learning algorithms, and further investigation is needed to apply these methods in multi-agent settings effectively.

3 Background

Decentralized Partially Observable Markov Decision Process: A cooperative multi-agent task with N agents can be represented as a decentralized partially observable Markov decision process (Dec-POMDP) [23], wherein each agent individually selects actions based on its local observations. We describe the Dec-POMDP using tuple $G = \langle S, A, P, R, \Omega, O, \gamma, N \rangle$. Here, S is the state space, A is the action space, P is the transition probability function, R is the reward function, Ω is the set of observations made by agents, O is the conditional observation probability, γ is the discount factor, and N is the number of agents. At each time step, each agent $i \in [1, N]$ selects an action $a^i \in A$. This choice leads to an environmental transition probability $P(s'|s, a)$, representing the likelihood of transitioning from the current global state s to the next global state s'. Due to partial observability, agent i makes an observation $o^i \in \Omega$ following the conditional observation probability $O(o|s', a)$ at each time step.

The action a_t^i of agent i is determined by its policy $\pi_i(a_t^i|o_t^i)$, where o_t^i is the local observation of agent i at time t. The policies of all agents collectively form a joint policy $\pi = (\pi^1, ..., \pi^N)$. The objective is to maximize the expected global reward $E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$ by optimizing the joint policy π. Here, $\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$ denotes the discounted return, and r_t is the global reward at time step t based on the reward function $R(s, a)$ shared by agents. The discount factor, γ, decides if an agent prefers immediate rewards ($\gamma = 0$) or values long-term gains more ($\gamma = 1$). The value of a future reward diminishes based on γ and the time until it is received.

4 Methodology

We propose Individual Prioritized Experience Replay with Subgoals for Sparse Reward Multi-Agent Reinforcement Learning (IPERS), which utilizes individual prioritized experience replay and subgoal-oriented intrinsic rewards to address the reward sparsity issue in multi-agent reinforcement learning. Figure 1 illustrates the diagrammatic representation of the IPERS architecture.

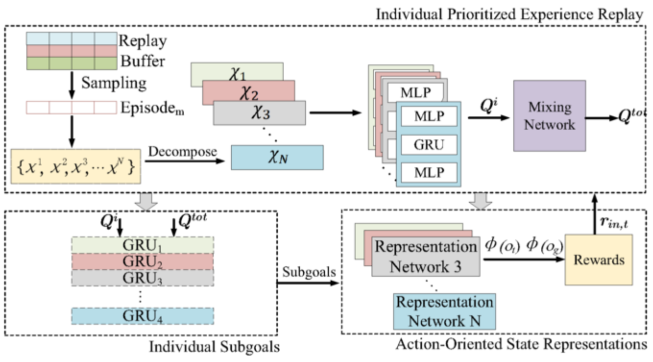

Figure 1. The overall process of the IPERS Method.

There are three modules in the IPERS scheme. The Individual Prioritized Experience Replay module starts by sampling training episodes (Episode$_m$) from the Replay Buffer using a normal distribution. Each episode contains the joint observation and action information from timestep 1 to t, denoted by $\{\chi^1, \ldots, \chi^t\}$. The joint information, χ^t, can be decomposed into distinct observation and action data relevant to the individual agent, where χ_i contains the individual observation and action information, $\{o_i, a_i\}$, for agent i over all timestep. Then, we incorporate a prioritized experience replay based on individual agents' χ_i. The prioritized experience replay assigns a priority to each state transition. This approach allows for a focus on the most beneficial experiences of each agent, thereby improving sample efficiency. The internal structure of each agent is a GRU network that processes the observation and action information, as shown in Fig. 1. Upon receiving the Q_t^i, these values are sent to the Mixing network, which enforces consistency constraints to ensure each agent's action contributes to the overall task. These mandatory consistency constraints are:

$$\frac{\partial Q^{tot}}{\partial Q^i} \geq 0, \forall i \in (1, 2, ..., N) \tag{1}$$

where Q^{tot} represents the joint action-value function, and Q^i represents the action-value function of agent i, for $i \in [1, N]$. Equation (1) ensures that global and individual rewards are non-negative. Finally, we obtain the global Q value, Q^{tot}, through the Mixing network.

In the Individual Subgoals module, agent i first leverages the individual Q and global Q^{tot} to identify the timestep that maximizes the Q value. The observation information o_g^i corresponding to this maximizing timestep is established as the target subgoal observation for the agent.

In the Action-Oriented State Representations module, we integrate a representation network within each agent to efficiently progress each agent toward its designated subgoal observation o_g^i. This module inputs the current observation o_t^i and the target subgoal observation o_g^i. It then evaluates the actionable distance between the agent's current and goal state encodings to define an intrinsic reward that facilitates faster convergence to the goal observation state o_g^i. The intrinsic rewards provide learning signals to aid agents in reaching subgoals that maximize long-term team performance in a sparse rewards environment.

4.1 Individual Prioritized Experience Replay

Figure 2 depicts the Individual Prioritized Experience Replay process, where agents receive a shared global reward after joint transitions. We decompose the joint rewards among individual agents and then derive a series of state transitions from the joint state transitions. Each state transition's TD error is calculated to evaluate its importance. A state transition with a high TD error implies that learning from the current model would be valuable, making it more likely to be chosen during training. However, the agents receive a shared

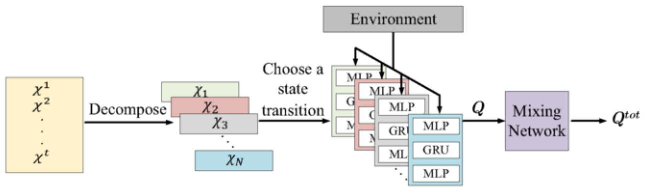

Figure 2. The process of decomposing joint actions and using individual prioritized experience replay.

global reward after joint transitions, making it challenging to distinguish the usefulness of each agent's individual experiences, especially when each agent plays a different role or has different responsibilities within the system [11]. To tackle this challenge, we employ a local utility network for each agent to get Q^i, an estimate drawn from the agent's observational data. But each agent's individual Q value is an estimate based on their behavior and may not truly depict the outcome of the team's actions. We employ the Invariance of Gradient to optimize agent networks using decomposed personal experiences and corresponding losses $\mathcal{L}^i$ for each agent i, as proposed in [12]. Therefore, our method utilizes experience decomposition using individual samples while preserving the original optimization goal, as given by:

$$\frac{\partial \mathcal{L}^{tot}}{\partial \theta_p} = \sum_{i=1}^{N} \frac{\partial \mathcal{L}^i}{\partial \theta_p}, \tag{2}$$

where θ_p represents the parameters of the local network, $\mathcal{L}^{tot}$ is the loss of Q^{tot} defined as

$$\mathcal{L}^{tot}(\theta) = [R_t^{tot} + \gamma \cdot \max_{a'} Q^{tot}(s_{t+1}, a'|\theta^-) - Q^{tot}(s_t, a_t|\theta)]^2, \tag{3}$$

where γ is the discount factor, $Q^{tot}(s_t, a_t|\theta)$ represents the global Q-value for state s_t with joint action a_t, and $\max_{a'} Q^{tot}(s_{t+1}, a'|\theta^-)$ is the maximum target global Q-value at the next state s_{t+1} over all joint actions a'. R_t^{tot} denotes the shared global reward at time step t. Additionally, due to the sparsity of the testing environment, we propose incorporating an intrinsic reward for each agent to the global reward. The definition of R_t^{tot} can be give by

$$R_t^{tot} = r_{ex,t} + \sum_{i=1}^{N} r_{in,t}^i,$$

where t denotes the timestep, $r_{ex,t}$ is the external rewards from the environment, $r^i_{in,t}$ is the intrinsic rewards as defined by

$$r^i_{in,t} = -\lambda \cdot \left\| \phi_i(o^i_t) - \phi_i(o^i_g) \right\|_2, \tag{4}$$

where λ is a hyperparameter and $\lambda \in [0, 1)$, ϕ_i, represents the representation model explained in Section 4.3, and its parameters are optimized based on the actionable distances. Here, o^i_t denotes the current observation information of agent i at timestep t, while o^i_g is the goal observation information. The network of agent i is updated using the individual TD-loss, $\mathcal{L}^i$, defined as follows:

$$\mathcal{L}^i(\theta_p) = [r^i_t + r^i_{in,t} + \gamma \cdot \max_{a'^i} Q^i(s^i_{t+1}, a'^i|\theta^-_p) - Q^i(s^i_t, a^i_t|\theta_p)]^2, \tag{5}$$

where r^i_t is the estimated individual reward for agent i, $Q^i(s^i_t, a^i_t|\theta_p)$ is the individual Q-value for agent i's state s^i_t and action a^i_t, and $\max_{a'^i} Q^i(s^i_{t+1}, a'^i|\theta^-_p)$ is the Maximum target individual Q-value at agent's next state s^i_{t+1} over actions a'^i.

By applying partial derivatives to θ_p from Eq. (3), we enable the transmission of team loss gradients through the mixing network to the agent's parameters. This process can be given by:

$$\begin{aligned}\frac{\partial \mathcal{L}^{tot}}{\partial \theta_p} &= \frac{\partial \mathcal{L}^{tot}}{\partial Q^{tot}} \cdot \sum_{i=1}^{N} \left(\frac{\partial Q^{tot}}{\partial Q^i} \cdot \frac{\partial Q^i}{\partial \theta_p} \right) \\ &= 2(R^{tot}_t + \gamma \cdot \max_{a'^i} Q^{tot} - Q^{tot}) \cdot \sum_{i=1}^{N} \left(\frac{\partial Q^{tot}}{\partial Q^i} \cdot \frac{\partial Q^i}{\partial \theta_p} \right).\end{aligned} \tag{6}$$

Similarly, we employ Eq. (5) to compute the partial derivatives of θ_p:

$$\begin{aligned}\sum_{i=1}^{N} \frac{\partial \mathcal{L}^i}{\partial \theta_p} &= \sum_{i=1}^{N} \frac{\partial \mathcal{L}^i}{\partial Q^i} \cdot \frac{\partial Q^i}{\partial \theta_p} \\ &= \sum_{i=1}^{N} 2(r^i_t + r^i_{in,t} + \gamma \cdot \max_{a'^i} Q^i - Q^i) \cdot \frac{\partial Q^i}{\partial \theta_p}.\end{aligned} \tag{7}$$

By substituting (6) and (7) into (2) and rearrange terms, we obtain:

$$r^i_t + r^i_{in,t} = (R^{tot}_t + \gamma \cdot \max_{a'} Q^{tot}_t - Q^{tot}_t) \cdot \frac{\partial Q^{tot}_t}{\partial Q^i_t} - \gamma \cdot \max_{a'} Q^i_t + Q^i_t. \tag{8}$$

Equation (8) illustrates that by estimating $r^t_i + r^i_{in,t}$ in this manner, the global experience can be decomposed into individual experiences whilst preserving parity with the initial optimization goal.

Moreover, we integrate the decentralized prioritized experience replay [10] for each agent as a means to enhance the experience replay process. Each agent interacts independently with the environment and stores transitions in a local replay buffer. Transitions are prioritized based on the TD error derived from the rewards as follows:

$$\xi^i = (r^i_t + r^i_{in,t}) + \gamma \cdot \max_{a'} Q^i_t - Q^i_t,$$

where ξ^i is the TD error for the state transitions of agent i. The transition is sampled with a probability denoted by $P(\chi^i_j)$, which can be expressed as:

$$P(\chi^i_j) = \frac{(p^i_j)^\sigma}{\sum_k (p^i_k)^\sigma}, \tag{9}$$

where χ^i_j refers to the j-th experience transition for agent i, and p^i_j is the priority of transition j and $p^i_j = |\xi^i_j| + \varepsilon$ with ε as a small constant to avoid zero TD-error. σ in Eq. (9) is a hyperparameter controlling the degree of prioritization. When the degree of priority $\sigma = 0$, it degenerates into a uniform sampling case. To correct bias, importance sampling weights [25], ω^i_j are used and is defined as

$$\omega^i_j = \mathbb{N}\left\{ (N \cdot P(\chi_{i,j}))^{-\beta} \right\},$$

where $\mathbb{N}\{\cdot\}$ is the normalization to its maximum value operation, and N is the size of the agent's replay buffer. The hyperparameter β is the exponent that controls the amount of importance sampling correction applied and is set to 0.6 in this study.

Subsets of transitions with priorities are shared across agents periodically. Received transitions are added to the local replay buffer of each agent with updated priorities. Agents then sample from a diverse shared memory, despite decentralized interaction. The exchange period determines the level of decentralization, with frequent exchanges resembling centralized replay and infrequent ones maintaining decentralization. Prioritized experience replay allows each agent, with its unique experiences, to sample key experiences for policy updates. TD error from each experience determines the sampling priority, shaping a prioritization sampling distribution based on the agent's unique experiences. This approach streamlines training individual agents' networks, drawing exclusively from each agent's experience buffer to focus on the most beneficial encounters.

4.2 *Individual Subgoals based on Q-Learning*

Subgoals have become a popular and effective technique [1] in reinforcement learning due to their utility in decomposing complex tasks into simpler steps. Subgoals offer more frequent reward signals by providing intermediate goals, accelerating learning more effectively than sparse episode completion rewards. IPERS selects multiple effective subgoals for agents from experience episodes by considering their local and global Q-values. The subgoals are generated using intrinsic network models, where models are learned to predict states with high intrinsic motivation and select these as subgoals. Figure 3 illustrates the allocation process of subgoals.

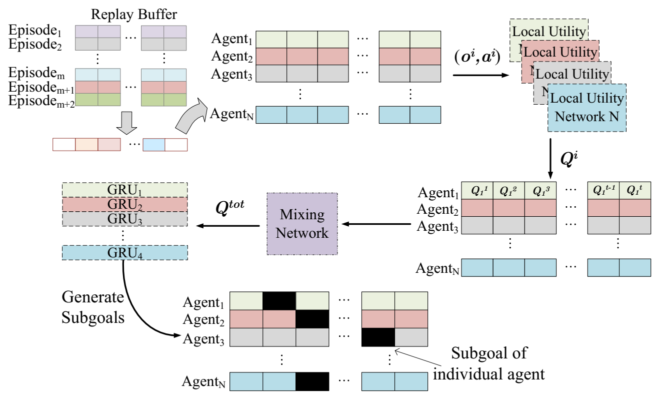

Figure 3. Subgoal Generation in the IPERS Algorithm

Upon randomly selecting training episodes for agents from the replay buffer, each agent's local observations and actions are input to its individual Q network, generating the local Q-value Q^i_t for each timestep t. Subsequently, the Mixing network receives the Q-values Q^i_t for the N agents and outputs the global Q-value Q^{tot}_t for each timestep. Ultimately, the local and global Q-values are considered to generate the subgoals for each agent throughout the episode. The objective is to predict the timestep t^i that maximizes the weighted sum of both Q_t^i and Q_t^{tot}. We introduce a policy network $\pi(t|Q;\theta)$ that selects subgoals by sampling from a learned distribution over potential subgoals conditioned on the current state. The policy is trained via gradients to maximize the expected return of the sampled subgoals according to the goal-conditioned value function. Let $\pi(t|Q;\theta)$ be a policy model parameterized by θ. This model is implemented as a GRU network that encodes the sequences of Q-values. The objective is to maximize the expected return $J(\pi)$:

$$J(\pi) = \mathbb{E}_{t\sim\pi}[\alpha \cdot Q^i(o_t^i) + (1-\alpha) \cdot Q^{tot}(o_t)],$$

where the return for timestep t is the weighted sum of local and global Q-values by the parameter α and can be optimized using the policy gradient:

$$\nabla_\theta J(\pi) = \mathbb{E}_{t\sim\pi}[(J(t)-b)\nabla_\theta \log \pi(t|Q;\theta)],$$

where b is a baseline value to reduce gradient variance, $J(t)$ is the return for timestep t, and $\nabla_\theta \log \pi(t|Q;\theta)$ is the gradient of the policy model. Once the subgoal state g^* is selected, the next step is producing the corresponding subgoal observation $o_{g^*}^i$ from the perspective of the agent. we can derive the goal state $o_{g^*}^i$ achieved by the agent upon completing the subgoal.

Training the policy with gradients allows stochastically exploring diverse subgoals, adapting to changing environments, and learning complex multi-modal selection strategies tailored to the task. By predicting subgoals that maximize local Q-value from Q-value sequences, we can generate personalized subgoals tailored to each agent's learning progress.

4.3 Learning Action-Oriented State Representations

Learning action-oriented state representations [7, 14] is well-motivated for reinforcement learning as they can simplify state and action spaces, thereby enhancing learning efficiency. We aim to extend action-oriented state representations to decentralized multi-agent reinforcement learning. These representations can simplify complex joint state and action representations and capture essential dynamics and dependencies for collaborative behavior. Taking inspiration from [7], we define the actionable distance as the function of the Q value of individual observations and actions. Q-values are ideally suited for representation learning in multi-agent reinforcement learning due to their action-centric signal. Available during the multi-agent learning process, Q-values are a ready, computation-free signal, improving over time to provide a progressively useful learning signal. Our actionable distance, D_{act}, is defined as:

$$D_{\text{act}}\left(Q_t^i, Q_{g^*}^i\right) = e^{-1-\rho},$$

where ρ is the correlation coefficient [6] that can be calculated as:

$$\rho = \frac{\text{cov}(Q_t^i, Q_{g^*}^i)}{\sigma(Q_t^i)\cdot\sigma(Q_{g^*}^i)},$$

where $\text{cov}(\cdot)$ denotes the covariance function, and $\sigma(\cdot)$ denotes the standard deviation of the random variable over all possible actions. D_{act} provides a supervised training signal to optimize the representation model $\phi_i(o_t^i)$, where o_t^i denotes the observation for agent i at time t, and distances in $\phi_i(o_t^i)$ reflect functional similarity. This enables using $\phi_i(o_t^i)$ for intrinsic rewards based on the distance. The loss function $\mathcal{L}_{\mathcal{D}}(\phi_i)$ is defined to optimize the representation model $\phi_i(o_t^i)$ by matching distances in the representation space to the actionable distances D_{act}, and it is defined as follows:

$$\mathcal{L}_{\mathcal{D}}(\phi_i) = \mathbb{E}_{o_t^i}\left[\left\|\phi_i(o_t^i) - \phi_i(o_g^i)\right\|_2 - D_{\text{act}}(o_t^i; o_g^i)\right]^2, \quad (10)$$

where the representation model $\phi_i(o_t^i)$ is trained to capture an action-oriented representation of the observation factors for decision-making.

The correlation between Q-value distributions for subgoal observations provides a statistical relationship capturing the subgoal dependencies, serving as a valuable measure for representation learning. Since the representation distances are optimized to match the actionable distances, the intrinsic reward correlates to the underlying reachability and functional similarity between observations in terms of required actions. Our intrinsic reward is given as in Eq. (4) based on the representation model $\phi_i(o_t^i)$. This reward would incentivize the agent to take actions that bring it closer to the goal, helping to guide its learning process and potentially speeding up the overall training process.

4.4 Overall Training and Evaluation

We integrate three methods to execute the overall training. Firstly, experiences are sampled from the replay buffer to optimize the team loss and update the parameters of the mixing network. The loss function, $\mathcal{L}(\theta, \phi_i)$, is defined by integrating Eq.(3) and (10):

$$\begin{aligned}\mathcal{L}(\theta,\phi_i) &= \mathcal{L}^{\text{tot}}(\theta) + \sum_{i=1}^{N} \lambda_{\mathcal{D}} \cdot \mathcal{L}_{\mathcal{D}}(\phi_i) \\ &= \left[R_t^{\text{tot}} + \gamma \cdot \max Q_t^{\text{tot}}(s_{t+1}, a') - Q_t^{\text{tot}}(s_t, a_t)\right]^2 \\ &\quad + \sum_{i=1}^{N} \lambda_{\mathcal{D}} \cdot \mathbb{E}_{o_t^i}\left[\left\|\phi_i(o_t^i) - \phi_i(o_g^i)\right\|_2 - D_{\text{act}}(o_t^i, o_g^i)\right]^2,\end{aligned}$$

where $\mathcal{L}(\theta, \phi_i)$ serves to refine the prediction of Q^{tot} using the shared global reward, while $\lambda_{\mathcal{D}}$ acts as a hyperparameter that regulates the weight of the reconstruction loss, typically a small value preventing the collapse of representations.

Next, the joint experiences are decomposed into individual experiences. The TD error from each experience determines the sampling priority, thereby shaping a priority sampling distribution based on the agent's unique experiences. Consequently, the algorithm optimizes each agent's network parameters through $\mathcal{L}_{\text{ind}}(\theta_p)$, as defined by

$$\begin{aligned}&\mathcal{L}_{\text{ind}}(\theta_p) \\ &\quad = \sum_j \omega_j^i \cdot \left[r_j^i + r_{in,j}^i + \gamma \max Q_j^i(s_{j+1}, a') - Q_j^i(s_j, a)\right]^2,\end{aligned}$$

where j is the index of the agent's experience transition. This loss function allows each agent to learn state transitions that benefit its model and improve the convergence performance. During training, we utilize Q_t^i in conjunction with the global Q^{tot} value to identify subgoals. Once each agent's subgoal is determined, we employ the actionable distance representation ϕ_i to generate the intrinsic rewards $r_{in,t}^i$ for each agent, guiding the agent's behavior and learning, even in sparse reward environments.

In the evaluation phase, every agent i independently employs its learned policy, sourced from its network parameters. This facilitates decentralized execution, where the agents determine actions solely based on their local observations and respective policies.

5 Experiments

To assess the effectiveness of the proposed approach, we have conducted experiments using the Starcraft II multi-agent environment (SMAC) [15] and Google research football (GRF) environment [16]. We evaluate our method in SMAC across four maps with various difficulties: 3m (easy), 3s_vs_3z (normal), MMM (normal), and 10m_vs_11m (hard). We employ both dense and sparse reward settings in these scenarios. We also examine IPERS in two distinct GRF scenarios: a 3_vs_1 setup with a goalkeeper and a challenging counterattack scenario. In this case, only SCORING rewards are applied to simulate a sparse reward situation. To benchmark the performance of our proposed algorithm, we compare our IPERS against four state-of-the-art multi-agent reinforcement learning algorithms, including PER [25], QMIX [24], MASER [14], and DIFFER [12]. The codes of IPERS for SMAC and GRF are also available at https://github.com/zyfdesign/IPERS-SMAC and https://github.com/zyfdesign/IPERS-GRF.

5.1 *Experimental settings*

Table 1 summarizes the reward settings for SMAC. Both the dense and sparse reward scenarios allocate a positive reward of 200 points to the agent when all enemy units are defeated. In the dense reward scenario, the agent obtains an additional 10 points for each enemy unit it defeats and is penalized by 5 points when a teammate falls. In the end, the agent's reward is adjusted based on the remaining health of enemies and teammates. The agent's reward is decreased by the enemy's remaining health points and increased by the remaining health of its teammates. On the other hand, in the sparse reward setting, the agent is not rewarded or penalized for defeating individual enemy units or losing teammates. Furthermore, the final reward calculation disregards the health statuses of both allies and enemies. Under these conditions, the agent can only earn rewards by eliminating all enemy units, thereby securing a total of 200 points. The parameter α is initialized to 0.5 for an equal blend of local and global Q-values, enabling the policy gradient to learn the task-specific balance during training. The parameter λ is set to 0.03 and $\lambda_{\mathcal{D}}$ is 0.001. Proper balancing through an ablation study grid-searching λ and $\lambda_{\mathcal{D}}$ have been evaluated for the impact on overall training efficiency and final performance.

Table 1. SMAC Reward Settings

	Dense reward	Sparse reward
Defeat all enemies	$+200$	$+200$
Defeat an enemy	$+10$	0
One teammate died	-5	0
Enemy's Health	-Enemy's Health	0
Teammate's Health	+Teammate's Health	0

In GRF environments, there are two reward types: SCORING and CHECKPOINT. SCORING gives a -1 reward for scoring a goal and a negative reward when the other team scores. Its effects may not be apparent in early training stages due to the need for a long sequence of events. CHECKPOINT tackles the sparsity issue by providing dense, incremental rewards to reinforce key sub-goals. However, it is deactivated in our experiments to simulate a sparse rewards environment.

5.2 *Experimental results*

The results for two different reward settings are depicted in this section. We repeated experiments five times for each map to maintain reliability and consistency. The solid line shows the average value of all trials, while the upper and lower boundaries signify the maximum and minimum values from the five experiments.

5.2.1 *The performance in SMAC tasks*

Results in the dense reward setting: Fig. 4 compares IPERS with four state-of-the-art MARL algorithms on SMAC maps of varying difficulty levels. IPERS exhibits rapid convergence during early stages of rising complexity. While all algorithms show similar success rates on simpler maps, IPERS excels on more challenging ones. On the Easy 3m map, IPERS reaches convergence at half a million steps with an average win rate of around 95%, outperforming MASER and PER with a 90% win rate at the same point. QMIX and DIFFER take one million timesteps to achieve an average 90% win rate. For the Normal 3s_vs_3z and MMM maps, IPERS achieves an average win rate of over 98% after one million steps, surpassing MASER and QMIX with their 90% average win rates at convergence. DIFFER converges after 1.5 million steps, while PER takes up to 2 million timesteps. On the Hard 10m_vs_11m map, IPERS maintains superior convergence speed and average win rate compared to other algorithms. After 1.5 million timesteps, IPERS achieves a 75% win rate, while MASER, QMIX, and PER reach around 60%. DIFFER performs the worst with an average win rate under 40%.

Results in the sparse reward setting: The performance of all analyzed algorithms is affected by reward sparsity. In the Easy 3m map, we observe that PER and QMIX undergo the most considerable performance reduction compared to the dense reward setting. Despite MASER's utilization of subgoals to partially counteract the impact of reward sparsity, it still experiences a performance degradation of approximately 30%. Our IPERS algorithm remains the least affected, with only a 5% decrease in the average win rate. Even in the Hard 10m_vs_11m map, where IPERS achieves an average win rate of 50%, it continues to outperform the compared algorithms. As the complexity of the Normal maps (3s_vs_3z and MMM) increases, the convergence speed of all algorithms is significantly hampered during the initial phases. However, our IPERS algorithm maintains its leading position in convergence rate, stabilizing at an average win rate of 95% after 1.5 million timesteps and effectively addressing the impact of reward sparsity on its performance. In the Hard 10m_vs_11m map, the average win rate of all algorithms suffers considerably. PER and QMIX achieve a win rate of merely 40% after 3 million timesteps, indicating their struggle to handle the Hard difficulty level. Although MASER and DIFFER algorithms exhibit superior convergence rates, they still lag 5% to 10% behind our IPERS algorithm. Notably, IPERS exhibits rapid improvements in win rate during early convergence and maintains an unmatched high average win rate upon convergence after 1.5 million timesteps across all map variations.

Fig. 4 and Fig. 5 highlight how reward sparsity negatively affects the convergence speed and average win rate of multi-agent algorithms. Value decomposition algorithms may function well in dense reward environments but falter when reward signals are scarce, resulting in low average win rates during the early stages of training, especially on Normal and Hard maps. While MASER somewhat counters the impact of sparse rewards by assigning subgoals using Q values, it grapples with the issue of inadequate reward signals upon fulfilling subgoals. DIFFER and PER, both using the experience replay mechanism to handle dense rewards, face considerably slower convergence speeds in sparse reward conditions during early training phases. Our IPERS algorithm integrates the advantages of subgoals and prioritized experience replay techniques. This fusion expedites

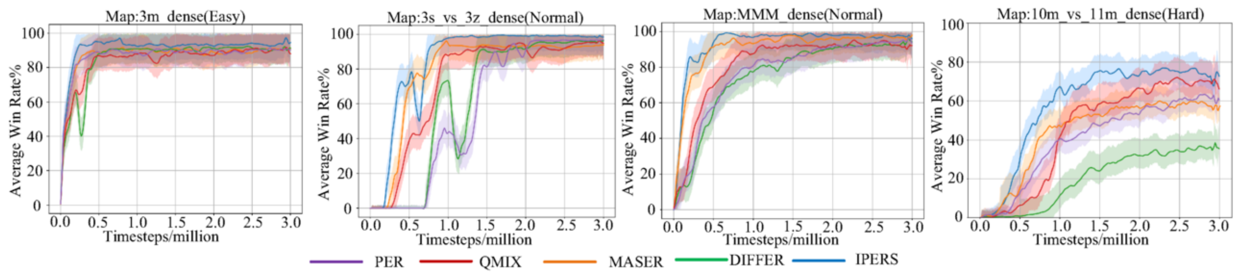

Figure 4. Experimental results of SMAC in dense reward settings

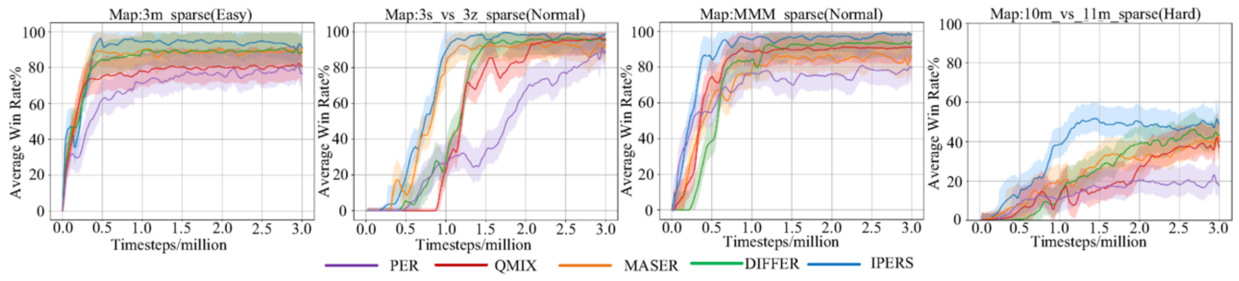

Figure 5. Experimental results of SMAC in sparse reward settings

the exploration of successful strategies and enhances the model by learning from significant state transitions. As anticipated, our proposed IPERS method excels in terms of convergence speed and is less affected by sparse reward information at convergence, thereby outperforming the four leading-edge algorithms we compared it against.

5.2.2 *The performance in GRF tasks*

We also evaluate IPERS and other algorithms in two Google Research Football (GRF) [16] academic scenarios: 3_vs_1_with_keeper and counterattack_hard. We manage the players on the left team, with the exception of the goalkeeper. The right team players are operated by built-in, rule-based bots. The agents are required to coordinate their positions to stage attacks, and rewards are exclusively given for scoring. The observations can be broken down into five parts: ball information, left team data, right team data, details of the controlled player, and the match state. We employ settings similar to those used in [9], where agents can choose from 19 discrete actions, including running, sliding, shooting, and passing. We evaluate five different random seeds and average the results for presentation.

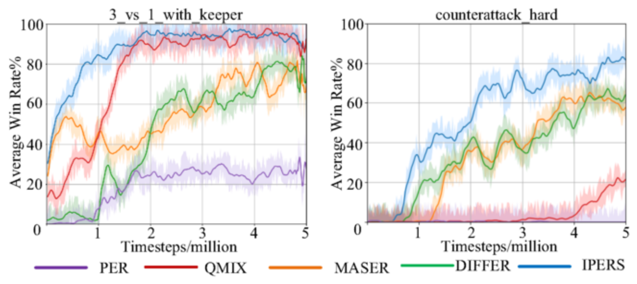

Figure 6. Experimental results of Google Research Football

Figure 6 shows the experimental results of our IPERS method compared to four algorithms: PER, QMIX, MASER, and DIFFER. IPERS achieves the highest average win rate and the fastest convergence speed during early training stages. In the 3_vs_1_with_keeper scenario, IPERS converges at 2 million timesteps with the highest average win rate. QMIX has a similar convergence speed but a slightly lower average win rate. MASER and DIFFER converge slower, around 4 million timesteps, with significantly lower average win rates. PER's average win rate remains around 30% at 5 million timesteps. In the counterattack_hard scenario, IPERS outperforms other algorithms in both convergence performance and average win rate. MASER and DIFFER lag behind IPERS, and QMIX and PER struggle to achieve commendable results. In conclusion, our proposed IPERS method excels in the GRF challenges, displaying superiority in both average win rate and convergence performance.

6 Conclusions

This work presents Individual Prioritized Experience Replay with Subgoals for Sparse Reward Multi-Agent Reinforcement Learning (IPERS). IPERS addresses reward sparsity by combining joint action decomposition and prioritized experience replay, thereby facilitating efficient decentralized training from a shared global reward signal. Personalized subgoals are generated for each agent by predicting the timestep that maximizes weighted local and global Q-values. These subgoals provide helpful intermediate rewards to guide exploration. In addition, IPERS also employs learning of goal-oriented state representations to simplify the observation space, extract meaningful features, and offer an efficient intrinsic reward to guide agents. Experiments conducted using the StarCraft II Multi-Agent Challenge and Google Research Football academic scenarios demonstrate that IPERS significantly outperforms four leading algorithms, demonstrating improved convergence speed and overall win rate performance in sparse reward environments.

References

[1] Elliot Chane-Sane, Cordelia Schmid, and Ivan Laptev, 'Goal-conditioned reinforcement learning with imagined subgoals', in *Pro-*

ceedings of the 38th International Conference on Machine Learning, 18-24 July 2021, volume 139, pp. 1430–1440. PMLR, (2021).

[2] Jiayu Chen, Yuanxin Zhang, Yuanfan Xu, et al., 'Variational automatic curriculum learning for sparse-reward cooperative multi-agent problems', in *Advances in Neural Information Processing Systems*, volume 34, pp. 9681–9693, (2021).

[3] Rui Chen, Peide Huang, and Laixi Shi, 'Latent goal allocation for multi-agent goal-conditioned self-supervised imitation learning', *Advances in Neural Information Processing Systems*, (2021).

[4] Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht, 'Shared experience actor-critic for multi-agent reinforcement learning', in *Advances in Neural Information Processing Systems*, volume 33, pp. 10707–10717, (2020).

[5] Tianhong Dai, Yali Du, Meng Fang, and Anil Anthony Bharath, 'Diversity-augmented intrinsic motivation for deep reinforcement learning', *Neurocomputing*, **468**, 396–406, (2022).

[6] Dominic Edelmann, Tamás F. Móri, and Gábor J. Székely, 'On relationships between the pearson and the distance correlation coefficients', *Statistics and Probability Letters*, **169**, 108960, (2021).

[7] Dibya Ghosh, Abhishek Gupta, and Sergey Levine, 'Learning actionable representations with goal conditioned policies', in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, (2019).

[8] Sven Gronauer and Klaus Diepold, 'Multi-agent deep reinforcement learning: a survey', *Artificial Intelligence Review*, 1–49, (2022).

[9] Jianye Hao, Xiaotian Hao, Hangyu Mao, Weixun Wang, Yaodong Yang, Dong Li, Yan Zheng, and Zhen Wang, 'Boosting multiagent reinforcement learning via permutation invariant and permutation equivariant networks', in *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, (2023).

[10] Dan Horgan, John Quan, David Budden, et al., 'Distributed prioritized experience replay', in *6th International Conference on Learning Representations, ICLR, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, (2018).

[11] Siyi Hu, Chuanlong Xie, Xiaodan Liang, and Xiaojun Chang, 'Policy diagnosis via measuring role diversity in cooperative multi-agent RL', in *International Conference on Machine Learning, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9041–9071. PMLR, (2022).

[12] Xunhan Hu, Jian Zhao, Wengang Zhou, and Houqiang Li, 'DIFFER: Decomposing individual reward for fair experience replay in multi-agent reinforcement learning', *arXiv preprint arXiv:2203.13319*, (2023).

[13] David Isele and Akansel Cosgun, 'Selective experience replay for lifelong learning', in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 3302–3309, (2018).

[14] Jeewon Jeon, Woojun Kim, Whiyoung Jung, and Youngchul Sung, 'MASER: Multi-agent reinforcement learning with subgoals generated from experience replay buffer', in *International Conference on Machine Learning*, pp. 10041–10052. PMLR, (2022).

[15] Muhammad Junaid Khan, Syed Hammad Ahmed, and Gita Sukthankar, 'Transformer-based value function decomposition for cooperative multi-agent reinforcement learning in starcraft', in *Proceedings of the Eighteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pp. 113–119, (2022).

[16] Karol Kurach, Anton Raichuk, et al., 'Google research football: A novel reinforcement learning environment', in *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 4501–4510, (2020).

[17] Zhixuan Liang, Jiannong Cao, Shan Jiang, et al., 'Hierarchical reinforcement learning with opponent modeling for distributed multi-agent cooperation', in *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, pp. 884–894. IEEE, (2022).

[18] Iou-Jen Liu, Unnat Jain, Raymond A. Yeh, and Alexander G. Schwing, 'Cooperative exploration for multi-agent deep reinforcement learning', in *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 6826–6836. PMLR, (2021).

[19] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch, 'Multi-agent actor-critic for mixed cooperative-competitive environments', in *Advances in Neural Information Processing Systems, December 4-9, 2017, Long Beach, CA, USA*, pp. 6379–6390, (2017).

[20] Washim Uddin Mondal, Mridul Agarwal, Vaneet Aggarwal, et al., 'On the approximation of cooperative heterogeneous multi-agent reinforcement learning (MARL) using mean field control (MFC)', *Journal of Machine Learning Research*, **23**(129), 1–46, (2022).

[21] Isack Thomas Nicholaus and Dae-Ki Kang, 'Robust experience replay sampling for multi-agent reinforcement learning', *Pattern Recognit. Lett.*, **155**, 135–142, (2022).

[22] Guido Novati and Petros Koumoutsakos, 'Remember and forget for experience replay', in *International Conference on Machine Learning*, pp. 4851–4860. PMLR, (2019).

[23] Frans A Oliehoek and Christopher Amato, *A concise introduction to decentralized POMDPs*, Springer, 2016.

[24] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, et al., 'Monotonic value function factorisation for deep multi-agent reinforcement learning', *The Journal of Machine Learning Research*, **21**(1), 7234–7284, (2020).

[25] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver, 'Prioritized experience replay', in *4th International Conference on Learning Representations. ICLR*, (2016).

[26] Mathieu Seurin, Florian Strub, Philippe Preux, and Olivier Pietquin, 'Don't do what doesn't matter: Intrinsic motivation with action usefulness', in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI, Virtual Event / Montreal, Canada, 19-27 August 2021*, pp. 2950–2956, (2021).

[27] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi, 'QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning', in *International conference on machine learning*, pp. 5887–5896. PMLR, (2019).

[28] Peiquan Sun, Wengang Zhou, and Houqiang Li, 'Attentive experience replay', in *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 5900–5907, (2020).

[29] Peter Sunehag, Guy Lever, et al., 'Value-decomposition networks for cooperative multi-agent learning based on team reward', in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS*, pp. 2085–2087, (2018).

[30] Li Wang, Yupeng Zhang, Yujing Hu, Weixun Wang, Chongjie Zhang, Yang Gao, Jianye Hao, Tangjie Lv, and Changjie Fan, 'Individual reward assisted multi-agent reinforcement learning', in *International Conference on Machine Learning*, pp. 23417–23432. PMLR, (2022).

[31] Yajie Wang, Dianxi Shi, Chao Xue, Hao Jiang, Gongju Wang, and Peng Gong, 'AHAC: actor hierarchical attention critic for multi-agent reinforcement learning', in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 3013–3020. IEEE, (2020).

[32] Zaipeng Xie, Cheng Ji, and Yufeng Zhang, 'Deep skill chaining with diversity for multi-agent systems', in *Artificial Intelligence - Second CAAI International Conference, CICAI 2022, Beijing, China, August 27-28, 2022, Revised Selected Papers, Part III*, volume 13606 of *Lecture Notes in Computer Science*, pp. 208–220. Springer, (2022).

[33] Huanhuan Yang, Dianxi Shi, Chenran Zhao, Guojun Xie, and Shaowu Yang, 'CIExplore: Curiosity and influence-based exploration in multi-agent cooperative scenarios with sparse rewards', in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 2321–2330, (2021).

[34] Yaodong Yang, Jianye Hao, Ben Liao, et al., 'Qatten: A general framework for cooperative multiagent reinforcement learning', *arXiv preprint arXiv:2002.03939*, (2020).

[35] Shangtong Zhang and Richard S. Sutton, 'A deeper look at experience replay', *arXiv preprint arXiv:1712.01275*, (2017).