Self-Motivated Multi-Agent Exploration

Shaowei Zhang*
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, P.R.China
zhangsw@lamda.nju.edu.cn

Jiahuan Cao*
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, P.R.China
caojh@lamda.nju.edu.cn

Lei Yuan
National Key Laboratory for Novel Software Technology, Nanjing University; Polixir Technologies, Nanjing, P.R.China
yuanl@lamda.nju.edu.cn

Yang Yu
National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, P.R.China
yuy@nju.edu.cn

De-Chuan Zhan†
National Key Laboratory for Novel Software Technology, Nanjing University; Polixir Technologies, Nanjing, P.R.China
zhandc@nju.edu.cn

ABSTRACT

In cooperative multi-agent reinforcement learning (CMARL), it is critical for agents to achieve a balance between self-exploration and team collaboration. However, agents can hardly accomplish the team task without coordination and they would be trapped in a local optimum where easy cooperation is accessed without enough individual exploration. Recent works mainly concentrate on agents’ coordinated exploration, which brings about the exponentially grown exploration of the state space. To address this issue, we propose Self-Motivated Multi-Agent Exploration (SMMAE), which aims to achieve success in team tasks by adaptively finding a trade-off between self-exploration and team cooperation. In SMMAE, we train an independent exploration policy for each agent to maximize their own visited state space. Each agent learns an adjustable exploration probability based on the stability of the joint team policy. The experiments on highly cooperative tasks in StarCraft II micromanagement benchmark (SMAC) demonstrate that SMMAE can explore task-related states more efficiently, accomplish coordinated behaviours and boost the learning performance.

KEYWORDS

Multi-agent Reinforcement Learning; Self-motivated Exploration; Multi-agent Cooperation

1 INTRODUCTION

Cooperative multi-agent reinforcement learning (CMARL) has achieved outstanding results [13, 25] and has been applied to many real-world applications, such as autonomous driving [54], multi-agent path finding [12], natural language processing tasks [42], and dynamic algorithm configuration [47]. Most of the methods can be divided into two categories: the value-based methods [34, 36, 43] and others based on policy gradient [10, 20, 49]. The value-based methods and their variants [6, 50, 51] can have better performance in complex tasks like the StarCraft II micromanagement benchmark [35]. However, they usually neglect efficient exploration, which is a particularly challenging problem in complex scenarios. Agents will be trapped in a local optimum if they only focus on team collaboration and cannot conduct coordinated behaviours if they only pay attention to exploration.

To address the problem of MARL exploration, a vanilla technique that is commonly used in value-based MARL algorithms is $\varepsilon$-greedy [13, 48]. Recently, a variety of methods have been proposed and they focus more on coordinated exploration. EITI & EDTI [45] use the interactions between the agents to measure the collaborative exploration. CMAE [19] focuses on the low-dimensional space that indeed affects the team cooperation reward. EMC [53] proposes that the local Q function is affected by the influence of other agents and guides the exploration based on this viewpoint. However, these algorithms ignore individual exploration and consider only joint exploration objective, which has a very large space and is not as efficient as individual exploration.

The mentioned methods can facilitate the MARL exploration ability in someway but are inefficient in complex scenarios, as those methods seldom consider self-exploration from an individual point of view. What’s more, they are far away from human coordination.
If a person is not familiar with the team behaviour in the current situation, one will first learn how to cooperate with others. Otherwise, one will explore more unknown situations to cooperate better with others in more situations [16]. Although those mentioned MARL exploration approaches can achieve coordination improvements in some MARL tasks by designing complex modules, we find that we can achieve competitive results with vanilla MARL methods such as QMIX [34], by simply focusing on self-exploration.

For example, in the widely-used benchmark SMAC [35] almost all existing MARL methods fail to succeed in 2 million steps in the challenging scenarios such as maps 3s5z_vs_3s6z and 6h_vs_8z, where sufficient exploration is required [34, 36, 43, 44, 53]. However, in our experiments, by adjusting the degree of self-exploration, we find that QMIX [34] and VDN [36] can achieve significant improvements in exploration and can succeed in super hard tasks (Figure 1). Concretely, here the self-exploration is adjusted by changing the $\epsilon$ end value of $\epsilon$-greedy (Figure 12 in Appendix C) which starts with the value 0.1 and decays linearly with time to a fixed end value $\epsilon_T$ in the popular implementation of QMIX [35]. The results show that the $\epsilon$ value has a strong association with the degree of exploration, and the probability that the agent adopts an exploration policy at one step during training time, should be applied at different timesteps during training to trade off between self-exploration and team cooperation to for high coordination. However, to achieve this, it needs extensive work for hyper-parameter tuning at each timestep, and it is not tractable to manually decide which $\epsilon$ value is suitable at each timestep in complex tasks.

Towards designing a method that can adaptively adjust the degree of exploration at different timesteps, we propose a novel multi-agent exploration method Self-Motivated Multi-Agent Exploration (SMMAE). Inspired by the widely used uncertainty measurement from single agent Reinforcement Learning [18, 31], we posit that learning a proper uncertainty about the multi-agent system and employing it to promote individual exploration can facilitate coordination in CMARL. In a multi-agent system, the uncertainty between agents’ actions and others’ observations is limited, the agents should explore individually to jump out of the local optimum. On the contrary, the agents’ lack of awareness of others makes it hard to achieve coordinated behaviors when the uncertainty in the multi-agent system is high, and the agents should explore less and learn how to cooperate first before exploring more. Specifically, the uncertainty of the joint policy for the agents can be measured using the correlation between each agent’s action and observations of other agents. We then use mutual information [17] to denote the correlation. The observation of other agents is used to predict the current agent’s action, and the cross entropy loss is used as a criterion. The smaller the cross entropy loss is, the less the uncertainty is. Our main contributions are:

- We study the exploration probability in MARL and reveal that the suitable $\epsilon$ value at different timestep has a huge valuable influence on exploration behaviours and final performance, especially in complex environments.
- SMMAE can adaptively adjust exploration probability according to the uncertainty in the multi-agent system to trade off between self-exploration and team cooperation.
- The experiments demonstrate the strong ability of SMMAE to explore task-related state space efficiently.

2 BACKGROUND

2.1 Cooperative Multi-Agent System Model

This paper considers a fully cooperative multi-agent system, where all the agents need to cooperate with each other to earn a shared team reward. The system can be modelled by a Dec-POMDP [23] tuple $G = (N, S, A, P, R, \Omega, O, n, \gamma)$, where $N$ is a finite set of $n$ agents, $s \in S$ is a global state of the environment in the set of possible states, $A$ is the finite action set and $\gamma \in [0, 1)$ is the discount factor. Due to the partially observable settings, agent $i \in N$ is only accessible to a local observation $o_i \in \Omega$ according to the observation function $O(s,i)$. Each agent has an observation-action trajectory history $\tau_i \in \mathcal{T} \equiv (\Omega \times A)^*$. At each timestep the joint action of the team is $a = (a_1, \ldots, a_n) \in A \equiv \mathcal{A}^n$ where $a_i \in \pi_i(a | \tau_i)$ is an action selected by each agent $i$. The team joint action will lead to the next global state $s'$ according to the environment transition function $P(s' | s, a)$ and the team will earn the shared reward $r = R(s, a)$. The joint policy $\pi \equiv (\pi_1, \ldots, \pi_n)$ aims to maximize the joint value function $V^\pi(s) = \mathbb{E}_{s_0,\omega_0} [\sum_{t=0}^{\infty} \gamma^t R_t | s_0 = s, \pi]$, which induces a joint action-value function $Q^\pi(s, a) = R(s, a) + \gamma \mathbb{E}_r [V^\pi(s')]$.

2.2 Centralized Training with Decentralized Execution (CTDE) paradigm

Centralized Training with Decentralized Execution (CTDE) [24] has been a popular paradigm for cooperative multi-agent reinforcement learning. In a CTDE paradigm, during training the global states are available for all the agents by using a centralized controller, while during test time each agent has to use its local network to select an action based on its local trajectories. Deep Q Network [22] and its derivatives [40, 46] have achieved great performance in reinforcement learning tasks, especially in game tasks. In the multi-agent system, the tuple in replay buffer $D$ is $(s_t, a_t, r_t, s_{t+1})$ and the joint action-value function is $Q_{\text{tot}}(\tau, a; \theta)$, where $\theta$ are the parameters of the Q network and will be learnt by the following Temporal Difference (TD) error [37]:

$$L(\theta) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim D} \left[ r + \gamma \max_{a_{t+1}} Q_{\text{tot}}(\tau_{t+1}, a_{t+1}; \theta^{-}) - Q_{\text{tot}}(\tau_t, a_t; \theta) \right]^2,$$

where $\theta^{-}$ are the parameters of the target network and will be updated by $\theta$ periodically. Considering the CTDE paradigm, many works [33, 34, 36, 43] adopt the decomposition structure between the joint action-value function $Q_{\text{tot}}$ and local action-value function $Q_i$ and obtain good performance.

3 RELATED WORK

3.1 Exploration in Reinforcement Learning

Various methods for exploration have been studied in single-agent reinforcement learning [48]. Some works explore by focusing on the environment dynamics. ICM [28] adopts both forward and inverse models to build a good feature space for curiosity and explore what is controlled by or affects the agent. Pathak et al. [29] propose to use an ensemble of dynamics models. They use the variance over the output of these networks in the ensemble to induce the exploration. There are some works focused on the environment novelty to induce exploration. Some works are the generation of
count-based methods and use density models which can generate pseudo-counts of visited states to measure the uncertainty of the agents [3, 26]. Some works use prediction error to reflect the novelty of the states. RND [5] uses a fixed randomly initialized network to get the embedding of the state, then uses a learnable network to reconstruct this embedding, and the reconstruction loss will be adopted as an intrinsic reward to guide exploration. Based on this, NovelD [52] proposes to use the difference of RND novelty between adjacent time step as the intrinsic reward, and adds a restriction in an episode so that the agent could earn a reward only when it visits the states for the first time. Therefore, NovelD prefers to explore unexplored boundary states, thereby exploring more valuable state regions. SMM [18] aims to learn a state marginal distribution that matches the prior distribution and uses this as the target of exploration. Some works adopt epsilon schedule for better exploration. Tokic [38] uses the difference between value functions to adjust the size of epsilon. When the agent’s knowledge becomes certain about the environment, the degree of exploration will be reduced. εz-greedy [8] replaces a single action with a sequence of actions (option), choosing an option instead of choosing a random action with probability ε. Some works focus on noise for better exploration. NoisyNet [11] replaces the fully connected value network with a learnable noise network. Flappert et al. [32] add adaptive-scale noise to the parameters of the policy. There are also some works that study exploration from some interesting perspectives. Eysenbach et al. [9] use diversity-driven exploration to learn distinguishable and diverse skills, while SMiRL [4] reduces the entropy of the visited states during exploration to exclude negative effects of perturbations and acquire complex behaviours and skills without supervision. C-BET [27] first learns a exploration policy across environments without extrinsic rewards, then transfers the learned exploration policy to the target tasks. Pislar et al. [30] study the problem when the agent should explore and reveal the results of the methods with different types of exploring temporal structure. Agent57 [1] adopts a more engineering approach. The Q network is divided into an intrinsic part and an extrinsic part. NGU is a policy that treats different degrees of exploration equally [2]. Based on NGU, Agent57 uses a meta-controller to select the policy adaptively.

3.2 Multi-Agent Exploration

There are also many studies on exploration in multi-agent reinforcement learning. To achieve committed exploration, MAVEN [21] adopts hierarchical control and the policies of the agents are conditioned on the shared latent variable generated by a hierarchical policy. Wang et al. [45] propose two exploration methods, EITI and EDTI, to induce cooperative exploration by capturing the influence of one agent’s on other agents. EITI quantifies the influences on the state transition dynamics, while EDTI quantifies both transition and reward influences. However, they are not scalable as they need to use a approximation way to measure the influence of other agents on one agent when there are many agents, which can cause the method to fail. CMAE [19] proposes that reward function only depends on a small subset of the large state space. Therefore, it first explores in the projected low-dimensional space of the high-dimensional state space, then select goals from the low-dimensional space and train the exploration policies to reach the goal to explore higher-dimensional space continuously. However, it’s hard to find the effective low-dimensional projection in complex MARL tasks. EMC [53] proposes that local Q function of each agent can capture the novelty of states and the influences between agents. To induce coordinated exploration, EMC proposes to use prediction errors of individual Q function as the intrinsic reward. However, EMC is also not scalable, as it has to maintain a huge episodic memory buffer.

4 METHODOLOGY

In this section, we introduce Self-Motivated Multi-Agent Exploration (SMMAE), a novel method for effective exploration in MARL. Figure 2 shows the whole SMMAE framework. Part (a) and (b) in Figure 2 are the common structures in Centralized Training with Decentralized Execution (CTDE) algorithms using value decomposition. Each agent i uses its own Q function and action as input and outputs local Q-value to select action. The mixing network takes all the outputs of the agents to train the multi-agent policy during training time. SMMAE mainly changes the structure of each individual agent in part (b), and the main contribution of our method is illustrated in part (c).

4.1 Adaptive Exploration

In ε-greedy, the value of ε will decay linearly to a fixed end value εT in a common implementation. However, the experiments prove that exploration probability with fixed end value is not efficient (Figure 1). Therefore, we adopt adaptive exploration probability. As the exploration probability ε is for all agents, we use α = (α1, ⋯, αn) to denote the exploration probabilities of n agents, where αi represents the probability of agent i to select the exploration policy.

We associate α with the uncertainty of the multi-agent system (Part (c) in Figure 2). The agents cooperate well when the uncertainty of the multi-agent system is limited and they should increase the exploration intensity to jump out of the local optimum by increasing αi. When the system uncertainty is high, the agents lack awareness of others and are hard to achieve coordinated behaviours. Under this condition, agents should take more exploitation to reduce the uncertainty by decreasing αi.

The uncertainty of the multi-agent system can be measured by the correlation between agents. When the correlation between agent i’s action a_i and the other agents’ observation o_{-i} is low, the uncertainty of the multi-agent system is high because the agents focus more on their own action selection than the team cooperation. We propose to use the mutual information \( I (a_i, o_{-i}) \) to reflect the correlation between the agent i’s action a_i and the other agents’ observation o_{-i}:

\[
F(o, a) = \sum_{i=1}^{n} I (a_i, o_{-i}) = \sum_{i=1}^{n} \mathcal{H} (a_i) - \mathcal{H} (a_i | o_{-i}),
\]

where \( \mathcal{H} \) denotes entropy. Then we can derive a lower bound for the team mutual information term using a variational estimator:

\[
F(o, a) = \sum_{i=1}^{n} \mathbb{E}_{a_i \sim p(a_i | o_{-i})} \left[ \log p (a_i | o_{-i}) \right] - \mathbb{E}_{a_i \sim p(a_i)} \left[ \log p (a_i) \right] \\
\geq \sum_{i=1}^{n} \mathbb{E}_{a_i \sim p(a_i | o_{-i})} \left[ \log q_{\xi_i} (a_i | o_{-i}) \right] - \mathbb{E}_{a_i \sim p(a_i)} \left[ \log p (a_i) \right],
\]

(3)
where $\tilde{q}(a_i | o_{-i})$ is a variational posterior estimator of $p(a_i | o_{-i})$ with parameter $\xi_i$. As $p(a_i)$ is the prior of actions, the last term is a constant and can be ignored. Therefore, we only need to consider the first term. We use a network to represent $p(a_i | o_{-i})$, where the input is $o_{-i}$ and the output is $\tilde{a}_i$. Then the first term $\mathbb{E}_{a_i \sim p(a_i | o_{-i})} \log \tilde{q}(a_i | o_{-i})$ is the network’s negative cross entropy loss $L_{ce}(a_i, \tilde{a}_i | o_{-i})$.

Based on this, we use a two-level heuristic method to adjust $a_t$:

$$a_{\text{new}} = \begin{cases} a_{\text{low}}, & L_{ce}(a_i, \tilde{a}_i | o_{-i}) \geq L_{\text{threshold}} \\ \min \left( a_{\text{old}}, a_{\text{high}} \right), & \text{else}, \end{cases}$$

(4)

where $a_{\text{low}}$ and $a_{\text{high}}$ are the lower bound value and upper bound value, respectively. $\lambda_\alpha$ is the scaling parameter for $\alpha$ increasing steps and it takes $\lambda_\alpha$ steps to increase from the lower bound to the upper bound, and $L_{\text{threshold}}$ is the loss threshold. For the policy convergence, $a_{\text{high}}$ will decrease linearly to $a_{\text{low}}$ at the end of the training. For stability, we update $\alpha$ every $E_{\text{tr}}$ episodes.

We need to mask agent $i$’s observation as we use other observation $o_{-i}$ to predict agent $i$’s action. It means computing $L_{ce}(a_i, \tilde{a}_i | o_{-i})$ for each agent $i$ requires optimizing $n$ networks simultaneously and will consume large amounts of computing resources. Instead, we utilize the structure of the attention mechanism [41] to estimate all the $n$ losses within one network:

$$g^{\text{att}}(a_i) = \max \left( \frac{\text{Mask}(Q^{\text{att}}(K^{\text{att}})^T)}{d_k} \right) V_i^{\text{att}},$$

(5)

where $Q^{\text{att}} = o_i W^Q, K^{\text{att}} = o_i W^K, V^{\text{att}} = o_i W^V$ and $W^Q, W^K, W^V$ denote the parameter matrix for the attention mechanism. Here $d_k$ is the attention key size. As we use other observation $o_{-i}$ of agent $i$ to predict, the function $\text{Mask}(X)$ will mask the correlation matrix and set the diagonal of the matrix to 0 (Figure 2). And the action prediction is: $a_i = f_a(g^{\text{att}}(a_i))$, where $f_a$ is an action prediction network with three fully-connected layers.

### 4.2 Explore by Maximizing State Entropy

Existing algorithms only focus on the global exploration policy, which will face the curse of dimensionality. Instead, we propose to use individual exploration and each agent can maximize its own exploration space, which is more efficient. SMMAE learns $n$ independent exploration policies $\mu_{\exp} = \{\mu_{\exp,1}, \ldots, \mu_{\exp,n}\}$ for agents. Inspired by SMM [18], we use state marginal matching to approximate a local state and match the visited observation distribution $\rho_{\pi_i}(o^i)$ with a target distribution $\rho^*(o^i)$, where

$$\rho_{\pi_i}(o^i) = \mathbb{E}_{a_i \sim \pi_i(a_i | o^i), o_{-i} \sim O(P(s_{t+1} | s_t, a_t))} \left[ \frac{1}{T} \sum_{t=1}^{T} 1(o_{-i} = o^i) \right],$$

(6)

and $\rho^*(o^i)$ is obtained using prior information. As there is not any prior information in our experiments, the target distribution of the exploration policy is uniform. To match the two distributions, for each policy we aim to optimize the following objective:

$$\min_{\mu_{\exp}} \mathbb{D}_{KL}(\rho_{\mu_{\exp}}(o^i) || \rho^*(o^i)) = \max_{\mu_{\exp}} \mathbb{E}_{o^i \sim \mu_{\exp}} \left[ \log \rho^*(o^i) + H_{\mu_{\exp}}(o^i) \right].$$

(7)

As $\rho^*(o^i)$ is uniform, to optimize the objective, each exploration policy only needs to maximize the last term $H_{\mu_{\exp}}(o^i)$ and uses
We use the buffer $D_{coop}$ for multi-agent Q-learning and an extra buffer $D_{exp}$ for exploration policy learning to record the tuple $\{(s_i, a_i, s_{i+1}, a'_{i+1}, r'_{i+1}) \mid i = 1, \cdots, n\}$, where the trajectories in $D_{exp}$ are the same as that in $D_{coop}$. The TD loss for all exploration policies $\mu_{exp}$ is:

$$L_{exp} = \frac{1}{n} \sum_{i=1}^{n} (Q_{\theta_{exp}}(s_{i+1}, a_i; \theta_{exp}) - Q_{\pi_{exp}}(s_i, a'_i; \theta_{exp}))^2,$$

(10)

where $\mu_{exp}$ is the exploration policy of agent $i$, $\theta_{exp}$ are the parameters of the target network and will be updated by $\theta_{exp}$ periodically. All the exploration policy $\mu_{exp}$ will be trained after the multi-agent Q-learning policy $\pi_{coop}$ is trained.

The whole algorithm is summarized by Algorithm 1. We induce the adaptive exploration probability based on the uncertainty of the multi-agent system, and then adopt a new individual exploration policy. We first describe the adaptive way to control the exploration probability, which uses the correlation between action and observation of the agents as the criterion (lines 2 and 4-11). Then we introduce the individual exploration policy, which is optimized by state marginal matching [18] (lines 14-15 and 18-19).

5 EXPERIMENTS

We conduct experiments in this section to validate the efficiency of SMMAE, and we benchmark it on QMIX [34] for its widely proved coordination ability in MARL. Implementation details of SMMAE are shown in Appendix B. We first select a training process of SMMAE and study the effects of adaptive exploration probability on the observation areas of the agents during this training process. Then, we choose the benchmark SMAC [35] as the testbed to study the efficient exploration of SMMAE in complex MARL tasks. We compare SMMAE with several methods, including EMC [53] that focuses on multi-agent exploration and some other baselines [21, 34, 36, 43, 44]. After that, we conduct ablation experiments to elaborate the effectiveness of each module. We also show the efficient exploration ability of SMMAE in a visual way. Finally, we show SMMAE is a general framework for MARL algorithms, and it can be applied for other environments like Level Based Foraging (LBF) [7]. More experimental results can be found in the appendix part.

5.1 Study on Adaptive Exploration

In this section, we analyze the effectiveness of adaptive exploration probability. To study the impact of adaptive exploration probability on one agent and the whole team, we experiment on the super hard map $6h_{vs.8z}$ in SMAC, and a training process is shown in Figure 3. The left figure of Figure 3 shows the adaptive exploration probability of one agent and the team test win rate changing over time. We find that the exploration probability of this agent shows a fluctuating decrease (the blue curve) as the training proceeds, which

\footnote{The experiments are based on PyMARL framework.}

\footnote{The SMAC version in our experiments is SC2 6.2.69232.}
is adaptive compared with the vanilla $\epsilon$-greedy. At the beginning of training, the agent explores with a high probability most of the time, while in the middle of training, the exploration probability of the agent varies continuously between high and low values. At the end of the training, the exploration probability tends to converge and gradually stabilizes at a low value.

To analyse the effect of exploration probability on observation areas, we select 4 timesteps, 240k, 510k, 960k, and 2M, where the exploration probability at 240k and 510k timesteps are the two peaks, while the exploration probability at 510k and 2M timesteps are the two valleys. All observations of this agent in the 10 episodes near these 4 timesteps are sampled (four shadow areas in the left figure of Figure 3), and 100 points were randomly selected from them. The right figure in Figure 3 shows the visualization of these points after we apply t-SNE [39] for dimensionality reduction. We find that for the two timesteps with higher exploration probability, there are more scattered clusters (240 and 960k, red and orange for each), while for the two timesteps with lower exploration probability, there are fewer clusters (510k and 2M, green and violet).

In addition, we can find some correlation between the brown test win rate curve and the blue exploration probability curve in the left figure of Figure 3. The red shadow area in the figure is in the early stages of training, where the test win rate remains 0% and the agent focuses on exploring and experiencing more state areas. The green shadow area in the figure has a low probability of exploration. At these timesteps, the correlation between the agents is low, so the agents are more focused on learning how to cooperate with other agents than exploring. Then, some improvement in the brown test win curve can be noticed in this region. The orange shadow area in the figure is a peak of exploration. At this time, the correlation between the agents is already high, so the agent learns to explore more areas, thus jumping out of the local optimum. Therefore, there is a significant drop in the test win rate near this area. In the violet region, the probability of exploration is close to convergence and the agent focuses on better cooperation. The test win rate reaches the peak of the entire training process, validating the effectiveness of our design of adaptive exploring probability.

5.2 Performance on SMAC Super Hard Maps

In order to test the effectiveness of SMMAE, we use the benchmark SMAC [35] as the testbed. We compare SMMAE with some baselines, where VDN [36] and QMIX [34] are popular baselines, RODE [44] and QPLEX [43] are the state-of-the-art baselines, and...
MAVEN [21] and EMC [53] are the latest methods on MARL exploration. For evaluation, we carry out each experiment with 5 random seeds, and the results are shown with a 95% confidence interval.

Figure 4 shows the training curves of SMMAE and other methods on several super hard tasks of SMAC. It can be seen that SMMAE achieves competitive results on tasks such as 6h vs 8z and corridor that require sufficient individual exploration and team cooperation. It shows that adaptive exploration can also achieve the results of other methods using complex modules. This also illustrates the effectiveness of SMMAE because it performs individual exploration and task-related exploration (Section 5.5 gives a visual explanation).

The final test win rate of SMMAE and that of baseline EMC on corridor are almost the same, but it should be noted that EMC needs to use 15GB of GPU when training corridor, while our SMMAE only needs about 4GB.

5.3 Ablation Study

As our method includes multiple modules, we compare SMMAE with three ablated methods for ablation study:

- SMMAE w/o \(\mu_{\text{exp}}\): SMMAE using random exploration instead of the exploration policy \(\mu_{\text{exp}}\).
- SMMAE w/o adaptive \(\alpha\): SMMAE using fixed ending \(\alpha\) instead of the adaptive \(\alpha\).
- QMIX: the QMIX baseline [34].

Figure 5 shows the average test win rate of these four approaches on the 6h vs 8z scenario of SMAC. Compared with the original QMIX, SMMAE w/o adaptive \(\alpha\) is slightly better, while SMMAE w/o \(\mu_{\text{exp}}\) has greatly improves performance. It illustrates the effectiveness of the two modules. Among these four methods, SMMAE achieves the best results, which indicates the effectiveness of using two modules simultaneously. However, it is also observed that the variance of SMMAE becomes larger compared to SMMAE w/o \(\mu_{\text{exp}}\), which indicates that increasing the exploration capability of the individual exploration policy may introduce some instability to the training process. Compared with SMMAE w/o \(\mu_{\text{exp}}\), SMMAE has similar convergence win rate. Therefore, We conducted experiments on two additional maps 2c vs 64zg and corridor in SMAC to further investigate the effect of the exploration policies \(\mu_{\text{exp}}\) in SMMAE (Figure 7). All of these results demonstrate that although random exploration can sometimes perform well, \(\mu_{\text{exp}}\) can explore more efficiently and speed up training.

To qualitatively analyze the exploration ability of the exploration policy, in the ablation experiment, we sample the episodes experienced by SMMAE and SMMAE w/o \(\mu_{\text{exp}}\). Then we sample global state points randomly from them, and visualize the points after dimensionality reduction using t-SNE [39]. Figure 6 shows the results. It can be found that SMMAE has more dispersed clusters in the two-dimensional t-SNE embedding space (Figure 6(a)), and SMMAE has more uniform density and wider range in one-dimensional t-SNE embedding space (Figure 6(b)). It shows that SMMAE, which uses the exploration policy \(\mu_{\text{exp}}\) that maximizes the state entropy of local visited observations, has a stronger exploration ability. Then we try to express the visualization results quantitatively. Because the state space is continuous, we retain two decimal places for each dimension of each global state and calculate an approximate entropy of the visited state based on state counting. The entropy of SMMAE is 15.238, and the entropy of SMMAE w/o \(\mu_{\text{exp}}\) is 14.982. The quantitative results are consistent with the visualization results (Figure 6).

5.4 Comparison with Finetuned-QMIX

Hu et al. [14] finetune QMIX [34] by adding a variety of implementation tricks and achieve good performance on SMAC. To show the necessity of the designs in SMMAE, we compare SMMAE with it, denoted as Hu-QMIX, on the four super hard maps in SMAC. The results show that our SMMAE, which is based on the vanilla QMIX outperforms the finetuned QMIX Hu-QMIX on all the maps (Figure 9). Compared with Hu-QMIX, SMMAE is able to achieve higher test win rate faster and converge to higher win rate on all maps. It demonstrates the effectiveness of SMMAE.

5.5 Efficient Exploration in Task-Related Space

In this section, we qualitatively compare the exploration efficiency of SMMAE with that of baseline EMC on 6h vs 8z by visualization. Figure 8 shows the visual results. The left picture and the right figure are the coordination patterns of the two algorithms during test time. Same as the operation in Figure 6(a), we sample the global state points from their training processes and use t-SNE [39] to visualize the points after dimensionality reduction (middle figure of Figure 8). The middle figure of Figure 8 shows that SMMAE has a similar number of clusters as that of EMC, which means SMMAE and EMC have similar sizes of exploration areas. Same as the calculation method of visiting state entropy in Section 5.3, the entropy of SMMAE is 13.574, and the entropy of EMC is 13.428. The replays of SMMAE during test time show that the agents have learned how to cooperate in different ways to attack the enemies, where the agents have learned how to surround the enemies and how to increase the distance to the enemies to take advantage of range (left figure of Figure 8). However, the replays of EMC during test time show that the agents only have learned how to escape (right figure of Figure 8). The different results demonstrate...
that although SMMAE and EMC have similar sizes of exploration areas, SMMAE can explore more efficiently because the space it explores is more task-related. In fact, our method itself does not explicitly aim to explore the task-related space. In SMMAE, each agent uses its own exploration policy to maximize the entropy of the states it visits, so the method can more efficiently cover a wider exploration space, thus covering task-related states.

5.6 Performance on Level Based Foraging and Application on VDN

Besides SMAC, we also use Level Based Foraging (LBF) [7] to test the ability of SMMAE (Figure 10). The results show that SMMAE can outperform QMIX on the simple task (Figure 10(a)). Because VDN performs better than QMIX on LBF using PyMARL, we also apply SMMAE to another baseline VDN, denoted as SMMAE-VDN, to test the ability of SMMAE. The results show that SMMAE-VDN can also outperform VDN on the task in LBF (Figure 10(b)). It demonstrates that SMMAE is an effective general framework, which can be applied on other value-based MARL methods, and the faster convergence speed means the baselines using SMMAE can explore more effectively.

6 CONCLUSION

With appropriate hyper-parameters, we empirically find that vanilla MARL algorithms using self-exploration can also achieve competitive performance. Consequently, in this paper we propose SMMAE, a novel algorithm that adaptively adjusts the individual exploration probability according to the uncertainty of the multi-agent system at different time steps. SMMAE focuses on each agent’s individual exploration ability by learning an individual exploration policy, which is optimized by target state distribution matching. We study SMMAE on a variety of tasks in the SMAC benchmark, and empirically reveal that SMMAE can explore more efficiently on task-related states and generate better cooperation policies. We take a step towards achieving a trade-off between individual exploration and team cooperation, and we think it is promising for SMMAE to solve complex tasks in multi-agent systems combined with other methods like communication in future work. It should be pointed out that directly matching the explored states to a uniform distribution may cause unexpected exploration, and the adaptive exploration probability is necessary for stabilizing training. It deserves further research on the target distribution of the exploration policy and the adaptive exploration probability.
REFERENCES


[17] Amir Abdi, Prafulla Dhariwal, Sandeep Raja, Greg Sed是个初步的多任务模型, 我们会继续


