Sample-Efficient Multiagent Reinforcement Learning with Reset Replay

Yaodong Yang [1] Guangyong Chen [2 3] Jianye Hao [4 5] Pheng Ann Heng [1 6]

Abstract

The popularity of multiagent reinforcement learning (MARL) is growing rapidly with the demand for real-world tasks that require swarm intelligence. However, a noticeable drawback of MARL is its low sample efficiency, which leads to a huge amount of interactions with the environment. Surprisingly, few MARL works focus on this practical problem especially in the parallel environment setting, which greatly hampers the application of MARL into the real world. In response to this gap, in this paper, we propose Multiagent Reinforcement Learning with Reset Replay (MARR) to greatly improve the sample efficiency of MARL by enabling MARL training at a high replay ratio in the parallel environment setting for the first time. To achieve this, first, a reset strategy is introduced for maintaining the network plasticity to ensure that MARL continually learns with a high replay ratio. Second, MARR incorporates a data augmentation technique to boost the sample efficiency further. Extensive experiments in SMAC and MPE show that MARR significantly improves the performance of various MARL approaches with much fewer environment interactions.

1. Introduction

In recent years, multiagent reinforcement learning (MARL) has achieved significant progress and success in a lot of multiagent fields that require swarm intelligence for complicated decision-making tasks such as real-time strategy games (Berner et al., 2019; Vinyals et al., 2019), distributed energy management (Novati et al., 2021), and urban traffic control (Wu et al., 2020). When applying MARL to real-world tasks, a fundamental drawback is the low sample efficiency that MARL needs a huge amount of interactions with the environment to learn satisfactory policies of a group of agents through trial and error. At the same time, as the interaction with the environment is time-consuming because of the highly complicated multiagent dynamics, the parallel environment setting is usually enabled to accelerate the sample collection. In this practical parallel setting where environment interactions are computationally expensive, the sample efficiency of MARL is more crucial for successfully solving tasks as the interaction budget is limited.

However, although sample-efficient MARL algorithms are highly attractive when applying MARL in realistic tasks, there are few works studying on this topic. Most of these previous works utilize the permutation prior in the multiagent system to achieve sample efficiency. For example, Ye et al. (2022) generate extra data by performing permutation transform for homogeneous agents with the consideration of the permutation invariance prior. Besides, Yu et al. (2023) take advantage of the global symmetry in the multiagent systems, where rotating the global state results in a permutation of the optimal joint policy, to augment data. More recently, Hao et al. (2023) exploit both the permutation invariance and permutation equivariance inductive biases in multiagent scenarios into the entity-wise network design to boost the learning of MARL algorithms. Different from these permutation-based works, in this paper, we achieve the MARL sample efficiency from a new perspective by enabling MARL training with a high replay ratio, especially for the realistic parallel environment setting.

As mentioned above, none of the previous works make efforts to enable the updating of MARL algorithms at a high replay ratio that repeatedly trains networks many times with collected experiences per environment interaction (D'Oro et al., 2023), thus limiting the sample efficiency to a higher degree. Additionally, most of these works for MARL sample efficiency are not specially designed for the more practical parallel environment setting. In response to these gaps, in this paper, we propose the Multiagent Reinforcement Learning with Reset Replay (MARR) algorithm to greatly improve the sample efficiency of MARL algorithms in the parallel environment setting. To achieve this, MARR consists of three essential elements: the high-replay-ratio training setting, the Shrink & Perturb reset strategy, and the random amplitude scale data augmentation technique.

[1]Department of CSE, CUHK [2]Zhejiang Lab [3]Shenzhen Institutes of Advanced Technology, CAS [4]Tianjin University [5]Noah's Ark Lab, Huawei [6]Institute of Medical Intelligence and XR, CUHK. Correspondence to: Guangyong Chen <gychen@zhejianglab.com>, Jianye Hao <jianye.hao@tju.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

First, one main difference between parallel sampling and series sampling is that the sampled experiences of parallel sampling in the replay buffer are more diverse. This diversity brings a chance to improve the sample efficiency of MARL algorithms by increasing the data utilization of these diverse samples. Motivated by this, we set to train MARL algorithms at a high replay ratio so that the updating of agent networks is performed many times per environment interaction. Second, learning at a high replay ratio would make agents incur a risk of overfitting to earlier experiences, negatively affecting the rest of the learning process (Nikishin et al., 2022). To overcome this overfitting phenomenon (Nikishin et al., 2022; Lyle et al., 2023), we introduce a Shrink & Perturb strategy (Ash & Adams, 2020) to reset each agent's network parameters periodically to maintain the network plasticity for continually learning. Third, as we update the networks of agents many times per environment interaction, we integrate the random amplitude scale data augmentation technique (Laskin et al., 2020) in every sampled batch of experiences to further increase the data efficiency. The novel combination of these synergistic and efficient elements for the first time enables MARL training at a higher replay ratio by an order of magnitude than previous cases in the parallel environment setting. Additionally, MARR is general enough and easy to plug into the mainstream off-policy MARL algorithms with only slight modification. Experiments on both the StarCraft Multi-Agent Challenge and Multiagent Particle Environment demonstrate that MARR significantly improves the performance of mainstream off-policy MARL approaches with much fewer environment interactions.

2. Background

2.1. Markov Games

We use the Markov games as the basic setting, which are an extended multiagent version of Markov Decision Processes (Littman, 1994). Markov games are described by a state transition function, $T : S \times A_1 \times ... \times A_N \rightarrow P(S)$, which defines the probability distribution over all possible next states, $P(S)$, given the current global state S and the action A_i produced by the i-th agent. The reward is usually given based on the global state and actions of all agents $R_i : S \times A_1 \times ... \times A_N \rightarrow \mathbb{R}$. If all agents receive the same rewards, i.e. $R_1 = ... = R_N$, then Markov games would become fully cooperative (Matignon et al., 2012): a best-interest action of one agent is also a best-interest action of other agents. Meanwhile, Markov games can be partially observable, and then each agent i receives a local observation $o_i : \mathcal{O}(S, i) \rightarrow O_i$ in this setting. Accordingly, each agent learns a policy $\pi_i : O_i \rightarrow P(A_i)$, which maps each agent's own local observation to a probability distribution over its action set, to maximize this agent's expected discounted cumulative returns, $J_i(\pi_i) = \mathbb{E}_{a_1 \sim \pi_1, ..., a_N \sim \pi_N, s \sim T}[\sum_{t=0}^{\infty} \gamma^t r_i(s_t, a_{1,t}, ..., a_{N,t})]$, where γ is the discounted factor and is in the range of $[0, 1)$.

2.2. Replay Ratio and Sample Efficiency

Replay ratio, also known as the update-to-data ratio, refers to the number of updates of an agent's parameters for each environment interaction (Chen et al., 2021; D'Oro et al., 2023). As each interaction with the environment comes at a cost, it is desired to perform more updates with the existing experiences before interacting with the environment again. Therefore, increasing the replay ratio is an appealing strategy for improving the sample efficiency of deep reinforcement learning algorithms (Chen et al., 2021; D'Oro et al., 2023; Schwarzer et al., 2023). Although researchers in the field of single-agent reinforcement learning have started to focus on this topic, there are seldom related works in the MARL domain trying to improve the replay ratio.

To better quantify the sample efficiency, Ye et al. (2022) define the expectation of the number of times each data experience collected in the replay buffer to be sampled to train the model as

$$\mathbb{E}[N_{sampled}] = \frac{N_{RR} \cdot N_B}{V \cdot T_U}, \tag{1}$$

where N_{RR} is the number of replay ratio that the updating is performed N_{RR} times when interacting with the environment once. N_B is the batch size that there are N_B data experiences in a sampled batch for updating. The data acquisition speed V is the number of transition data experiences being collected at each time step of environment interaction. And T_U is the update interval that the updating is conducted every T_U time steps. Typically, works focusing on training reinforcement learning agents at a high replay ratio (Chen et al., 2021) aim to improve N_{RR} while keeping N_B, V, and T_U unchanged to reach a higher sample efficiency indicated by $\mathbb{E}[N_{sampled}]$. This behaviour is also termed as the replay ratio scaling (D'Oro et al., 2023). However, simply increasing the replay ratio N_{RR} will exacerbate the overfitting phenomenon (Nikishin et al., 2022; Lyle et al., 2023; Sokar et al., 2023; Abbas et al., 2023) to earlier experiences when training, resulting in losing the network plasticity to learn good policies in the rest of the learning process. Next, we introduce this network plasticity loss problem in detail, which has recently been studied in the field of single-agent deep reinforcement learning while receiving comparatively little attention in the MARL domain until now.

2.3. Plasticity Loss in Reinforcement Learning

The study of plasticity has concerned neuroscience for several decades (Abbott & Nelson, 2000), but has only recently emerged as a topic of interest in deep learning (Ash & Adams, 2020) and deep reinforcement learning (Nikishin et al., 2022). Lyle et al. (2023) define plasticity $\mathcal{P}$ to be the difference between the baseline b and the expectation of the final loss obtained by an optimization process after starting from an initial parameter value θ_t and optimizing a sampled loss function l in $\mathcal{L}$.

$$\mathcal{P}(\theta_t) = b - \mathbb{E}_{l\sim\mathcal{L}}[l(\theta_t^*)], \text{where } \theta_t^* = \mathcal{OPT}(\theta_t, l). \quad (2)$$

The optimization algorithm $\mathcal{OPT} : (\theta_t, l) \rightarrow \theta_t^*$ takes initial parameters θ_t and some objective function l and outputs a new set of parameters θ_t^* after the optimization process. Then the plasticity loss along a K-time training trajectory $(\theta_t)_{t=0}^K$ is defined as the difference $\mathcal{P}(\theta_{t=K}) - \mathcal{P}(\theta_{t=0})$ and is independent of the baseline b. The plasticity loss formally defines the decreasing fitting ability of network parameters during the learning process and this phenomenon is repeatedly observed in the experiments across several studies in the field of single-agent reinforcement learning (Nikishin et al., 2023; Sokar et al., 2023; Abbas et al., 2023).

The recently identified tendency of neural networks to lose their network plasticity to learn and generalize from new information during training, against which most reinforcement learning methods deploy no countermeasures, has been the main roadblock in achieving better sample efficiency through replay ratio scaling (D'Oro et al., 2023). This overfitting phenomenon in deep reinforcement learning is also called primary bias (Nikishin et al., 2022), dormant neuron (Sokar et al., 2023), and more frequent plasticity loss (Lyle et al., 2023; Nikishin et al., 2023; Abbas et al., 2023; Kumar et al., 2023). The broad experimental evidence in these works shows that, with the learning progress continuing, the fitting ability of the agent network to newly coming experiences decreases especially when the replay ratio is high. Although there emerges a lot of works in single-agent reinforcement learning to address the plasticity loss, especially at a high replay ratio (D'Oro et al., 2023; Schwarzer et al., 2023), no work yet studies the plasticity loss phenomenon in MARL and we are the first work to address this problem towards successfully training MARL at a high replay ratio.

3. Method

The experience diversity in the parallel setting brings the benefit of improving MARL sample efficiency by increasing the data utilization through repeatedly updating with these diverse samples. Inspired by this, we set MARL algorithms to learn at a high replay ratio in parallel environments. More specifically, the updating of agent network parameters is performed many times by setting a large N_{RR} while interacting with the environment once given the same update interval T_U, data acquisition speed V, and batch size N_B. Surprisingly, despite this idea seeming to be straightforward, there are seldom MARL works reporting a successful attempt. A lot of practices in both single-agent reinforcement learning (D'Oro et al., 2023; Abbas et al., 2023) and MARL demonstrate that, if the replay ratio is increased to a large number, the learning process becomes unstable and significantly impedes the quality of the learned agent policies. In this paper, we propose an algorithm called Multiagent Reinforcement Learning with Reset Replay (MARR) to address this problem to enable successful MARL training in the parallel environment setting with a high replay ratio.

The description of the proposed MARR algorithm is shown in Algorithm 1. MARR follows the general workflow of standard off-policy MARL algorithms but employs two additional operations. The main additional operation of MARR is in Line 19, given the network reset interval T_R time steps, MARR performs Shrink & Perturb to inject plasticity into both the centralized critic network and each agent's policy or Q-value network to recover learning ability of these networks. The second additional operation is in Line 10. After sampling a transition batch, MARR employs the data augmentation on the sampled transition batch B to further increase the diversity of samples. This data augmentation operation takes full advantage of the N_{RR} updates to force networks to learn consistencies in the input representations.

Next, we introduce the details of the above techniques integrated into MARR. The first is the Shrink & Perturb strategy. Learning at a high replay ratio would make the agents incur a risk of overfitting to earlier experiences, negatively affecting the rest of the learning process (Nikishin et al., 2022). To overcome this overfitting phenomenon, we introduce the Shrink & Perturb strategy (Ash & Adams, 2020) to reset the network parameters of agents periodically to maintain the network plasticity. This reset strategy was originally proposed to warm-start neural network training to incorporate newly arriving data without sacrificing generalization (Ash & Adams, 2020) and has been recently employed in the field of single-agent reinforcement learning to prevent overfitting under a high replay ratio setting (D'Oro et al., 2023; Schwarzer et al., 2023; Lyle et al., 2023).

The formulation of the Shrink & Perturb strategy in MARR for the multiagent case is defined as

$$\theta_t^i \leftarrow \alpha\theta_t^i + (1-\alpha)\theta_0^i, \text{for } i = 1, 2, ..., N, \quad (3)$$

and

$$\phi_t \leftarrow \alpha\phi_t + (1-\alpha)\phi_0, \quad (4)$$

where θ_0^i is agent i's initial policy or Q-value network parameters and ϕ_0 is the initial centralized critic network parameters. θ_t^i and ϕ_t are the current agent network parameters and centralized critic network parameters respectively. The interpolation factor α decides how much the current network parameters are kept. The motivation behind Shrink & Perturb is that, while current network parameters are trained to fit the transition experiences in the replay buffer well, it gradually loses its learning ability for newly coming experiences especially when the replay ratio is high. At the same

Algorithm 1 Multiagent Reinforcement Learning with Reset Replay (MARR)

```
1: Initialize each agent's policy or Q-value network parameters $\theta^1, \theta^2, ..., \theta^N$, the centralized critic network parameters $\phi$, and an empty replay buffer $D$. Set target network parameters $\bar{\theta}^i \leftarrow \theta^i$ for $i = 1, 2, ..., N$, and $\bar{\phi} \leftarrow \phi$. Set network update interval $T_U$, target network update interval $T_C$, and network reset interval $T_R$.
2: for each time step $t$ do
3:   for each parallel environment $e$ do
4:     Each agent $i$ takes action $a_{i,t,e} \sim \pi_{\theta^i}(\cdot|o_{i,t,e})$. Step into state $s_{t+1,e}$. Receive reward $r_{t,e}$ and observe $o_{i,t+1,e}$.
5:     Add transition data to the replay buffer: $D \leftarrow D \cup \{(s_{t,e}, \mathbf{o}_{t,e}, \mathbf{a}_{t,e}, r_{t,e}, s_{t+1,e}, \mathbf{o}_{t+1,e})\}$.
6:   end for
7:   if $t \bmod T_U = 0$ then
8:     for each update time $n_{RR}$ from 1 to $N_{RR}$ do
9:       Sample a mini-batch $B = \{(s, \mathbf{o}, \mathbf{a}_t, r, s', \mathbf{o}')\}$ from $D$.
10:      Perform the random amplitude scale data augmentation as Equation (5) and (6) on sampled transition batch $B$.
11:      Update the parameters of centralized critic $\phi$ with the augmented transition batch $B$.
12:      Update each agent's policy or Q-value network parameters $\theta^1, \theta^2, ..., \theta^N$ with the augmented transition batch $B$.
13:    end for
14:  end if
15:  for every $T_C$ time steps do
16:    Update the target network of centralized critic $\bar{\phi} \leftarrow \phi$, and target networks of agents $\bar{\theta}^1 \leftarrow \theta^1$, ..., $\bar{\theta}^N \leftarrow \theta^N$.
17:  end for
18:  for every $T_R$ time steps do
19:    Perform Shrink & Perturb as Equation (3) and (4) on both the critic $\phi$ and agent networks $\theta^1, \theta^2, ..., \theta^N$.
20:  end for
21: end for
```

time, the initial network parameters are believed to own the network plasticity for learning but with little experience or knowledge. Therefore, we need an approach to combine the two types of status of network parameters. The Shrink & Perturb strategy allows interpolation between current network parameters and an initialized parameter vector on each reset, which could be regarded as injecting the plasticity of the initial network into the current trained network.

With Shrink & Perturb, we are able to update the policy or value networks many times (N_{RR}) after an update interval of a number of time steps (T_U) for environment interactions. In this high-replay-ratio setting, for every T_U time steps, there are $N_{RR} \cdot N_B$ transition experiences sampled with replacement from the replay buffer for updating. Therefore, there is a high probability of updating with the same transition experience in the same or adjacent updating time steps. To avoid duplicate updates, we introduce the random amplitude scale (Laskin et al., 2020) here to increase the diversity of sampled experiences. Specifically, given a sampled transition experience $(s, \mathbf{o}, \mathbf{a}_t, r, s', \mathbf{o}')$, the formulation of the random amplitude scale in MARR for the multiagent case is

$$o_i \leftarrow o_i * z, o'_i \leftarrow o'_i * z, \text{for } i = 1, 2, ..., N, \tag{5}$$

and

$$s \leftarrow s * z, s' \leftarrow s' * z, \tag{6}$$

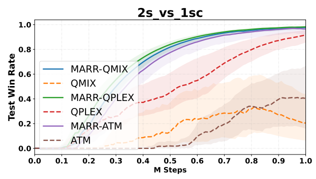
where $z \sim U(a, b)$ is a random value z sampled from an uniform distribution over a range $[a, b]$. Note that the random amplitude scale augmentation is applied randomly across the batch experiences but consistent across time, i.e., the same randomization to current and next input observations as well as states. The intuition behind it is to randomize the amplitude of input states while maintaining their intrinsic information (e.g., sign of inputs) (Laskin et al., 2020). This data augmentation operation facilitates agents to learn intrinsic consistencies in the input representations, especially at a high replay ratio. For more insights about the random amplitude scale, please refer to Appendix G.

The novel combination of the two simple yet efficient extended techniques in the multiagent scenarios for the first time enables MARL training in the parallel environment setting with a high replay ratio. Moreover, MARR is general and easy to plug into the mainstream off-policy MARL algorithms with only slight modification as shown in Algorithm 1. Next, we conduct experiments on various tasks of SMAC and MPE to validate the effectiveness of MARR.

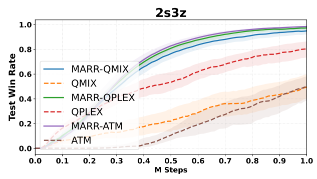
4. Experiment

In this section, we validate MARR[1] on both the StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al., 2019) with discrete action space and the Multiagent Particle Environment (MPE) (Lowe et al., 2017) with continuous action space. We show that the replay ratio could be scaled up to

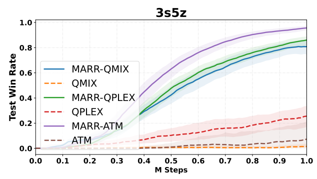
[1]Code is available at GitHub.



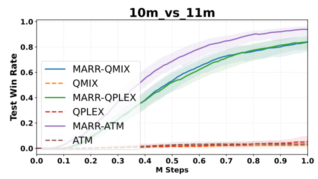
(a) 2c_vs_1sc



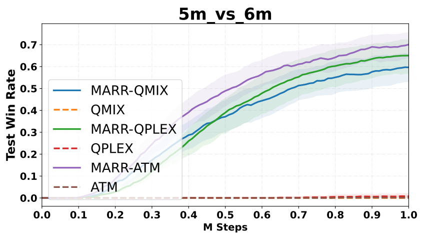
(b) 2s3z



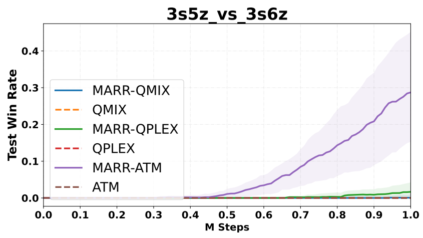
(c) 3s5z



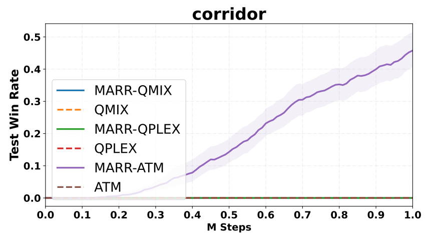
(d) 10m_vs_11m



(e) 5m_vs_6m



(f) 3s5z_vs_3s6z



(g) corridor

Figure 1. Results of the base MARL algorithms and their MARR-based variants on different SMAC scenarios.

even 50 in SMAC and 25 in MPE. For all the tasks, we set α at 0.8 and the reset interval T_R at 2000 for Shrink & Perturb, and set a at 0.8 and b at 1.2 for the random amplitude scale.

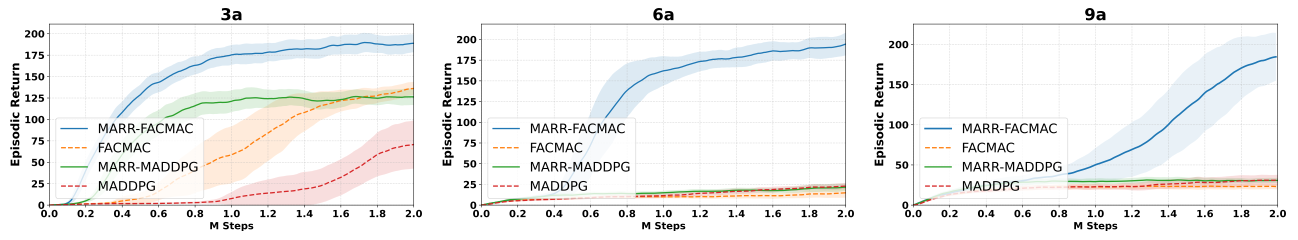
4.1. SMAC

First, we evaluate our method in the StarCraft II decentralized micromanagement tasks and use StarCraft Multi-Agent Challenge environment (Samvelyan et al., 2019) as our testbed, which has become a commonly-used benchmark for evaluating state-of-the-art MARL approaches. At the beginning of each episode, the enemy units are going to attack the allies. We train multiple agents to control allied units respectively to beat the enemy, while a built-in handcrafted AI controls the enemy units. Training and evaluation schedules such as the testing episode number and training hyper-parameters are kept unchanged. The SMAC environment is with discrete action space and the used version of StarCraft II is 4.6.2. We implement MARR based on the pymarl framework (Samvelyan et al., 2019).

We perform the experiments on 7 tasks i.e., 2s_vs_1sc, 2s3z, 3s5z, 10m_vs_11m, 5m_vs_6m, 3s5z_vs_3s6z, and corridor. These tasks include homogeneous and heterogeneous multiagent scenarios, as well as symmetrical and asymmetrical multiagent scenarios for a comprehensive evaluation of MARR. In the 2s_vs_1sc map, there are 2 Stalkers against 1 Spine Crawler. For map 2s3z, both sides have 2 Stalkers and 3 Zealots. For map 3s5z, both sides have 3 Stalkers and 5 Zealots. In the map of 10m_vs_11m, there are 10 allied marines against 11 marine enemies. In the map of 5m_vs_6m, there are 5 allied marines against 6 marine enemies. In the map of 3s5z_vs_3s6z, there are 3 Stalkers and 5 Zealots against 3 Stalkers and 6 Zealots. In the corridor map, 6 allied Zealots are against 24 Zerglings.

Here we plug MARR into QMIX (Rashid et al., 2018) and QPLEX (Wang et al., 2021), two of the most representative off-policy algorithms in SMAC. Additionally, we plug MARR into a more recent MARL algorithm ATM (Yang et al., 2022), which employs a transformer-based working memory mechanism and action binding to achieve superior performance especially on the SMAC's super hard scenarios. We run these algorithms and their MARR-based variants with the official codes and configurations released by their authors. We set the number of parallel environments to 8, which means agents are interacting with 8 parallel environments at the same time and the number of environment interactions is increased by 8 for each time step. For MARR, we set the replay ratio N_{RR} to 50 (updating networks 50 times after one training episode) while this value in the default implementation is 1 (updating once after one training episode). We use the test win rate as the performance metric. Results of these base algorithms and their MARR-based versions in SMAC are shown in Figure 1. The x-axis counts the step number of environment interactions, which is limited to 1 million. The resulting plots include the median performance over 6 independent training runs with different random seeds as well as the shaded 25-75% percentiles.

As we can see, with MARR enabling the MARL training at a high replay ratio of 50, the performance and learning speed are dramatically improved when compared with the base MARL algorithms when the environment interaction budget is fixed (i.e., 1 million steps). In the scenarios of 2s_vs_1sc, 2s3z, 3s5z, 10m_vs_11m, and 5m_vs_6m, MARR successfully boosts all the base algorithms by a large performance margin. In the more difficult scenarios of 3s5z_vs_3s6z and corridor, MARR can also boost ATM to obtain good performance. We also run QMIX, QPLEX, and ATM by 5 million steps for an intuitive display of how much MARR helps



(a) 3 agents vs 1 agent (b) 6 agents vs 2 agents (c) 9 agents vs 3 agents

Figure 2. Results of the base MARL algorithms and their MARR-based variants on different predator-prey tasks in MPE.

accelerate these MARL algorithms in Appendix A.

4.2. MPE

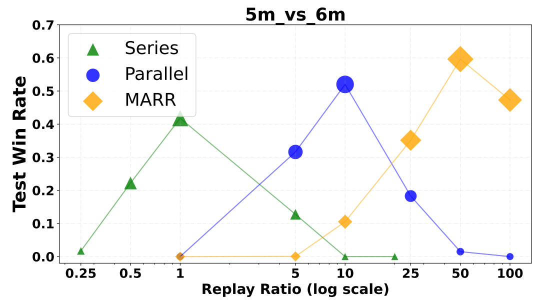
Next, we evaluate MARR in the Multiagent Particle Environment environment (Lowe et al., 2017), which is with continuous action space. We use a set of classical predator-prey tasks (Peng et al., 2021) that several slower cooperating circular agents, each with continuous movement action spaces, must catch the faster circular prey on a randomly generated two-dimensional toroidal plane with two large landmarks blocking the way. If one agent collides with the prey while at least another one is close enough, a team reward of +10 is given. However, if only one agent collides with the prey without any other agent being close enough, a negative team reward of -1 is given. Otherwise, no reward is provided. In this task, each agent can observe the relative positions of the other agents, the relative position and velocity of the preys, and the relative positions of the landmarks. Besides, each agent has an agent view radius, which restricts the agents from receiving information about other entities (including all landmarks, the other agents, and the preys) that are out of range. We perform the experiments on 3 tasks with different agent numbers i.e., 3a where 3 agents catch 1 prey, 6a where 6 agents catch 2 preys, and 9a where 9 agents catch 3 preys. We follow the environmental settings and training configurations the same as Peng et al. (2021).

Here we plug MARR into MADDPG (Lowe et al., 2017) and FACMAC (Peng et al., 2021), two representative off-policy MARL algorithms for continuous multiagent action space. We run these algorithms and their MARR-based variants with the codebase and configurations from Peng et al. (2021) and set the number of parallel environments to 4. For MARR, we set the replay ratio N_{RR} to 25 while the default value is 1. All the results are averaged over 6 independent runs with different random seeds and the resulting plots include the 95% confidence interval. The results of the standard MARL algorithms and their MARR-based versions in MPE are shown in Figure 2. As we can see, MARR greatly improves the sample efficiency of both MADDPG and FACMAC within fixed environment interactions. We also note that, in 6a and 9a, MARR fails to boost MADDPG within the given environment steps. The reason behind this may be that MADDPG is with difficulty learning in the two scenarios and MARR cannot guarantee that it always helps algorithms solve challenging tasks that are hard for the base algorithm itself. We also run MADDPG and FACMAC for 5 million steps and the results are provided in Appendix B.

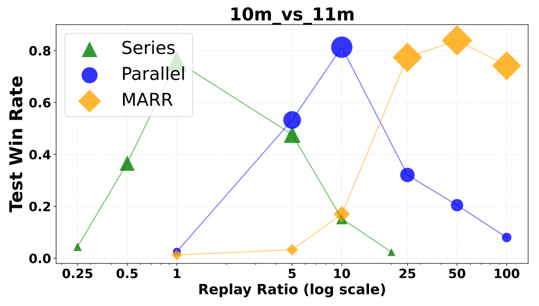
In both SMAC and MPE, the promising performance improvements of MARR over standard MARL algorithms given the same number of environment interactions indicate that MARR is a widely applicable approach to achieve high sample efficiency for off-policy MARL algorithms. Next, we conduct the ablation studies and experimental analysis to validate each essential element ensembled in MARR.

4.3. Parallel Setting versus Series Setting

We first conduct an ablation study to verify the hypothesis that the parallel setting is more suitable for training at a high replay ratio than the series setting. We run QMIX at a series environment setting and a parallel environment setting (8 environments) with different replay ratios. Both settings have the same number of environment interactions of 1 million. The results are shown in Figure 3. ‘Series’ means QMIX in the series environment setting. ‘Parallel’ means QMIX in the parallel environment setting with 8 parallel environments. We could see that, when the replay ratio is too low or too high, the learning performance decreases significantly. Only with a proper replay ratio (e.g., $N_{RR} = 1$ in the series setting and $N_{RR} = 10$ in the parallel setting), the algorithm can achieve relatively good performance. Importantly, we see the parallel setting supports a higher replay ratio than the series setting. For example, when the replay ratio is 10, QMIX achieves the highest test win rate in the parallel setting while failing to learn in the series setting with the same replay ratio. Although the parallel setting supports a high N_{RR}, it still fails to train the algorithm with a fixed number of environment interactions when N_{RR} continually increases to a much larger value (e.g., N_{RR} is 50 or higher). This naturally calls for a technique to support higher replay ratios for sample efficiency and MARR achieves this.

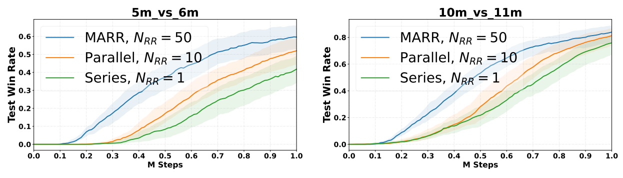


(a) Different replay ratios in series and parallel setting on 5m_vs_6m



(b) Different replay ratios in series and parallel setting on 10m_vs_11m

Figure 3. Comparison of different replay ratios in the series and parallel environment settings with 1 million environment interactions. 'Parallel' means QMIX in the parallel environment setting with 8 parallel environments. 'MARR' means MARR-based QMIX in the parallel environment setting with 8 parallel environments.



(a) 5m_vs_6m (b) 10m_vs_11m

Figure 4. Results of QMIX in the series setting, QMIX in the parallel setting, and MARR-based QMIX in the parallel setting with their optimal replay ratios at indicated in Figure 3 respectively.

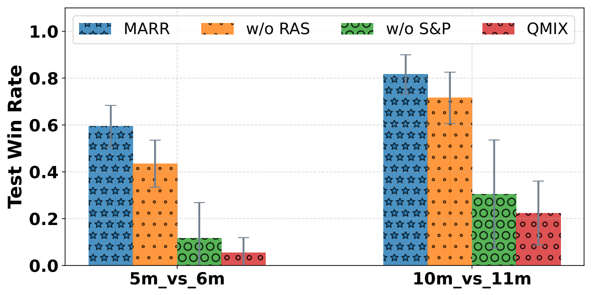


Figure 5. Ablation study for each component of MARR. The 'w/o S&P' is MARR without Shrink & Perturb while the 'w/o RAS' is MARR without random amplitude scale. If MARR is without both Shrink & Perturb and random amplitude scale, it degrades to QMIX. All the methods are running with replay ratio N_{RR} at 50.

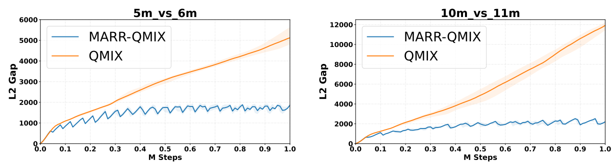
4.4. Experiments of Replay Ratio for MARR

Another interesting ablation study is how MARR performs at different replay ratios. To answer this question, we run MARR-based QMIX in 1 million environment interactions with N_{RR} ranging from 1 to 100. The results are provided in Figure 3. With increased replay ratios, the performance of MARR improves until $N_{RR} = 50$. However, a larger replay ratio such as $N_{RR} = 100$ also hurts the performance of MARR. This may result from overfitting by training with the same transition data in the replay buffer too many times. Moreover, we want to point out that the optimal replay ratio may differ for each scenario although $N_{RR} = 50$ achieves the best performance among several replay ratios on the two tested scenario cases. Meanwhile, we also plot the learning process of QMIX in each setting with optimal replay ratios, which is shown in Figure 4. We clearly see that, throughout the learning process, MARR significantly boosts the performance of QMIX in terms of the same interaction steps by training with N_{RR} at 50.

4.5. Ablation Study for MARR

Then we conduct the ablation study to validate each component of MARR, especially the Shrink & Perturb strategy. We plot the mean test win rate with the 95% confidence interval as the error bar. The results are shown in Figure 5. As we can see, if MARR is without Shrink & Perturb (MARR w/o S&P), the performance degenerates significantly in both tested scenarios. Meanwhile, the random amplitude scale is helpful to further boost the sample efficiency when Shrink & Perturb is employed to enable high-replay-ratio training. However, the random amplitude scale itself contributes slightly when there lacks of Shrink & Perturb. This component ablation study indicates that Shrink & Perturb is the vital technique to enable MARL training at a high replay ratio while the random amplitude scale could further boost the sample efficiency when Shrink & Perturb is employed.

4.6. Experimental Analysis of Network Plasticity



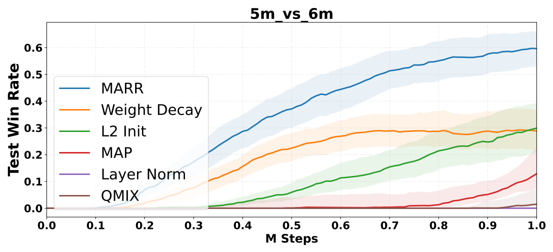
(a) L2 gap on 5m_vs_6m (b) L2 gap on 10m_vs_11m

Figure 6. Measurements of L2 gap between the trained MARR-based QMIX and QMIX networks to their initial networks respectively. The replay ratio N_{RR} is set to 50 for each method.

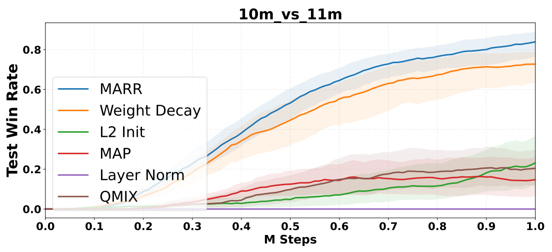
Here we experimentally show how MARR works to maintain the network plasticity to newly coming transition experiences. As we hold the assumption that initial network parameters own the full plasticity to learn from experiences compared with the trained network parameters (Lyle et al., 2023), we use the L2 gap between the trained network parameters and the initial network parameters to measure the network plasticity as inspired by Kumar et al. (2023).

The proxy measurements of network plasticity of MARR and QMIX are shown in Figure 6. MARR maintains the L2 gap to the initial network parameters at a lower level during the training process when compared with QMIX. This indicates that MARR maintains the ability to fit newly coming experiences at a high replay ratio by periodically injecting plasticity through Shrink & Perturb. Meanwhile, the network parameters of QMIX are departing from the initial network parameters fast with training proceeding.

4.7. Comparison with Other Plasticity Techniques



(a) Plasticity techniques on 5m_vs_6m



(b) Plasticity techniques on 10m_vs_11m

Figure 7. Results of different methods for maintaining plasticity in SMAC. The replay ratio for all methods in both scenarios is at 50.

Here we compare MARR with other plasticity techniques. Kumar et al. (2023) propose L2 Init to maintain network plasticity by designing an L2 regularization loss function toward initial parameters in the domain of continual learning. Another natural baseline is weight decay, which regularizes the network parameters to prevent overfitting, especially with a high replay ratio (Schwarzer et al., 2023). We also incorporate a baseline of multiagent permutation (MAP) in the global state (Ye et al., 2022) to shuffle the entities in the agent group and enemy group respectively for data augmentation. Besides, we use layer normalization as a baseline. Figure 7 shows the results of different plasticity techniques and MARR performs best in the tested scenarios.

4.8. Analysis of Running Time

MARR enables parallel MARL training at a high replay ratio to reduce environment interactions, which shifts the running time from environment simulation to network parameter training. This brings benefits to the real-world application of MARL algorithms by reducing the expensive environment interactions to collect transition data. Ideally, the running wall-clock time of MARL could be mainly decomposed as

$$h_{tot} = \frac{T_{tot} * h_{env}}{P_{env}} + \frac{T_{tot} * N_{RR} * h_{upt}}{T_U} + h_{rst}, \quad (7)$$

where T_{tot} is the total step number of environment interactions and h_{env} is the running time per environment interaction. P_{env} is the number of parallel environments and equals 1 in the series environment setting. N_{RR} is the replay ratio, T_U is the updating interval, and h_{upt} is running time per network updating. The last term h_{rst} is the running time for rest operations. As indicated by Equation (7), when training in the parallel environment setting, the first term can be reduced. However, the performance also decreases as shown in Figure 3 where the default $N_{RR} = 1$ in the series setting performs poorly in the parallel setting. MARR is able to compensate for the performance with the same environment interaction budget by increasing N_{RR} in the second term. Although this also leads to an increase in the second term, in realistic scenarios with sophisticated system dynamics to simulate, the first term dominates h_{tot} as $h_{env} \gg h_{upt}$. In this case, MARR could help the MARL algorithms achieve higher performance with less running wall-clock time.

5. Conclusion

In this paper, we propose MARR to enable MARL training with a high replay ratio for the first time. First, we introduce the Shrink & Perturb strategy into MARL to maintain the network plasticity during the high-replay-ratio training process. Second, we integrate the random amplitude scale to further boost performance by exploiting the repeated updates. The novel combination of these efficient and synergistic techniques extended for the multiagent case greatly improves the sample efficiency of the mainstream off-policy MARL algorithms with only slight modification. Experiments in both the classical MPE and the challenging SMAC demonstrate that MARR significantly improves the performance of mainstream off-policy MARL approaches with much fewer environment interactions, which is of great potential for the application of MARL in the real world.

For future work, on the one hand, further understanding and revealing the underlying mechanism for the plasticity loss in MARL is important. On the other hand, extending MARR into offline or continual MARL settings is also promising.

Acknowledgements

This work is supported by the National Key R&D Program of China (2022YFE0200700), a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Number: 14201321), a grant from the Hong Kong Innovation and Technology Fund (Project Number: MHP/092/22), the National Natural Science Foundation of China (Project No. 6237073934, 3234101132), and the Natural Science Foundation of Guangdong Province (2022A1515011579).

Impact Statement

This paper presents work whose goal is to advance the field of MARL. MARL is a powerful paradigm that can model real-world systems, such as autonomous driving, network optimization, and energy distribution. The proposed MARR could accelerate the learning and improve the sample efficiency of existing off-policy MARL algorithms, thus increasing the practicability of MARL in real-world applications. However, when applied to real-world tasks, the learning process of MARL with MARR still needs some explorations which may lead to unsafe situations and additional safeguard procedures may be needed. On the other hand, there still exists the risk of using MARL with MARR to do unethical actions such as performing network attacks.

References

Abbas, Z., Zhao, R., Modayil, J., White, A., and Machado, M. C. Loss of Plasticity in Continual Deep Reinforcement Learning. In *Proceedings of The 2nd Conference on Lifelong Learning Agents*, volume 232, pp. 620–636. PMLR, 2023.

Abbott, L. F. and Nelson, S. B. Synaptic plasticity: taming the beast. *Nature Neuroscience*, 3(S11):1178–1183, 2000.

Ash, J. and Adams, R. P. On Warm-Starting Neural Network Training. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 3884–3894. Curran Associates, Inc., 2020.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., Józefowicz, R., Gray, S., Olsson, C., Pachocki, J., Petrov, M., Pinto, H. P. d. O., Raiman, J., Salimans, T., Schlatter, J., Schneider, J., Sidor, S., Sutskever, I., Tang, J., Wolski, F., and Zhang, S. Dota 2 with Large Scale Deep Reinforcement Learning, 2019. arXiv:1912.06680.

Chen, X., Wang, C., Zhou, Z., and Ross, K. W. Randomized Ensembled Double Q-Learning: Learning Fast Without a Model. In *International Conference on Learning Representations*, 2021.

D'Oro, P., Schwarzer, M., Nikishin, E., Bacon, P.-L., Bellemare, M. G., and Courville, A. Sample-Efficient Reinforcement Learning by Breaking the Replay Ratio Barrier. In *The Eleventh International Conference on Learning Representations*, 2023.

Hao, J., Hao, X., Mao, H., Wang, W., Yang, Y., Li, D., ZHENG, Y., and Wang, Z. Boosting Multiagent Reinforcement Learning via Permutation Invariant and Permutation Equivariant Networks. In *The Eleventh International Conference on Learning Representations*, 2023.

Kumar, S., Marklund, H., and Roy, B. V. Maintaining Plasticity in Continual Learning via Regenerative Regularization, 2023. arXiv:2308.11958.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement Learning with Augmented Data. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 19884–19895. Curran Associates, Inc., 2020.

Littman, M. L. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings*, pp. 157–163. Elsevier, 1994.

Lowe, R., WU, Y., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Lyle, C., Zheng, Z., Nikishin, E., Pires, B. A., Pascanu, R., and Dabney, W. Understanding Plasticity in Neural Networks. In *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 2023.

Matignon, L., Laurent, G. J., and Le Fort-Piat, N. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1):1–31, 2012.

Nikishin, E., Schwarzer, M., D'Oro, P., Bacon, P.-L., and Courville, A. The Primacy Bias in Deep Reinforcement Learning. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 16828–16847. PMLR, 2022.

Nikishin, E., Oh, J., Ostrovski, G., Lyle, C., Pascanu, R., Dabney, W., and Barreto, A. Deep Reinforcement Learning with Plasticity Injection. In *Advances in Neural Information Processing Systems*, volume 36, pp. 37142–37159. Curran Associates, Inc., 2023.

Novati, G., De Laroussilhe, H. L., and Koumoutsakos, P. Automating turbulence modelling by multi-agent reinforcement learning. *Nature Machine Intelligence*, 3(1): 87–96, 2021.

Peng, B., Rashid, T., de Witt, C. S., Kamienny, P.-A., Torr, P., Boehmer, W., and Whiteson, S. FACMAC: Factored multi-agent centralised policy gradients. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021.

Rashid, T., Samvelyan, M., Witt, C. S. d., Farquhar, G., Foerster, J. N., and Whiteson, S. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 4292–4301, 2018.

Samvelyan, M., Rashid, T., Schroeder de Witt, C., Farquhar, G., Nardelli, N., Rudner, T. G. J., Hung, C.-M., Torr, P. H. S., Foerster, J., and Whiteson, S. The StarCraft Multi-Agent Challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2186–2188, 2019.

Schwarzer, M., Obando-Ceron, J., Courville, A., Bellemare, M. G., Agarwal, R., and Castro, P. S. Bigger, Better, Faster: Human-Level Atari with Human-Level Efficiency. In *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 2023.

Sokar, G., Agarwal, R., Castro, P. S., and Evci, U. The Dormant Neuron Phenomenon in Deep Reinforcement Learning. In *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 2023.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., Oh, J., Horgan, D., Kroiss, M., Danihelka, I., Huang, A., Sifre, L., Cai, T., Agapiou, J. P., Jaderberg, M., Vezhnevets, A. S., Leblond, R., Pohlen, T., Dalibard, V., Budden, D., Sulsky, Y., Molloy, J., Paine, T. L., Gulcehre, C., Wang, Z., Pfaff, T., Wu, Y., Ring, R., Yogatama, D., Wünsch, D., McKinney, K., Smith, O., Schaul, T., Lillicrap, T., Kavukcuoglu, K., Hassabis, D., Apps, C., and Silver, D. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354, 2019.

Wang, J., Ren, Z., Liu, T., Yu, Y., and Zhang, C. Qplex: Duplex dueling multi-agent q-learning. In *International Conference on Learning Representations*, 2021.

Wu, T., Zhou, P., Liu, K., Yuan, Y., Wang, X., Huang, H., and Wu, D. O. Multi-Agent Deep Reinforcement Learning for Urban Traffic Light Control in Vehicular Networks. *IEEE Transactions on Vehicular Technology*, 69(8):8243–8256, 2020.

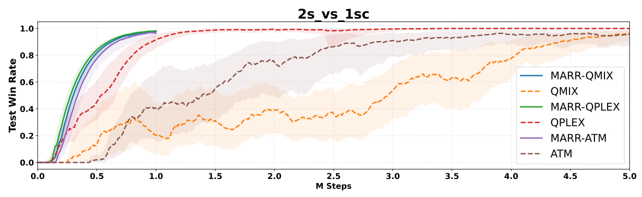
Yang, Y., Chen, G., Wang, W., Hao, X., HAO, J., and Heng, P.-A. Transformer-based Working Memory for Multiagent Reinforcement Learning with Action Parsing. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K. (eds.), *Advances in Neural Information Processing Systems*, 2022.

Ye, Z., Chen, Y., Jiang, X., Song, G., Yang, B., and Fan, S. Improving sample efficiency in Multi-Agent Actor-Critic methods. *Applied Intelligence*, 52(4):3691–3704, 2022.

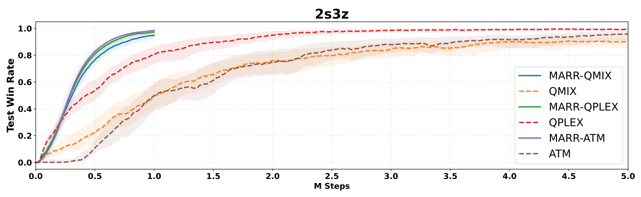
Yu, X., Shi, R., Feng, P., Tian, Y., Luo, J., and Wu, W. ESP: Exploiting Symmetry Prior for Multi-Agent Reinforcement Learning. In *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2023.

A. Additional Experimental Results in SMAC

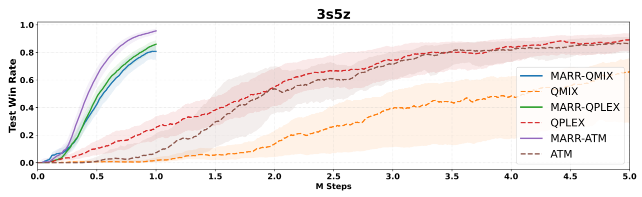
Here we give the performance of QMIX, QPLEX, and ATM with 5 million environment interaction steps to demonstrate how much MARR helps accelerate these MARL algorithms with discrete action space. The resulting plots include the median performance over 6 independent training runs with different random seeds as well as the shaded 25-75% percentiles. Results are shown in Figure 8. As we can see, MARR greatly boosts the performance of all three MARL algorithms with much fewer environment interactions. Even on the super-hard scenarios 3s5z_vs_3s6z and corridor, MARR improves the performance of ATM by a large margin with 1 million environment interaction steps while ATM has test win rates near 0 with even 5 million environment interaction steps. These extending results further validate that MARR achieves high sample efficiency for MARL algorithms by enabling training MARL algorithms at a high replay ratio.



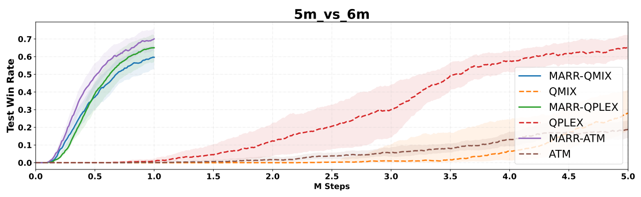
(a) 2c_vs_1sc



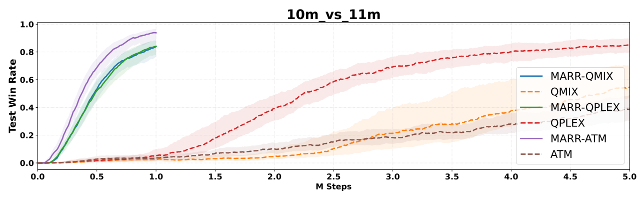
(b) 2s3z



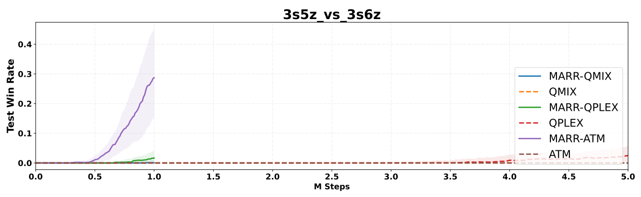
(c) 3s5z



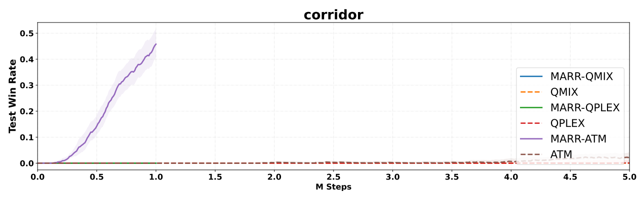
(d) 5m_vs_6m



(e) 10m_vs_11m



(f) 3s5z_vs_3s6z



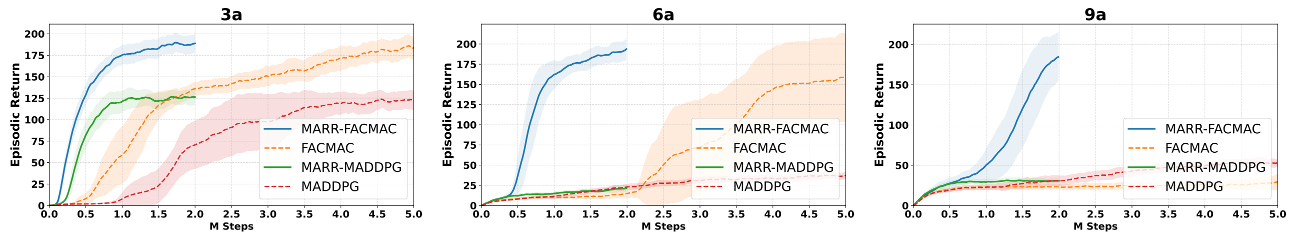
(g) corridor

Figure 8. Results on different SMAC scenarios. The number of environment interaction steps for standard QMIX, QPLEX, and ATM is 5 million. The environment interaction steps for MARR-based QMIX, MARR-based QPLEX, and MARR-based ATM are 1 million.

B. Additional Experimental Results in MPE

Here we also give the performance of MADDPG and FACMAC with 5 million environment interaction steps to demonstrate how much MARR helps accelerate the two off-policy MARL algorithms with continuous action space. The resulting plots include the averaged performance over 6 independent training runs with different random seeds as well as the shaded 95% confidence interval. Results of the two algorithms and their MARR-based variants in MPE are shown in Figure 9.

As we can see, MARR greatly boosts the performance of both the MARL algorithms with much fewer environment interactions. As shown in Figure 9(a), MARR successfully boosts both the MADDPG and FACMAC. The performance of



(a) 3 agents vs 1 agent (b) 6 agents vs 2 agents (c) 9 agents vs 3 agents

Figure 9. Results on different predator-prey tasks in MPE. The environment interaction steps for standard MADDPG and FACMAC are 5 million. The environment interaction steps for MARR-based MADDPG and MARR-based FACMAC are 2 million.

MARR-MADDPG and MARR-FACMAC at the end of 2 million steps is even better than the performance of MADDPG and FACMAC at the end of 5 million steps. Figure 9(b) and Figure 9(c) show that MARR improves the performance of FACMAC but fails to boost MADDPG within 2 million environment interaction steps. The reason may be that MADDPG has difficulty in learning this scenario and MARR-based MADDPG meets the same problem. Nevertheless, MARR shows a strong ability to accelerate the learning of MARL algorithms with much fewer environment interactions.

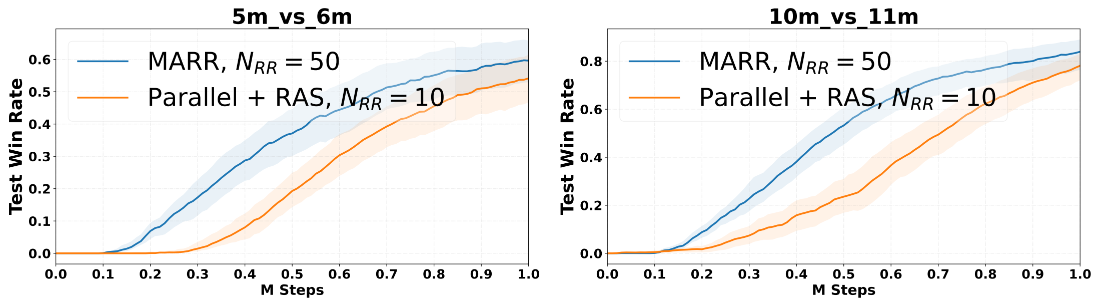
C. Hyperparameters in MARR

For all the tasks in both the SMAC and MPE, we set the interpolation factor α at 0.8 and reset interval T_R at 2000 for the Shrink & Perturb, and set a at 0.8 and b at 1.2 for the random amplitude scale. The interpolation factor for Shrink & Perturb is recommended in the previous single-agent work (D'Oro et al., 2023) and we follow this recommended value. The hyperparameter values for the random amplitude scale are the default configurations in the original paper (Laskin et al., 2020). Therefore, we use these default values and do not exhaustively tune these hyperparameters for MARR.

D. Additional Ablation Studies

D.1. Further Comparison of Parallel QMIX and MARR

From Figure 4, we see QMIX in the parallel environment setting with replay ratio N_{RR} at 10 performs well. Here we conduct an ablation study to see whether the random amplitude scale could further boost and how it compares with MARR. The results are shown in Figure 10. We see that even though QMIX has the optimal replay ratio and the random amplitude scale in the parallel setting, MARR still learns much faster than it in the learning process. This further validates the necessity of the Shrink & Perturb strategy of MARR in the high-replay-ratio MARL training setting.

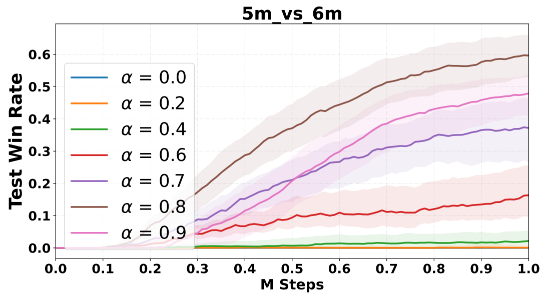


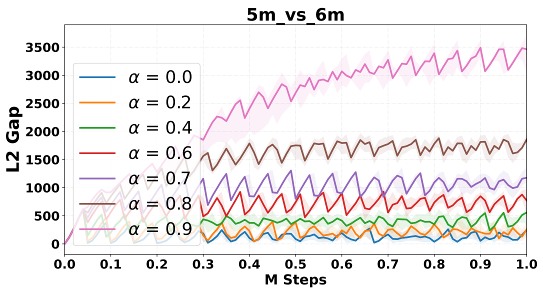
(a) 5m_vs_6m (b) 10m_vs_11m

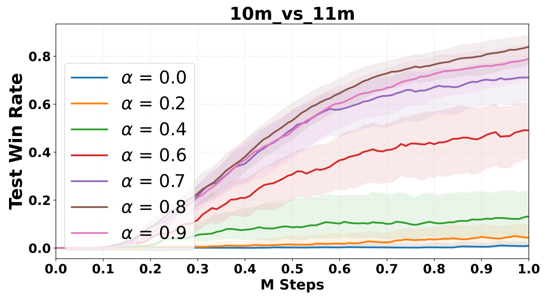
Figure 10. Results of QMIX with random amplitude scale ('Parallel + RAS') and MARR-based QMIX in the parallel setting.

D.2. Ablation Study for Interpolation Factor of MARR

The core hyperparameter in MARR is the interpolation factor α. Although we follow the recommended interpolation factor value $\alpha = 0.8$ in the multiagent version of Shrink & Perturb, we also conduct an ablation study to show how the interpolation factor α, which decides how much current network parameters are kept, influences the performance of MARR. The results of different α for MARR in both 5m_vs_6m and 10m_vs_11m are shown in Figure 11.


(a) Test win rate on 5m_vs_6m


(b) L2 gap on 5m_vs_6m


(c) Test win rate on 10m_vs_11m

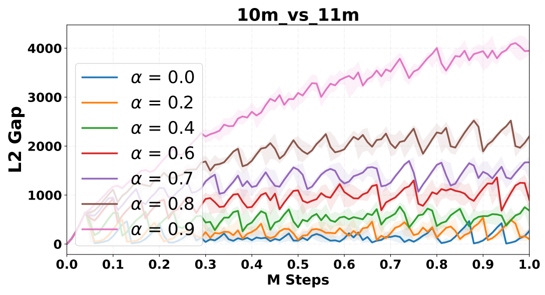
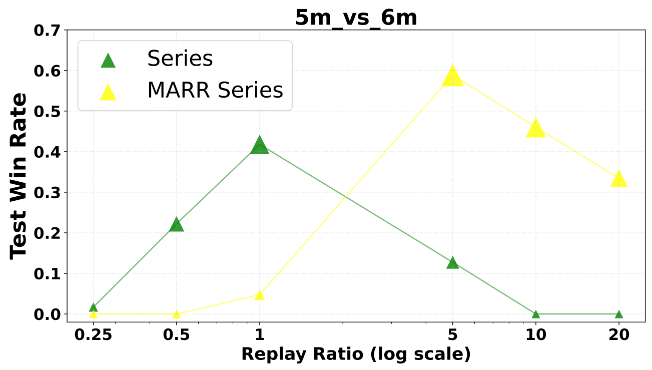

(d) L2 gap on 10m_vs_11m

Figure 11. Ablation study for the interpolation factor of Shrink & Perturb in MARR.

We can see that, although the smaller α has a closer L2 distance to the initial network parameters, MARR with a small α value does not perform well. This is because the learned knowledge in the trained network parameters is lost by resetting a large portion of network parameters to the initial ones. On the other hand, when $\alpha = 0.9$, the performance of MARR also decreases as the network plasticity gradually loses as indicated by the increasing L2 gap to the initial network parameters, which is shown in Figure 11(b) and 11(d). Only with a proper interpolation factor α such as 0.8, the learning progress and the network plasticity of MARR-based algorithms could be balanced towards the desired sample efficiency.

E. Experiments of Replay Ratio for MARR in Series Setting


(a) Different replay ratios in series and MARR-series setting on 5m_vs_6m

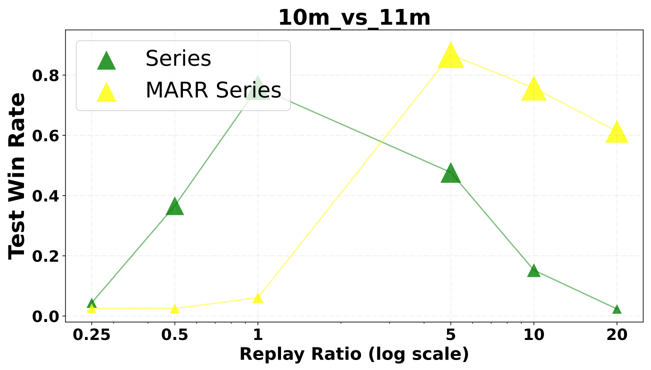

(b) Different replay ratios in series and MARR-series setting on 10m_vs_11m

Figure 12. Comparison of different replay ratios in the series environment setting with 1 million environment interaction steps. 'Series' means QMIX in the series environment setting. 'MARR-Series' means MARR-based QMIX in the series environment setting.

We also conduct experiments on the replay ratio of MARR in the series setting. In Figure 12, we can see the trend that

MARR shifts the replay ratio of QMIX to a higher degree in the series setting with higher performance. It indicates that MARR also has the ability to improve the sample efficiency of MARL algorithms in the series environment setting.

F. How Removing Shrink & Perturb and Random Amplitude Scale Affect Replay Ratio

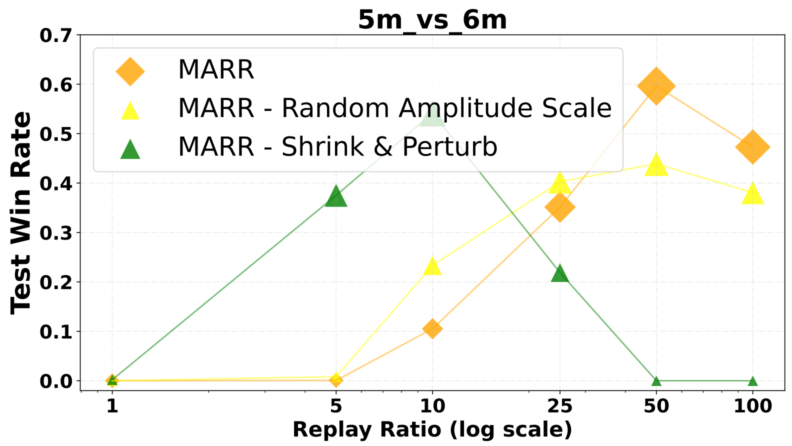


Figure 13. How removing Shrink & Perturb and removing Random Amplitude Scale affect the replay ratio in 1 million environment interaction steps. All settings are running in parallel environments. 'MARR - Random Amplitude Scale' means removing the Random Amplitude Scale. 'MARR - Shrink & Perturb' means removing the Shrink & Perturb. We report the median test win rate.

In Figure 13, we can see how removing the Random Amplitude Scale and Shrink & Perturb affects the optimal replay ratio. When removing the Shrink & Perturb, the optimal replay ratio changes from 50 to 10. When removing the Random Amplitude Scale, the optimal replay ratio is kept at 50.

G. Ablation Study on Random Amplitude Scale

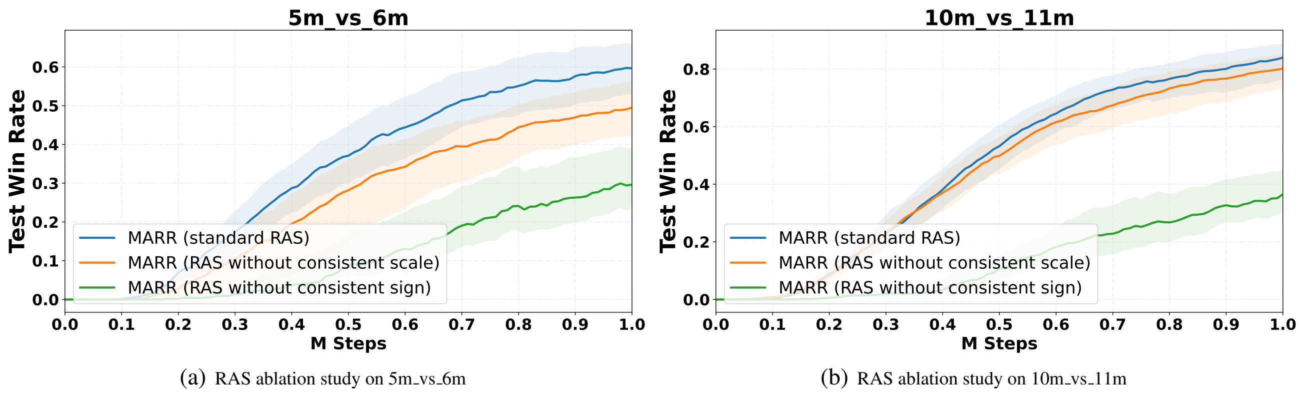
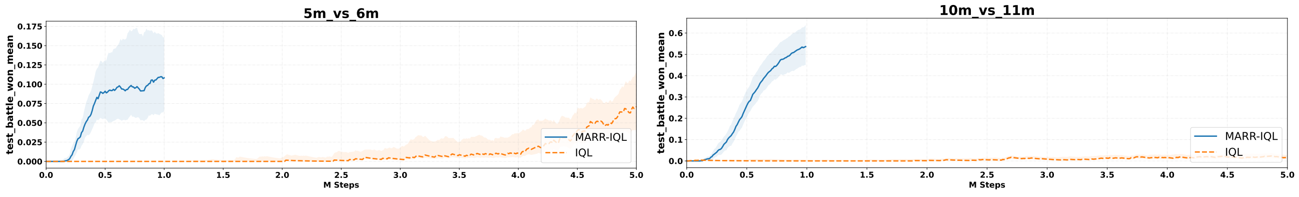


Figure 14. How intrinsic consistencies affect the performance of MARR. 'RAS' means Random Amplitude Scale.

The intrinsic consistencies mean that some intrinsic information is unchanged by random amplitude scaling such as the sign of inputs along adjacent time steps (Laskin et al., 2020). For example, in SMAC, the agent's local observation features include the relative x and relative y to other units. Through the signs of the relative x and relative y, the agent could know the direction of the allies and enemies ($x > 0$ and $y > 0$ mean that the unit is upper right to the agent). This is a kind of intrinsic information. The intrinsic consistency is ensured by random amplitude scaling under the common setting such as $z \sim U[0.8, 1.2]$ on each transition sample $(s_t, \mathbf{o}_t, \mathbf{a}, r, s'_{t+1}, \mathbf{o}'_{t+1})$ across the adjacent time steps t and $t + 1$.

Here we show how intrinsic consistencies behind the Random Amplitude Scale (RAS) affect the performance of MARR. Besides MARR with standard RAS, we also implement two variants. The first is RAS without the consistent scale, which means the amplitude scale z is different for the current observation and the next observation, as well as different for the current state and the next state. The second is RAS without the consistent sign, which means $|z|$ is the same for o and o', as well as the same for s and s', but with random signs ($+$ or $-$). The results are shown in Figure 14. We can see that intrinsic consistencies such as the same amplitude among two adjacent steps and the same amplitude sign are important to the Random Amplitude Scale.

H. MARR for Fully-Decentralized MARL



(a) MARR-IQL on 5m_vs_6m

(b) MARR-IQL on 10m_vs_11m

Figure 15. Performance of IQL and MARR-based IQL on 5m_vs_6m and 10m_vs_11m.

In this section, we test whether MARR could be generalized into fully decentralized MARL algorithms. We integrate MARR into the classical Independent Q-Learning (IQL) and the results are shown in Figure 15. Impressively, MARR could significantly boost the sample efficiency of IQL, especially in the scenario of 10m_vs_11m.

I. Limitations

Our study may have limitations under extensive consideration. One main limitation of MARR is that it is not suitable for the on-policy MARL algorithms as it utilizes the replay buffer with a high replay ratio. Another potential limitation or challenge is the data quality when applying the MARR algorithm to realistic multiagent scenarios. As MARR enforces a high relay ratio to the collected data samples, the data quality becomes important. When applying MARR to practical tasks, if the environment contains a lot of noise to pollute the collected data, MARR may not deal with this case well. Finally, when using the random amplitude scale for special cases such as linearly scaled features, it needs caution and further investigation.