Boosting Multiagent Reinforcement Learning at High Replay Ratios with Ensemble Reset

Yaodong Yang[1,2] **Hongyao Tang**[3] **Guangyong Chen**[4*] **Pheng-Ann Heng**[2]
[1]School of Computing and Artificial Intelligence,
Southwestern University of Finance and Economics, Chengdu, China
[2]CSE, The Chinese University of Hong Kong, Hong Kong, China
[3]School of Computer Software, Tianjin University, Tianjin, China
[4]HIM, Chinese Academy of Sciences, Hangzhou, China

Abstract

Increasing the replay ratio, where an agent's network is updated multiple times per environment interaction, is an effective strategy for improving sample efficiency in reinforcement learning. However, its effects on the network capacity of multiagent reinforcement learning (MARL) are not yet well investigated. In this paper, we show that high replay ratios induce a severe dormant neuron problem in the centralized global Q-network of MARL, where a large fraction of neurons become inactive, thereby reducing network capacity and destabilizing learning. To address this problem, we propose Ensemble Reset (EnSet) to stabilize MARL training at high replay ratios. First, guided by theoretical analysis, EnSet employs an ensemble of global Q-value networks with periodic resets to mitigate neuron dormancy during frequent updates. Second, EnSet diversifies replay experience by exploiting multiagent translation invariance as an inductive bias in the global Q-value function to prevent overfitting. Extensive experiments in SMAC, MPE, and SMACv2 environments demonstrate that EnSet consistently improves various MARL algorithms at high replay ratios with fewer environment steps.

1 Introduction

In recent years, multiagent reinforcement learning (MARL) has achieved significant advances in diverse domains that rely on collective intelligence, where multiple agents interact to solve complex sequential decision-making problems. Notable successes include real-time strategy games [1, 26], distributed energy management [32, 18], urban traffic signal control [28], etc. When applying MARL to real-world tasks, a major drawback of existing approaches is low sample efficiency, which requires a huge amount of environment interactions to learn satisfactory policies.

Although sample-efficient MARL is critical in real-world deployment, where interaction steps with an environment are often limited, improving sample efficiency in MARL remains challenging, with limited progress to date. Most existing works leverage the multiagent permutation prior to improve sample efficiency. For example, Ye et al. [33] generate additional data by applying a permutation transformation to homogeneous agents, exploiting permutation invariance. In addition, Yu et al. [34] utilize global system symmetry to augment data, where rotating a global state induces a permutation of the optimal joint policy. Later, Hao et al. [7] incorporate both permutation invariance and permutation equivariance inductive biases into agent network design to address the curse of dimensionality at the entity level. Complementary to these approaches, another line of research increases the replay ratio of MARL to improve sample efficiency. Replay ratio is defined as the number of network updates per environment interaction. Increasing it is an appealing strategy for improving sample efficiency

*Corresponding author: Guangyong Chen (chenguangyong@him.cas.cn). The code is available at GitHub.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

by performing more updates within a fixed environment interaction budget [2, 4]. Recently, Yang et al. [30] and Xu et al. [29] found that properly increasing the replay ratio is efficient in accelerating MARL learning. Specifically, Yang et al. [30] periodically reset network parameters to mitigate plasticity loss [17, 14] when training MARL at high replay ratios. However, despite recent progress, it remains unclear how high-frequency updates affect the internal status of network neurons and correspondingly alter agents' learning capacity in MARL.

In this paper, we explore effectively training MARL at high replay ratios by taking the perspective of the global Q-network as a key to further unlocking gains in sample efficiency. For the first time, we identify that the centralized global Q-network of MARL suffers from a severe dormant neuron problem at high replay ratios, where a large portion of network neurons become inactive, undermining network capacity and hindering learning. Prior work studied dormant neurons in MARL only at the standard replay ratio of 1 [20] or only in the agent network [29]. In contrast, we locate the failure at high replay ratios in the centralized global Q-network, which is unique to MARL and widely adopted in the popular CTDE paradigm. Accordingly, we develop a novel method named Ensemble Reset (EnSet) to effectively train MARL in the high-replay-ratio setting by two innovations: (1) global Q-network ensemble reset and (2) global Q-value translation invariance. First, along with theoretical analysis, we find that a network ensemble successfully mitigates the severe dormant neuron phenomenon. Motivated by this finding, EnSet employs an ensemble of global Q-networks combined with periodic parameter reset to reduce dormant neurons for stabilizing training. Second, EnSet diversifies the global state in replay experience using a prior of global Q-value translation invariance, where a system-level coordinate translation of units results in the same global Q-value, to prevent overfitting at a high update frequency. Extensive experiments in SMAC, MPE, and SMACv2 environments demonstrate that EnSet significantly improves multiple MARL algorithms at high replay ratios with far fewer environment interactions, while beating various network reset baselines.

2 Background

2.1 Markov Games

We use Markov games as the basic setting, which are an extended multiagent version of Markov Decision Processes [12]. Markov games are described by a state transition function, $T : S \times A_1 \times ... \times A_N \to P(S)$, which defines a probability distribution over all possible next states given the current global state $s \in S$ and each agent i's action $a_i \in A_i$. The reward is given based on the global state and actions of all agents $R_i : S \times A_1 \times ... \times A_N \to \mathbb{R}$. Meanwhile, Markov games can be partially observable, where each agent i perceives the environment through a local observation function $o_i = \mathcal{O}(s, i) \in O_i$, where O_i is agent i's observation space. Consequently, each agent learns a policy $\pi_i : O_i \to P(A_i)$ that maps each agent's 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 $\gamma \in [0, 1)$ is a discount factor. If all agents share the same reward function, i.e., $R_1 = ... = R_N$, Markov games are fully cooperative [15]: a best-interest action of one agent is best-interest for others. In this work, we focus on the partially observable and fully cooperative setting.

Many representative off-policy MARL algorithms, such as QMIX [21] and MADDPG [13], adopt the centralized training with decentralized execution (CTDE) paradigm to learn agent policies. CTDE allows agents to access global information during training for better coordination and to act independently based on local observations only when deployed for execution. In CTDE, there is usually a centralized global Q-value network, which evaluates the joint policy given a system's state and provides gradients to update each agent's decentralized policy, to ensure agents learn cooperative behaviors. In this work, we follow the CTDE paradigm.

2.2 Replay Ratio

Replay ratio refers to the number of updates of an agent's network for each environment interaction [4]. As each environment interaction comes at a cost, it is desirable to perform more updates with 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 (RL) [2, 4, 23, 16, 11], which has received much attention nowadays. For example, Kim et al. [9]

sequentially initialize and adaptively compose ensemble agents for sample efficiency at high replay ratios while preventing initialization collapses in safe RL.

To quantify sample efficiency and clarify the critical role of the replay ratio, we adopt a definition from Ye et al. [33], which formalizes the expected number of times each experience in the replay buffer is sampled for training 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 ratios for updating, which is performed N_{RR} times when interacting with the environment once. N_B is the batch size of a sampled batch for updating. The data acquisition speed V is the number of transition data experiences that are collected at each time step of environment interaction. T_U is the update interval where updating is conducted every T_U time steps. Typically, works focusing on training RL agents at high replay ratios [2] aim to increase N_{RR} while keeping N_B, V, and T_U unchanged. In this way, they achieve a higher sample efficiency indicated by a higher $\mathbb{E}[N_{sampled}]$ via different approaches, finally improving performance with the same number of environment interactions, i.e., the same number of collected data experiences. This behavior is also termed replay ratio scaling [4], which is much less studied in the MARL domain than single-agent RL. Until recently, Yang et al. [30] and Xu et al. [29] found that properly increasing the replay ratio is an efficient way to improve sample efficiency of MARL algorithms.

2.3 The Dormant Neuron Phenomenon

Sokar et al. [24] identify the dormant neuron phenomenon in deep RL, where an agent's network suffers from an increasing number of inactive neurons during the training process, thereby degrading network capacity. The definition of a dormant neuron is given below.

Definition 2.1 (α-Dormant Neuron)**.** Given an input distribution D, let $\rho_j^l(x)$ denote the activation of neuron j in layer l under input $x \in D$ and N_l be the number of neurons in layer l. The normalized activation score d_j^l of a neuron j in layer l is defined as follows:

$$d_j^l = \frac{\mathbb{E}_{x \in D} |\rho_j^l(x)|}{\frac{1}{N_l} \sum_{k=1}^{N_l} \mathbb{E}_{x \in D} |\rho_k^l(x)|}. \tag{2}$$

Then neuron j in layer l is α-dormant if its activation score $d_j^l \leq \alpha$. In this paper, we set α at 0.1 as in [20]. The dormant neuron rate of a network is defined as the proportion of α-dormant neurons.

A few works have started to study the dormant neuron phenomenon in MARL. For example, Xu et al. [29] use the dormant neuron rate as an indicator of plasticity when training MARL at high replay ratios. They found that the dormant neuron rate in an agent's policy network remains at a low level during training due to the agent's RNN structure. However, they do not consider dormant neurons in the global Q-network, which is the core module of CTDE-style MARL algorithms. More recently, Qin et al. [20] found that, in multiagent value factorization algorithms and at the standard replay ratio of 1, dormant neurons in the global Q-network increase as training proceeds. Accordingly, they propose ReBorn, which transfers weights from overactive neurons to dormant neurons to help learning. However, they do not further study in the high-replay-ratio setting, leaving effective scaling of the replay ratio for better sample efficiency unexplored. Another recent work is MARR [30], which investigates a closely related concept, plasticity loss [14], in MARL. MARR introduces a multiagent Shrink & Perturb strategy to periodically reset network parameters to maintain network plasticity. The multiagent Shrink & Perturb is defined as

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

and

$$\phi_t \leftarrow \alpha_r \phi_t + (1 - \alpha_r)\phi_0, \tag{4}$$

where θ_0^i are agent i's initial policy or individual Q-value network parameters. ϕ_0 are initial global Q-network parameters. θ_t^i and ϕ_t are the current agent and global Q-value network parameters, respectively. The interpolation factor α_r decides how much the current network parameters are kept. However, despite these recent advances, fundamental issues concerning the internal neuron status and learning capacity of MARL networks, particularly the global Q-value network, are underexplored at high replay ratios, thereby hindering further gains in MARL sample efficiency.

3 Ensemble Reset

In this section, we propose EnSet, a novel method designed to boost MARL at high replay ratios, concentrating on the global Q-value network. The whole framework of EnSet is illustrated in Figure 1. First, we found that the dormant neuron phenomenon in the global Q-network is exacerbated when improving replay ratios, as demonstrated in Section 3.1. Second, in Section 3.2, we show that a network ensemble, which employs multiple global Q-networks, surprisingly mitigates the phenomenon of dormant neurons, especially when combined with periodic network reset. Third, we introduce translation invariance to augment the state input of the global Q-network in Section 3.3.

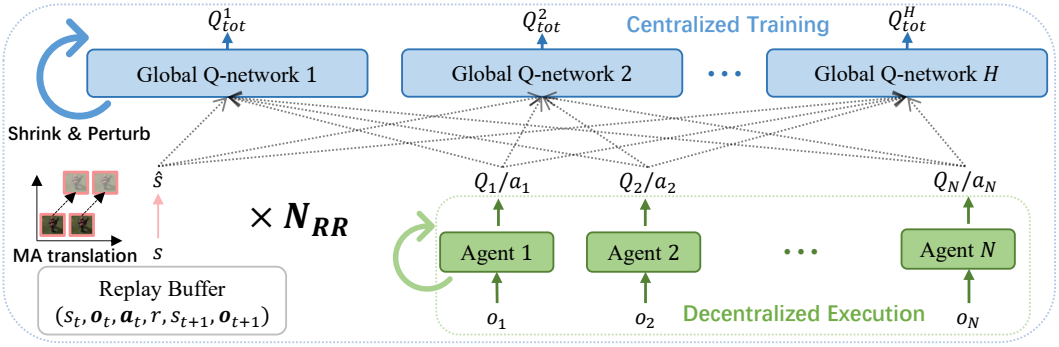


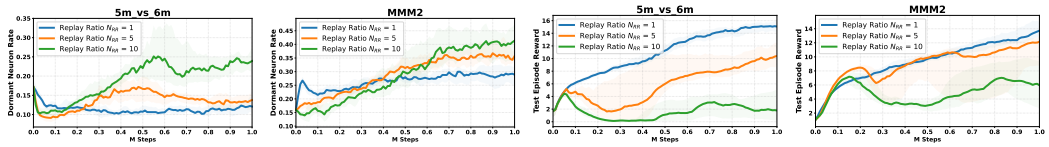
Figure 1: The illustration of the EnSet framework, which is trained under a high replay ratio of N_{RR}. The global Q-network ensemble with network reset tackles dormant neurons under high N_{RR}. The global state translation invariance is introduced to diversify the state s.

3.1 Dormant Neurons at High Replay Ratios

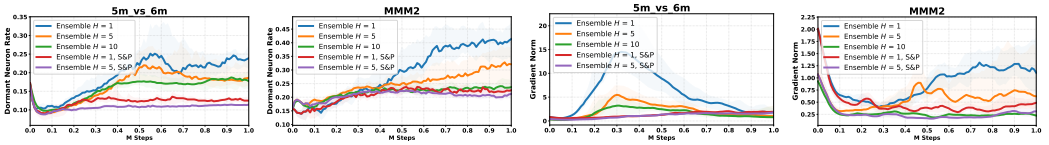
First, we show how dormant neurons in the global Q-network correlate with the replay ratios, which have not been explicitly studied before to the best of our knowledge. We use QMIX [21], one of the most popular MARL algorithms, as the tested base algorithm. Following Qin et al. [20], we measure dormant neurons in the hypernetworks of QMIX's global Q-network. In the standard setting, the replay ratio is 1, which means that network parameters are updated once after one episode. We increase the replay ratio to 5 and 10. Dormant neuron rates in the global Q-value network of QMIX under different replay ratios are given in Figure 2(a) and 2(b). The dormant neuron phenomenon becomes severe enough to hinder the learning of QMIX in the high-replay-ratio setting, as shown in Figure 2(c) and 2(d). Identifying and quantifying this phenomenon motivates us to reduce dormant neurons when increasing the replay ratio to boost the sample efficiency of MARL algorithms. The severe dormant neurons of the global Q-network in more scenarios, as well as the dormant neuron rates in the agent network maintaining a low level, are shown in Appendix A.

3.2 Global Q-network Ensemble with Network Reset

Next, we show that the network ensemble is an efficient way to tackle the severe dormant neuron phenomenon at high replay ratios. Specifically, Ensemble Reset utilizes an ensemble of global Q-networks with network parameter reset to stabilize MARL training at a high updating frequency. As shown in Figure 1, we employ H global Q-networks in the centralized training stage. In contrast, we do not enable ensemble on agent policy or individual Q-value networks. This is because previous works [29, 20], along with our complete analysis in Appendix A, show that no severe dormant neuron phenomenon exists in agent networks. Additionally, it ensures that, in the decentralized execution stage, no extra storage or computation resources are needed to deploy learned agent networks without the ensemble. When updating, each global Q-network receives the same mini-batch B with size N_B. To calculate the target global Q-value, each global Q-network with network parameters ϕ^h uses its own target network backup $\bar{\phi}^h$. The total loss is $L = \frac{1}{H}\sum_{h=1}^{H} L(\phi^h)$ and the loss function for the



(a) Dormant neuron on 5m_vs_6m. (b) Dormant neuron on MMM2. (c) Episode reward on 5m_vs_6m. (d) Episode reward on MMM2.



(e) Ensemble size on 5m_vs_6m. (f) Ensemble size on MMM2. (g) Gradient norm on 5m_vs_6m. (h) Gradient norm on MMM2.

Figure 2: (a-b) The dormant neuron rate in the global Q-network is exacerbated at high replay ratios. (c-d) When N_{RR} is 10, learning collapses with many dormant neurons being inactive. (e-f) Dormant neurons in global Q-networks are mitigated by the network ensemble and multiagent Shrink & Perturb (denoted as S&P) when N_{RR} is 10. (g-h) The ensemble smooths the network updating gradients. Results are over 6 independent seeds, showing the median with shaded 25-75% percentiles.

h-th global Q-value network is

$$L(\phi^h) = \frac{1}{N_B} \sum_{(s,\boldsymbol{a},s') \in B} (Q_{tot}^{\phi^h}(s,\boldsymbol{a}) - (r + \gamma \max_{\boldsymbol{a}'} Q_{tot}^{\bar{\phi}^h}(s',\boldsymbol{a}')))^2. \tag{5}$$

Each agent network θ^i is updated once per gradient step by the same total loss L as the global Q-networks. Each $L(\phi^h)$ depends on θ^i through the agent outputs fed into $Q_{tot}^{\phi^h}$. Since L averages the H losses, the agent gradient averages the gradient flows from all global Q-networks, $\frac{\partial L}{\partial \theta^i} = \frac{1}{H}\sum_{h=1}^{H} \frac{\partial L(\phi^h)}{\partial \theta^i}$ (Theorem B.2). This preserves the expected gradient magnitude of the single-network case. Dormant neuron rates with different global Q-network ensemble sizes under a replay ratio of 10 are shown in Figure 2(e) and 2(f). For an ensemble, the reported dormant neuron rate is the average over the H global Q-networks, each computed by Definition 2.1. The dormant neuron rate decreases as the ensemble size increases. We theoretically show that the ensemble reduces the gradient mean and variance of each global Q-network, thus reducing the probability of dormant neurons in Theorem 3.1, with a complete analysis provided in Appendix B. Figure 2(g) and 2(h) support our theoretical analysis as the gradient norm decreases if the ensemble size increases. These benefits hold even when all members reach similar quality, as detailed in Appendix B.2.

Theorem 3.1 (Reduced Probability of Neuron Dormancy)**.** *Under a fixed input x, consider a neuron with ReLU activation and pre-activation $z = w^T x$, with $z > 0$ as the neuron is already dormant for this input when $z \leq 0$. Let $g = \frac{\partial L}{\partial w}$ and $z' = z - \eta g^T x$ after a gradient step in the case of a single global Q-network. Assuming $g^T x \sim \mathcal{N}(\mu, \sigma^2)$, which is a common approximation under the central limit effect of mini-batch gradients [6], then under an ensemble of H networks, a neuron enters dormant with the probability as*

$$\mathbb{P}(z' < 0) = 1 - \Phi\Big(\frac{Hz/\eta - \mu}{\sigma}\Big), \tag{6}$$

where Φ denotes the cumulative distribution function (CDF) of the standard normal distribution. As H increases, the argument of Φ grows linearly, causing the tail probability to decay exponentially, reducing the probability of neuron dormancy. The complete proof is provided in Appendix B.

In addition to the global Q-network ensemble, to maintain network plasticity under high replay ratios, we incorporate the multiagent Shrink & Perturb strategy as in Equation (3) and (4) to reset network parameters to initial parameters periodically. As shown in Figure 2 (e-f), this combined ensemble reset operation further reduces dormant neurons in the global Q-value network.

Algorithm 1 Multiagent Reinforcement Learning with Ensemble Reset (EnSet)

1: Initialize each agent's policy or individual Q-value network parameters $\theta^1, \theta^2, ..., \theta^N$, an ensemble of H global Q-value network parameters $\phi^1, \phi^2, ..., \phi^H$ and an empty replay buffer D. Set target network parameters $\bar{\theta}^i \leftarrow \theta^i$ for $i = 1, 2, ..., N$, and $\bar{\phi}^h \leftarrow \phi^h$ for $h = 1, 2, ..., H$. 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: Each agent i takes action $a_{i,t} \sim \pi_{\theta^i}(\cdot|o_{i,t})$. Step into state s_{t+1}. Receive r_t and observe $o_{i,t+1}$.
4: Add transition data to the replay buffer: $D \leftarrow D \cup \{(s_t, \mathbf{o}_t, \mathbf{a}_t, r_t, s_{t+1}, \mathbf{o}_{t+1})\}$.
5: **if** $t \bmod T_U = 0$ **then**
6: **for** each update time n_{RR} from 1 to N_{RR} **do**
7: Sample a mini-batch $B = \{(s, \mathbf{o}, \mathbf{a}, r, s', \mathbf{o}')\}$ from D.
8: Perform the multiagent translation augmentation on global state s and s' to get $\hat{s}$ and $\hat{s}'$ in B.
9: Update each global Q-network on augmented mini-batch B by $L = \frac{1}{H}\sum_{h=1}^{H} L(\phi^h)$, where

$$L(\phi^h) = \frac{1}{N_B} \sum_{(\hat{s}, \boldsymbol{a}, \hat{s}') \in B} (Q_{tot}^{\phi^h}(\hat{s}, \boldsymbol{a}) - (r + \gamma \max_{\boldsymbol{a}'} Q_{tot}^{\bar{\phi}^h}(\hat{s}', \boldsymbol{a}')))^2.$$

10: Update each agent's Q-value network θ^i on augmented B by the total loss L (or each agent's policy network by $L_\pi = -\frac{1}{H}\sum_{h=1}^{H} Q_{tot}^{\phi^h}(\hat{s}, \boldsymbol{a})$ with $\boldsymbol{a}$ from the current policies).
11: **end for**
12: **end if**
13: **if** $t \bmod T_C = 0$ **then**
14: Update each target global Q-network $\bar{\phi}^h \leftarrow \phi^h$ and each target agent network $\bar{\theta}^i \leftarrow \theta^i$.
15: **end if**
16: **if** $t \bmod T_R = 0$ **then**
17: Shrink & Perturb as Equation (3) and (4) on each global Q-network ϕ^h and agent network θ^i.
18: **end if**
19: **end for**

3.3 Global Q-value Translation Invariance

Beyond ensemble reset, we found that diversifying the replay experience via intrinsic properties in multiagent systems is also beneficial in enhancing performance. Learning at a high replay ratio would make a network incur a risk of overfitting to earlier experiences, negatively affecting the rest of the learning process [17]. Therefore, the multiagent experience augmentation could be naturally incorporated into the high-replay-ratio setting of MARL. In this paper, we exploit a multiagent translation invariance [25] of the global Q-function to augment the state input of the global Q-network. For multiagent translation invariance, we have the following definition:

Definition 3.2 (Multiagent Translation Invariance)**.** In a D-dimensional coordinate system, a function $f : S \to Y$, where the global state $s \in S$ contains features of multiple agents and units as $s = (x_1, \cdots, x_N, x_o)$, $x_i = (c_i, e_i)$ includes the agent i's coordinate features $c_i \in \mathbb{R}^D$ as well as agent i's extra unit features e_i, and x_o represents other system features such as joint actions if f needs, is said to be translation invariant if for all $c_z \in \mathbb{R}^D$:

$$f((c_1 + c_z, e_1), \cdots, (c_N + c_z, e_N), x_o) = f(x_1, \cdots, x_N, x_o), \tag{7}$$

where a translation variable c_z in each coordinate dimension is uniformly sampled from $[-\alpha_z, \alpha_z]$. When f becomes the global Q-value function Q_{tot}, we are inspired to apply the multiagent translation invariance into the input global state s of the global Q-network as

$$Q_{tot}((c_1 + c_z, e_1), \cdots, (c_N + c_z, e_N), x_o) = Q_{tot}(x_1, \cdots, x_N, x_o). \tag{8}$$

This invariance property also applies to other units in the environment if they have coordinate features. If there are environmental boundaries or units that are not all included, such as terrain, the equation becomes an approximation. Although the multiagent translation invariance is a natural data augmentation operation for the global Q-value function, few works utilize this property in MARL.

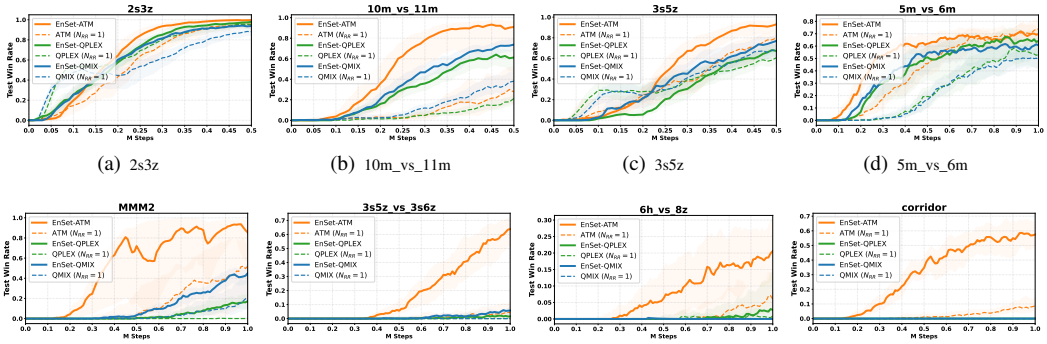
We give detailed algorithm descriptions of EnSet in Algorithm 1. EnSet follows a standard workflow of off-policy MARL algorithms. Specifically, in Line 1, EnSet employs an ensemble of H global Q-value networks. In Line 8, EnSet transforms a global state s to $\hat{s}$ in a mini-batch B with the multiagent translation augmentation. Then, in Line 9, each global Q-network is updated with a

temporal difference error calculated with the augmented global states. In Line 10, each agent network is updated with the average loss over the H global Q-networks. Finally, in Line 17, given a network reset interval of every T_R time steps, EnSet performs Shrink & Perturb to inject plasticity into both global Q-networks and agent networks. Next, we experimentally evaluate EnSet.

4 Experiments

In this section, we conduct comprehensive experiments to validate the proposed EnSet. First, we integrate EnSet into classical MARL algorithms such as QMIX [21], QPLEX [27], and ATM [31] in the StarCraft Multi-Agent Challenge (SMAC) platform [22] of a discrete action space in Section 4.1. Second, in Section 4.2, we compare EnSet with other multiagent network reset techniques in the high-replay-ratio setting. Third, we test EnSet in the classical multiagent particle environment (MPE) [13] of a continuous action space by plugging it into MADDPG [13] and FACMAC [19] in Section 4.3. Fourth, further experiments of combining EnSet with finetuned QMIX in SMACv2 [5] are given in Section 4.4. Fifth, we conduct the ablation study of EnSet in Section 4.5. Finally, we compare EnSet with standard MARL algorithms given more environment steps in Section 4.6.

4.1 StarCraft II Multi-Agent Challenge



(a) 2s3z (b) 10m_vs_11m (c) 3s5z (d) 5m_vs_6m

(e) MMM2 (f) 3s5z_vs_3s6z (g) 6h_vs_8z (h) corridor

Figure 3: Benchmark experiments of EnSet for boosting MARL in SMAC. EnSet-QMIX, EnSet-QPLEX, and EnSet-ATM are trained with a replay ratio of 10, while QMIX, QPLEX, and ATM are trained with a default replay ratio value of 1.

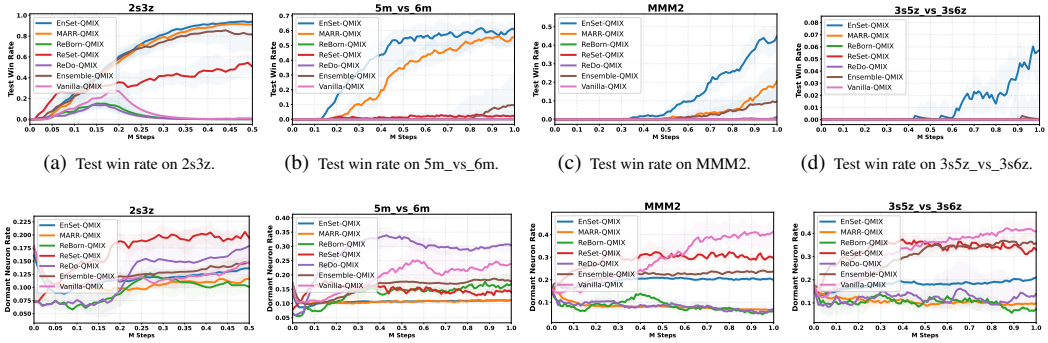
First, we experiment in StarCraft Multi-Agent Challenge (SMAC) [22], a widely adopted testbed for evaluating MARL approaches. We train multiple agents to control allied units respectively to beat enemies, while a built-in handcrafted AI controls these enemy units. The SMAC environment is with discrete action space, and the version of StarCraft II is 4.6.2. We implement EnSet based on the pymarl framework [22] with default training and evaluation configurations. In SMAC, we evaluate on 8 tasks, i.e., 2s3z, 10m_vs_11m, 3s5z, 5m_vs_6m, MMM2, 3s5z_vs_3s6z, 6h_vs_8z, and corridor. These tasks include homogeneous and heterogeneous multiagent scenarios, as well as symmetrical and asymmetrical multiagent scenarios for a comprehensive evaluation. For example, on 2s3z, each side has 2 Stalkers and 3 Zealots, where the system is heterogeneous and symmetrical. More descriptions of these tasks are given in Appendix D.

To validate that EnSet boosts MARL at high replay ratios, we integrate EnSet into QMIX, QPLEX, and ATM with a replay ratio of 10, where 10 updates are executed after one episode. For EnSet, the ensemble size H of the global Q-network is set at 5. The coordinate translation variable α_z in Equation (7) is set at 0.2. The hyperparameters of multiagent Shrink & Perturb follow MARR [30]. All these special hyperparameters of EnSet are summarized in Appendix C and kept the same across methods and tasks. We run all methods over 6 independent training runs with different random seeds.

The experimental results are shown in Figure 3, including the median performance as well as the shaded 25-75% percentiles. EnSet significantly boosts QMIX, QPLEX, and ATM. For easy scenarios, including 2s3z, 10m_vs_11m, and 3s5z, the number of environment interactions is set at 0.5 million. With such a small interaction budget, EnSet successfully accelerates learning, particularly

on 10m_vs_11m, demonstrating its superior sample efficiency. For the remaining scenarios, we set the number of environment interactions at 1 million steps. EnSet also substantially speeds up the learning of each MARL algorithm, while standard ones with a default replay ratio of 1 learn slowly. Specifically, in super hard scenarios such as 3s5z_vs_3s6z and corridor, EnSet successfully improves the performance of ATM by a large margin. These results indicate that EnSet is efficient in boosting MARL algorithms at a high replay ratio to achieve sample efficiency. Additionally, results of EnSet with different global Q-network ensemble sizes H are reported in Appendix H. Results of EnSet with a standard replay ratio of 1 are given in Appendix I. Although EnSet-QMIX at a replay ratio of 10 performs 10 times more gradient updates than QMIX at a replay ratio of 1, it takes only about twice the training time (Table 3), with more details in Appendix D.

4.2 Comparing Other MARL Network Reset Methods



(a) Test win rate on 2s3z. (b) Test win rate on 5m_vs_6m. (c) Test win rate on MMM2. (d) Test win rate on 3s5z_vs_3s6z.

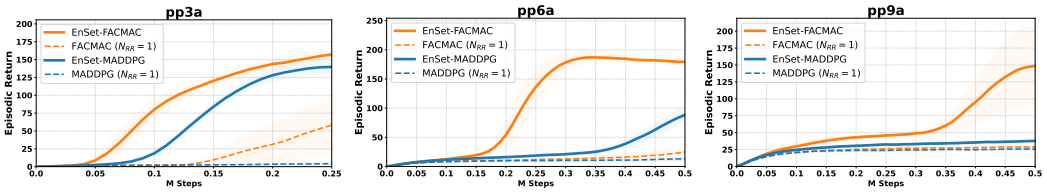
(e) Dormant neuron on 2s3z. (f) Dormant neuron on 5m_vs_6m. (g) Dormant neuron on MMM2. (h) Dormant neuron on 3s5z_vs_3s6z.

Figure 4: Benchmarking different MARL network reset methods in SMAC. Replay ratio is set at 10.

Second, we compare EnSet with other MARL network reset methods such as MARR [30], ReBorn [20], ReSet [17], and ReDo [24] on the basis of QMIX. MARR [30] introduces the Shrink & Perturb strategy into MARL to reset network parameters periodically. ReBorn [20] transfers weights from overactive neurons to dormant neurons to prevent both overactive and dormant neurons. ReSet [17] addresses early agent experience bias by resetting the last layer of a neural network to avoid overfitting. ReDo [24] reinitializes the input weights of dormant neurons and zeros the output weights of dormant neurons. We also present ensemble QMIX with multiple global Q-networks and vanilla QMIX under a replay ratio of 10 for comparison. The performance and the dormant neuron rates of the global Q-network are plotted in Figure 4. It is clear that EnSet maintains dormant neuron rates at a low level and achieves the best performance compared to baselines. On the other hand, ensemble QMIX is not sufficient to solve challenging scenarios although it performs well on 2s3z.

Interestingly, despite the fact that MARR, ReBorn, and ReDo have lower dormant neuron rates than EnSet in MMM2 and 3s5z_vs_3s6z, they perform worse than EnSet. This indicates a one-directional relation between dormant neurons and performance. Severe dormancy at high replay ratios is sufficient to destroy learning (Figure 2). In contrast, minimizing dormancy is not sufficient for good learning, and how dormancy is reduced matters. Two further interventions support this. First, replacing the ensemble in EnSet with a learning rate divided by H mimics the gradient scaling of the ensemble without its diversity. It yields an even lower dormant neuron rate than EnSet, yet performs clearly worse (Appendix B.1). Second, replacing Shrink & Perturb in EnSet with ReSet drives the dormant neuron rate close to 0 on all maps, yet performs poorly, since hard reinitialization destroys learned knowledge along with dormancy (Appendix L). Forcing a much lower dormant neuron rate may constrain network parameters and hurt network representation ability. Moreover, post-hoc interventions, such as ReBorn and ReDo, may frequently interfere with ongoing learning dynamics at high replay ratios. In summary, EnSet reduces dormant neurons via ensemble and periodically shrinks and perturbs learning neurons, thereby balancing the representation capacity and learned knowledge to achieve superior performance. We also eliminate the possibility that a single large global Q-network could achieve similar effects to EnSet in Appendix J. More comparisons between EnSet and MARR are provided in Appendix G.

4.3 Multiagent Particle Environment



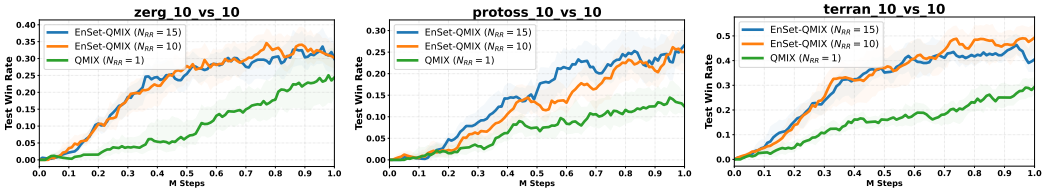
(a) pp3a. (b) pp6a. (c) pp9a.

Figure 5: EnSet boosts MARL in the MPE environment. EnSet-MADDPG and EnSet-FACMAC have a replay ratio of 25, while the standard MADDPG and FACMAC have a replay ratio of 1.

We further evaluate EnSet in the Multiagent Particle Environment (MPE) [13], which features a continuous action space. Following [19], we adopt a set of predator-prey tasks where multiple slower cooperative agents must capture faster prey in a continuous two-dimensional toroidal space with obstacle landmarks. More environment details are provided in Appendix D. We experiment on 3 tasks with different agent numbers, i.e., pp3a where 3 agents catch 1 prey, pp6a where 6 agents catch 2 preys, and pp9a where 9 agents catch 3 preys. Training configurations follow [19]. Hyperparameters of EnSet in MPE are the same as in SMAC and listed in Appendix C. Next, we test EnSet by combining it with MADDPG and FACMAC with a replay ratio of 25. In both actor-critic algorithms, the global Q-networks are trained by the total loss L, and each actor is updated by the averaged policy loss $L_\pi = -\frac{1}{H}\sum_{h=1}^{H} Q_{tot}^{\phi^h}(\hat{s}, \boldsymbol{a})$, where $\boldsymbol{a}$ is given by the current policies. Results in MPE are shown in Figure 5, which are averaged over 6 independent runs with a 95% confidence interval. Impressively, EnSet boosts the learning of both MADDPG and FACMAC, indicating EnSet is general for off-policy MARL algorithms. The dormant neuron rates are given in Appendix E. Additional experiments on how different replay ratios affect EnSet's performance are provided in Appendix K.

4.4 The SMACv2 Environment

Next, we evaluate EnSet in SMACv2 [5], which enforces sufficient stochasticity and meaningful partial observability for benchmarking MARL algorithms. We conduct the experiments with EnSet on 3 scenarios, including zerg_10_vs_10, protoss_10_vs_10, and terran_10_vs_10. More details of these SMACv2 tasks refer to Appendix D. In SMACv2, we build EnSet on top of pymarl2 [8], which introduces various code-level optimizations for MARL. We follow the default hyperparameters of pymarl2, where the sampling environment number is optimized to 8. The hyperparameters of EnSet are the same as in SMAC and MPE. We experimented with EnSet-QMIX using two replay ratios of 10 and 15. The results are shown in Figure 6, including the median performance as well as the shaded 25-75% percentiles. We see that, under both high replay ratios, EnSet successfully boosts QMIX in these challenging SMACv2 tasks. The dormant neuron rates are given in Appendix F.

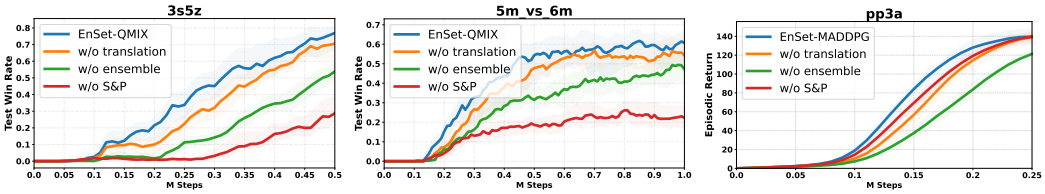


(a) zerg_10_vs_10. (b) protoss_10_vs_10. (c) terran_10_vs_10.

Figure 6: Benchmark experiments of EnSet for boosting MARL in SMACv2. EnSet-QMIX is trained with a replay ratio of 10 or 15, while QMIX is trained with a default replay ratio value of 1.

4.5 Ablation Study of EnSet

We conduct the ablation study to validate each component of EnSet, including the global Q-network ensemble, the multiagent Shrink & Perturb strategy, as well as the multiagent state translation. The

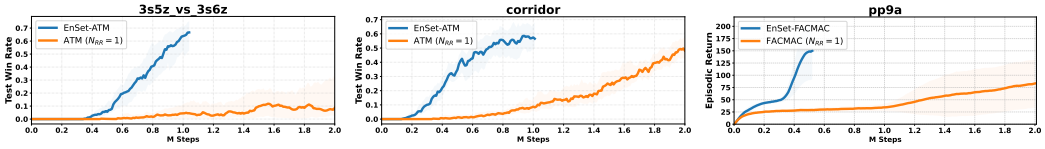


(a) Ablation study on 3s5z. (b) Ablation study on 5m_vs_6m. (c) Ablation study on pp3a.

Figure 7: The ablation study of EnSet in SMAC and MPE. Replay ratio is 10 in SMAC and 25 in MPE. 'w/o ensemble' means the number of global Q-networks is 1. 'w/o translation' means no multiagent translation for states. 'w/o S&P' means no multiagent Shrink & Perturb.

ablation results of these components are shown in Figure 7. We see that the global Q-network ensemble and multiagent Shrink & Perturb in ensemble reset largely affects performance, implying that tackling severe dormant neurons is key to keeping effective learning at high replay ratios. Specifically, the ensemble contributes the most to the performance in pp3a, further highlighting its importance. Meanwhile, the multiagent state translation consistently improves performance slightly during learning in the tested scenarios. More comparison of multiagent state translation with other state augmentation techniques is available in Appendix M.

4.6 Comparing Standard MARL with More Environment Steps



(a) 3s5z_vs_3s6z. (b) corridor. (c) pp9a.

Figure 8: The comparison of EnSet with standard MARL algorithms with more environment steps. EnSet-ATM has a high replay ratio of 10 in SMAC. EnSet-FACMAC has a high replay ratio of 25 in MPE. The standard ATM and FACMAC are of a default replay ratio of 1.

We further compare EnSet with the standard MARL algorithms given more environment steps on the most challenging tasks, including 3s5z_vs_3s6z and corridor in SMAC and pp9a in MPE, as shown in Figure 8. We use ATM and FACMAC as the tested algorithms, which show superior performance in SMAC and MPE, respectively. On 3s5z_vs_3s6z and corridor, even with twice the environment steps, the standard ATM still performs worse than EnSet-ATM. On pp9a, the standard FACMAC with four times the environment steps also performs worse than EnSet-FACMAC.

5 Conclusion

In this paper, we propose EnSet to boost MARL at high replay ratios for sample efficiency. EnSet consists of two key innovations concentrated on the global Q-network: the global Q-network ensemble reset and global Q-value translation invariance. First, we found that dormant neurons in the global Q-network become severe at high replay ratios. As indicated by our theoretical analysis, the global Q-network ensemble effectively mitigates this issue to stabilize learning, together with network reset. Second, we introduce multiagent translation invariance into the global state to generate diverse replay experiences for training the global Q-network. Extensive experiments in SMAC, MPE, and SMACv2 show that EnSet substantially improves the learning of MARL in the high-replay-ratio setting.

For future work, first, dynamically adjusting the replay ratio during MARL learning is interesting. Second, integrating EnSet into on-policy MARL algorithms (e.g., MAPPO) is worth exploring. Third, adapting EnSet for substantially larger MARL settings, with up to hundreds of agents, is valuable. Fourth, studying the ensemble and reset mechanism for dormant neurons at high replay ratios in single-agent RL is also worthwhile, since different pathologies and solutions may arise there.

Acknowledgments and Disclosure of Funding

This work was supported by the National Key R&D Program of China for Young Scientists (Grant No. 2025YFC2708700) and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. T45-401/22-N). The authors declare no competing interests.

References

[1] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Christopher Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique Pondé de Oliveira Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with Large Scale Deep Reinforcement Learning. *CoRR*, abs/1912.06680, 2019. arXiv: 1912.06680.

[2] Xinyue Chen, Che Wang, Zijian Zhou, and Keith W. Ross. Randomized Ensembled Double Q-Learning: Learning Fast Without a Model. In *International Conference on Learning Representations*, 2021.

[3] Shibhansh Dohare, J. Fernando Hernandez-Garcia, Qingfeng Lan, Parash Rahman, A. Rupam Mahmood, and Richard S. Sutton. Loss of plasticity in deep continual learning. *Nature*, 632(8026):768–774, August 2024.

[4] Pierluca D'Oro, Max Schwarzer, Evgenii Nikishin, Pierre-Luc Bacon, Marc G. Bellemare, and Aaron Courville. Sample-Efficient Reinforcement Learning by Breaking the Replay Ratio Barrier. In *The Eleventh International Conference on Learning Representations*, 2023.

[5] Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob N. Foerster, and Shimon Whiteson. SMACv2: an improved benchmark for cooperative multi-agent reinforcement learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc. event-place: New Orleans, LA, USA.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[7] Jianye Hao, Xiaotian Hao, Hangyu Mao, Weixun Wang, Yaodong Yang, Dong Li, YAN ZHENG, and Zhen Wang. Boosting Multiagent Reinforcement Learning via Permutation Invariant and Permutation Equivariant Networks. In *The Eleventh International Conference on Learning Representations*, 2023.

[8] Jian Hu, Siying Wang, Siyang Jiang, and Musk Wang. Rethinking the Implementation Tricks and Monotonicity Constraint in Cooperative Multi-agent Reinforcement Learning. In *The Second Blogpost Track at ICLR 2023*, 2023.

[9] Woojun Kim, Yongjae Shin, Jongeui Park, and Youngchul Sung. Sample-efficient and safe deep reinforcement learning via reset deep ensemble agents. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc. event-place: New Orleans, LA, USA.

[10] Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. event-place: Vancouver, BC, Canada.

[11] Hojoon Lee, Youngdo Lee, Takuma Seno, Donghu Kim, Peter Stone, and Jaegul Choo. Hyperspherical Normalization for Scalable Deep Reinforcement Learning. In *Forty-second International Conference on Machine Learning*, 2025.

[12] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings*, pages 157–163. Elsevier, 1994.

[13] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6382–6393, Red Hook, NY, USA, 2017. Curran Associates Inc. event-place: Long Beach, California, USA.

[14] Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, Honolulu, Hawaii, USA, 2023. JMLR.org.

[15] Laetitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. *The Knowledge Engineering Review*, 27(1):1–31, 2012.

[16] Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample-efficient continuous control. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2025. Curran Associates Inc. event-place: Vancouver, BC, Canada.

[17] Evgenii Nikishin, Max Schwarzer, Pierluca D'Oro, Pierre-Luc Bacon, and Aaron Courville. The Primacy Bias in Deep Reinforcement Learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 16828–16847. PMLR, July 2022.

[18] Guido Novati, Hugues Lascombes De Laroussilhe, and Petros Koumoutsakos. Automating turbulence modelling by multi-agent reinforcement learning. *Nature Machine Intelligence*, 3(1):87–96, January 2021.

[19] Bei Peng, Tabish Rashid, Christian A. Schroeder de Witt, Pierre-Alexandre Kamienny, Philip H. S. Torr, Wendelin Böhmer, and Shimon Whiteson. FACMAC: factored multi-agent centralised policy gradients. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, NIPS '21, Red Hook, NY, USA, 2021. Curran Associates Inc.

[20] Haoyuan Qin, Chennan Ma, Mian Deng, Zhengzhu Liu, Songzhu Mei, Xinwang Liu, Cheng Wang, and Siqi Shen. The Dormant Neuron Phenomenon in Multi-Agent Reinforcement Learning Value Factorization. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 35727–35759. Curran Associates, Inc., 2024.

[21] Tabish Rashid, Mikayel Samvelyan, Christian Schröder de Witt, Gregory Farquhar, Jakob N. Foerster, and Shimon Whiteson. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4292–4301, 2018.

[22] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob Foerster, and Shimon Whiteson. The StarCraft Multi-Agent Challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, pages 2186–2188, Richland, SC, 2019. International Foundation for Autonomous Agents and Multiagent Systems. event-place: Montreal QC, Canada.

[23] Max Schwarzer, Johan Obando-Ceron, Aaron Courville, Marc G. Bellemare, Rishabh Agarwal, and Pablo Samuel Castro. Bigger, Better, Faster: Human-Level Atari with Human-Level Efficiency. In *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 2023.

[24] Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The Dormant Neuron Phenomenon in Deep Reinforcement Learning. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023. Place: Honolulu, Hawaii, USA.

[25] Cristian-Ioan Vasile, Mac Schwager, and Calin Belta. SE(N) invariance in networked systems. In *2015 European Control Conference (ECC)*, pages 186–191. IEEE, July 2015.

[26] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, November 2019.

[27] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. QPLEX: Duplex Dueling Multi-Agent Q-Learning. In *International Conference on Learning Representations*, 2021.

[28] Tong Wu, Pan Zhou, Kai Liu, Yali Yuan, Xiumin Wang, Huawei Huang, and Dapeng Oliver Wu. Multi-Agent Deep Reinforcement Learning for Urban Traffic Light Control in Vehicular Networks. *IEEE Transactions on Vehicular Technology*, 69(8):8243–8256, August 2020.

[29] Linjie Xu, Zichuan Liu, Alexander Dockhorn, Diego Perez-Liebana, Jinyu Wang, Lei Song, and Jiang Bian. Higher Replay Ratio Empowers Sample-Efficient Multi-Agent Reinforcement Learning. In *2024 IEEE Conference on Games (CoG)*, pages 1–8, Milan, Italy, August 2024. IEEE.

[30] Yaodong Yang, Guangyong Chen, Jianye Hao, and Pheng Ann Heng. Sample-efficient multiagent reinforcement learning with reset replay. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, Vienna, Austria, 2024. JMLR.org.

[31] Yaodong Yang, Guangyong Chen, Weixun Wang, Xiaotian Hao, Jianye Hao, and Pheng Ann Heng. Transformer-based working memory for multiagent reinforcement learning with action parsing. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. event-place: New Orleans, LA, USA.

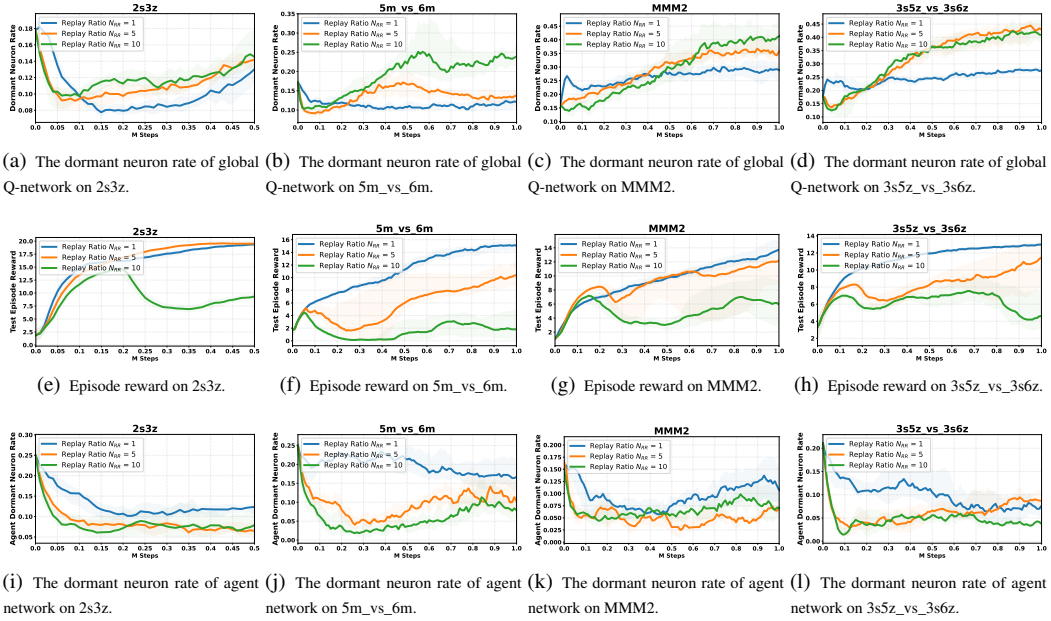
[32] Yaodong Yang, Jianye Hao, Yan Zheng, and Chao Yu. Large-Scale Home Energy Management Using Entropy-Based Collective Multiagent Deep Reinforcement Learning Framework. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 630–636. International Joint Conferences on Artificial Intelligence Organization, 2019.

[33] Zhenhui Ye, Yining Chen, Xiaohong Jiang, Guanghua Song, Bowei Yang, and Sheng Fan. Improving sample efficiency in Multi-Agent Actor-Critic methods. *Applied Intelligence*, 52(4):3691–3704, 2022.

[34] Xin Yu, Rongye Shi, Pu Feng, Yongkai Tian, Jie Luo, and Wenjun Wu. ESP: Exploiting Symmetry Prior for Multi-Agent Reinforcement Learning. In *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2023.

A More Results of Dormant Neurons at High Replay Ratios

Here, we show more results of dormant neurons in the high-replay-ratio setting in Figure 9. We observe that, when the replay ratio is either 5 or 10, the dormant neuron rate in the global Q-network is higher than when the replay ratio is 1. Furthermore, when the replay ratio reaches 10, the learning cripples in all the tested cases. This necessitates EnSet, which stabilizes MARL at high replay ratios by tackling dormant neurons in the global Q-network. At the same time, the dormant neuron rate in the agent network maintains a low level even when the replay ratio increases.



(a) The dormant neuron rate of global Q-network on 2s3z. (b) The dormant neuron rate of global Q-network on 5m_vs_6m. (c) The dormant neuron rate of global Q-network on MMM2. (d) The dormant neuron rate of global Q-network on 3s5z_vs_3s6z.

(e) Episode reward on 2s3z. (f) Episode reward on 5m_vs_6m. (g) Episode reward on MMM2. (h) Episode reward on 3s5z_vs_3s6z.

(i) The dormant neuron rate of agent network on 2s3z. (j) The dormant neuron rate of agent network on 5m_vs_6m. (k) The dormant neuron rate of agent network on MMM2. (l) The dormant neuron rate of agent network on 3s5z_vs_3s6z.

Figure 9: The dormant neuron phenomenon in the global Q-network is exacerbated when increasing replay ratios. Higher replay ratios cause learning instability. When the replay ratio is 10, the learning collapses with lots of dormant neurons being inactive in the global Q-network. On the contrary, the dormant neuron rate in the agent network maintains a low level even when the replay ratio is high.

B Theoretical Analysis of Global Q-network Ensemble

In this section, we analyze how the global Q-network ensemble benefits the updating of the global Q-network and the individual agent policy/Q-value network.

Theorem B.1 (Gradient Scaling of the Global Q-value Network)**.** *Consider an ensemble of H independent global Q-value networks ϕ^h with the ensemble loss*

$$L = \frac{1}{H}\sum_{k=1}^{H} L^k. \tag{9}$$

Then the gradient of each global Q-value network ϕ^h is scaled by $\frac{1}{H}$ compared to a single network as

$$\frac{\partial L}{\partial \phi^h} = \frac{1}{H}\frac{\partial L^h}{\partial \phi^h}. \tag{10}$$

Proof. By the chain rule,

$$\frac{\partial L}{\partial \phi^h} = \frac{\partial L}{\partial L^h}\frac{\partial L^h}{\partial \phi^h} = \frac{\partial \frac{1}{H}\sum_{k=1}^{H} L^k}{\partial L^h}\frac{\partial L^h}{\partial \phi^h} = \frac{1}{H}\frac{\partial L^h}{\partial \phi^h}.$$

□

Therefore, the gradient of global Q-value networks ϕ^h becomes $\frac{1}{H}$ times smaller after applying the global Q-network ensemble. In the case of a single global Q-value network where $H = 1$, if we take ϕ^h as the parameters of the single global Q-value network, the gradient of the single global Q-value network becomes

$$\frac{\partial L}{\partial \phi^h} = \frac{\partial L^h}{\partial \phi^h}. \tag{11}$$

It means that, for a sampled experience batch B, the information indicated by the value of L is updated into the ensemble networks via gradients that are reduced by a factor of $\frac{1}{H}$ compared to the single global Q-value network case.

Similarly, in the case of a single global Q-value network, if we take ϕ^h as the parameters of the single global Q-value network, the gradient of the individual agent Q-value/policy network becomes

$$\frac{\partial L}{\partial \theta^i} = \frac{\partial L^h}{\partial \theta^i}. \tag{12}$$

Then, we investigate how the global Q-network ensemble influences the gradient of the individual agent network.

Theorem B.2 (Gradient Preservation of Agent Networks)**.** *Let θ^i denote the parameters of an individual agent i's Q-value/policy network. Then the expected gradient magnitude for θ^i is preserved under the ensemble:*

$$\frac{\partial L}{\partial \theta^i} = \frac{1}{H} \sum_{h=1}^{H} \frac{\partial L^h}{\partial \theta^i}. \tag{13}$$

Proof. Each agent receives independent gradient flow from each global network:

$$\frac{\partial L}{\partial \theta^i} = \sum_{h=1}^{H} \frac{\partial L}{\partial L^h} \frac{\partial L^h}{\partial \theta^i} = \sum_{h=1}^{H} \frac{\partial \frac{1}{H} \sum_{k=1}^{H} L^k}{\partial L^h} \frac{\partial L^h}{\partial \theta^i} = \sum_{h=1}^{H} \frac{1}{H} \frac{\partial L^h}{\partial \theta^i} = \frac{1}{H} \sum_{h=1}^{H} \frac{\partial L^h}{\partial \theta^i}.$$

Hence, the expected magnitude is the same as in the single-network case. □

It means that, for a sampled experience batch B, the information indicated by the value of L will be updated into an agent Q-value/policy network via the same magnitude of expected gradients as the case of a single global Q-value network. This indicates that the global Q-network ensemble brings the minimal influence to the agent network, which does not suffer from severe dormant neurons.

Next, we examine the variance of the gradient. When there is a single global Q-value network, we remark $Var(\frac{\partial L^h}{\partial \phi^h}) = \Sigma$.

Theorem B.3 (Variance Reduction of Ensemble Gradients)**.** *Assuming independent gradients across global Q-value networks and $Var(\frac{\partial L^h}{\partial \phi^h}) = \Sigma$, the variance of the gradient for each network in the ensemble is reduced by a factor of $\frac{1}{H^2}$ as*

$$Var\Big(\frac{\partial L}{\partial \phi^h}\Big) = \frac{1}{H^2}\Sigma. \tag{14}$$

Proof. By scaling the gradient of each network by $\frac{1}{H}$ and assuming independence, we have

$$Var\Big(\frac{\partial L}{\partial \phi^h}\Big) = Var\Big(\frac{1}{H}\frac{\partial L^h}{\partial \phi^h}\Big) = \frac{1}{H^2} Var\Big(\frac{\partial L^h}{\partial \phi^h}\Big) = \frac{1}{H^2}\Sigma.$$

This smooths the parameter updates and reduces extreme gradient effects. □

The variance of the update gradient for each global Q-value network decreases by $\frac{1}{H^2}$ while the gradient magnitude is reduced by $\frac{1}{H}$. This reduction in update magnitude makes the updates for each network in the ensemble smoother, mitigating the impact of extreme gradients on neurons at each update. Consequently, it alleviates the issue in a single network where certain neurons may enter saturation or become dormant due to large-gradient updates, especially at high replay ratios. Furthermore, by distributing the learning of the same sample information across multiple networks,

the ensemble effectively stores knowledge in a distributed manner, preventing excessive specialization or premature dormancy of neurons.

Next, we show that the probability of dormant neurons in global Q-networks decreases with reduced gradient mean and variance. Under a fixed input x, a neuron with ReLU activation becomes inactive when its pre-activation $z = w^T x \leq 0$, resulting in zero local gradient with respect to its parameters for that input. In this case, subsequent updates driven by the same input x will not modify this neuron, making it difficult to reactivate. High-variance gradient updates can cause large fluctuations in parameters, increasing the probability that z crosses into the negative region as $z' = z - \eta g^T x$ (see Theorem B.4), where η is the learning rate and $g = \frac{\partial L}{\partial w}$ is the gradient. Once in this region, the neuron receives no gradient signal from this input, and thus tends to remain inactive under repeated updates with the same input.

Theorem B.4 (Reduced Probability of Neuron Dormancy)**.** *Under a fixed input x, consider a neuron with ReLU activation and pre-activation $z = w^T x$, with $z > 0$ as the neuron is already dormant for this input when $z \leq 0$. Let $g = \frac{\partial L}{\partial w}$ and $z' = z - \eta g^T x$ after a gradient step in the case of a single global Q-network. Assuming $g^T x \sim \mathcal{N}(\mu, \sigma^2)$, which is a common approximation under the central limit effect of mini-batch gradients [6], then under an ensemble of H networks, a neuron enters dormant with the probability as*

$$\mathbb{P}(z' < 0) = 1 - \Phi\Big(\frac{Hz/\eta - \mu}{\sigma}\Big), \tag{15}$$

where Φ denotes the cumulative distribution function (CDF) of the standard normal distribution. As H increases, the argument of Φ grows linearly, causing the tail probability to decay exponentially and significantly reducing the probability of neuron dormancy.

Proof. For the case of a single global Q-network, we define $Y = g^T x \sim \mathcal{N}(\mu, \sigma^2)$. When using an ensemble of H independent global Q-networks with the loss $L = \frac{1}{H}\sum_{k=1}^{H} L^k$, the gradient for each network is scaled by $\frac{1}{H}$. Consequently, for each network the random variable Y is replaced by $Y' = \frac{1}{H} Y$ as $\mathbb{E}[Y'] = \frac{\mathbb{E}[Y]}{H} = \frac{\mu}{H}$, and $Var(Y') = \frac{Var(Y)}{H^2} = \frac{\sigma^2}{H^2}$. Then

$$Y' = g'^T x \sim \mathcal{N}(\mu/H, \sigma^2/H^2). \tag{16}$$

A single gradient descent step updates

$$w' = w - \eta g', \tag{17}$$

where $\eta > 0$ is the learning rate. The new pre-activation becomes

$$z' = (w')^T x = (w - \eta g')^T x = z - \eta\,(g'^T x). \tag{18}$$

Then the neuron becomes inactive for this input after the update if

$$z' < 0 \iff Y' > \frac{z}{\eta}. \tag{19}$$

Therefore,

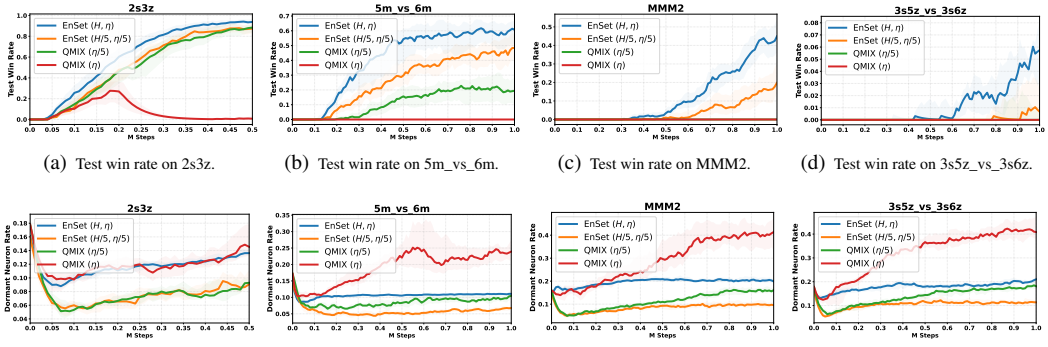
$$\mathbb{P}(z' < 0) = \mathbb{P}\Big(Y' > \frac{z}{\eta}\Big) = 1 - \mathbb{P}\Big(Y' \leq \frac{z}{\eta}\Big) = 1 - \Phi\Big(\frac{\frac{z}{\eta} - \mu/H}{\sigma/H}\Big) = 1 - \Phi\Big(\frac{Hz/\eta - \mu}{\sigma}\Big). \tag{20}$$

where Φ denotes the CDF of the standard normal distribution. □

By reducing the gradient mean and variance, the ensemble stabilizes parameter updates and reduces the probability of such sign flips, thereby mitigating neuron dormancy. Under a high replay ratio, each sampled experience is reused multiple times for gradient updates within a short time window. As a result, the same input contributes repeatedly to parameter updates, making the fixed-input analysis a reasonable local approximation of the training dynamics, especially in the early stage of learning. Additionally, the ensemble introduces model diversity (i.e., different initial network parameters) and capacity (i.e., more initial learnable neurons) beyond a single network as well as decouples global Q-network stabilization from agent network learning dynamics. On the other hand, some previous works [2] show that an ensemble helps in the high-replay-ratio setting, but they study single-agent reinforcement learning and do not connect the ensemble's benefit to the dormant neuron phenomenon, especially for the centralized global Q-network in MARL.

B.1 Comparing Learning Rate and Network Ensemble

As shown in Theorem 3.1, decreasing the learning rate also decreases the probability of neuron dormancy. Here, we compare the network ensemble and learning rate η to study how they affect the dormant neurons and the learning performance. As $H = 5$ in EnSet, we decrease η by 5 times. Results are shown in Figure 10. As we can see, a smaller learning rate decreases the dormant neuron rate of the global Q-network. However, replacing the network ensemble with a smaller learning rate hurts the performance of EnSet. This suggests that the network ensemble brings additional benefits than a smaller learning rate, such as model diversity and loss smoothing.



(a) Test win rate on 2s3z. (b) Test win rate on 5m_vs_6m. (c) Test win rate on MMM2. (d) Test win rate on 3s5z_vs_3s6z.

(e) Dormant neuron on 2s3z. (f) Dormant neuron on 5m_vs_6m. (g) Dormant neuron on MMM2. (h) Dormant neuron on 3s5z_vs_3s6z.

Figure 10: Comparing learning rate and network ensemble. Replay ratio is set at 10. The default $H = 5$ and $\eta = 0.0005$ for the global Q-value network in EnSet.

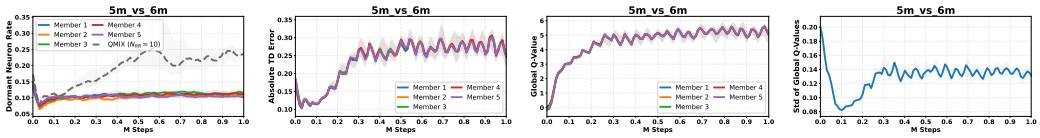
B.2 Per-Member Analysis of the Global Q-network Ensemble

The benefit of the ensemble does not require any member to be better than the others. It rests on two mechanisms. First, under the total loss L, each member receives the gradient $\frac{1}{H}\partial L(\phi^h)/\partial\phi^h$. Compared with a single network, its expected magnitude is scaled by $1/H$ and its variance by $1/H^2$ (Theorems B.1 and B.3). Each member is thus updated more gently, which lowers the probability of a pre-activation being pushed below zero (Theorem 3.1). This holds regardless of the quality difference across members. Second, each agent is updated by the average of the H gradient flows (Theorem B.2). The members start from different initializations, so their errors are partially independent even when their quality is at the same level. For example, if the gradient from each member to an agent network is the true gradient g plus an independent Gaussian noise ϵ^h, the average $g + \frac{1}{H}\sum_{h=1}^{H}\epsilon^h$ keeps g while the noise variance shrinks to $1/H$ of a single member's. Each agent thus receives a lower-variance update direction.

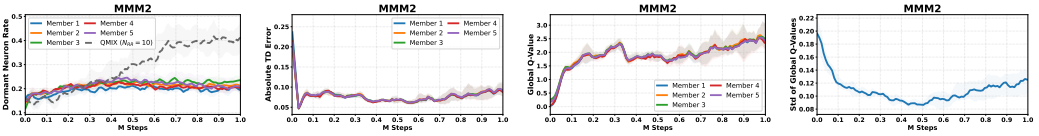
To verify the two premises, we analyze each member of EnSet-QMIX with $H = 5$ on 5m_vs_6m and MMM2 under a replay ratio of 10. Results are shown in Figure 11. The dormant neuron rates of the five members stay within a narrow band, far below that of vanilla QMIX at a replay ratio of 10. The TD errors and global Q-values of the five members almost overlap throughout training, so no member is clearly better than the others. Meanwhile, the standard deviation of the global Q-values across members stays clearly above zero, so the members do not collapse into copies of one another. The members thus reach similar quality while remaining diverse, which matches the two premises above.

C Hyperparameters of EnSet

All special hyperparameters of EnSet are the same across algorithms, tasks, and environments in this paper. The global Q-network ensemble size of EnSet H is set at 5. The translation variable α_z for multiagent translation in Equation (7) is set at 0.2. The hyperparameters of multiagent Shrink & Perturb follow MARR [30], where the interpolation factor α_r is set at 0.8 and the reset interval T_R is set at 2000. For the agent observation enhancement, we follow MARR to enable random amplitude scaling [10] on agent observation features in EnSet, where the scale variable is sampled from a



(a) Dormant neuron on 5m_vs_6m. (b) TD error on 5m_vs_6m. (c) Global Q-value on 5m_vs_6m. (d) Q-value std on 5m_vs_6m.



(e) Dormant neuron on MMM2. (f) TD error on MMM2. (g) Global Q-value on MMM2. (h) Q-value std on MMM2.

Figure 11: Per-member analysis of EnSet-QMIX with $H = 5$ under a replay ratio of 10. The dashed line in (a) and (e) is vanilla QMIX under a replay ratio of 10. (d) and (h) show the standard deviation of the global Q-values across members. Results are over 6 independent seeds, showing the median with shaded 25-75% percentiles.

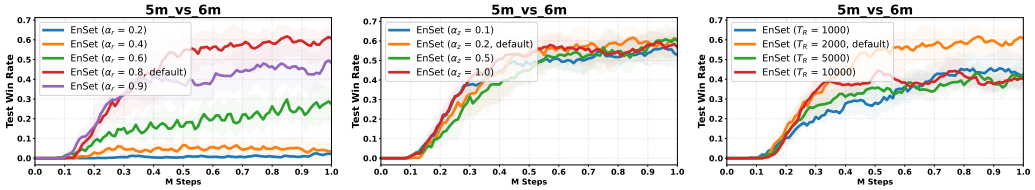
uniform distribution over a range $[0.8, 1.2]$. The special hyperparameters of EnSet are summarized in Table 1. Other hyperparameters except the replay ratio follow the standard MARL algorithms.

Table 1: Summary of special hyperparameters of EnSet.

EnSet hyperparameters	Component	Value
global Q-network ensemble size of EnSet H	Ensemble Reset	5
interpolation factor α_r	Ensemble Reset	0.8
reset interval T_R	Ensemble Reset	2000
translation variable α_z	Multiagent Translation	0.2
scale variable range	Random Amplitude Scaling	[0.8, 1.2]

C.1 Sensitivity to Hyperparameters

We further sweep the three remaining hyperparameters of EnSet on 5m_vs_6m under a replay ratio of 10, including the interpolation factor $\alpha_r \in \{0.2, 0.4, 0.6, 0.8, 0.9\}$, the translation variable $\alpha_z \in \{0.1, 0.2, 0.5, 1.0\}$, and the reset interval $T_R \in \{1000, 2000, 5000, 10000\}$. Each sweep varies one hyperparameter and keeps the others at their default values. Results are shown in Figure 12. The default $\alpha_r = 0.8$ and $T_R = 2000$ perform best. A small α_r or a short T_R resets the network too aggressively and may discard learned knowledge, which approaches the failure of ReSet in Appendix L. In particular, EnSet fails to learn with $\alpha_r = 0.2$ and 0.4. A large α_r or a long T_R resets the network too weakly, which approaches EnSet without Shrink & Perturb in Section 4.5. In contrast, the performance stays within a narrow band for α_z from 0.1 to 1.0, so a moderate translation is sufficient. The sensitivity to the ensemble size H and the replay ratio is given in Appendix H and Appendix K, respectively.



(a) Interpolation factor α_r. (b) Translation variable α_z. (c) Reset interval T_R.

Figure 12: Hyperparameter sensitivity of EnSet-QMIX on 5m_vs_6m under a replay ratio of 10. Results are over 6 independent seeds, showing the median with shaded 25-75% percentiles.

D Summary of Experimental Tasks and Computing Resources

Here, we briefly summarize the tasks in SMAC, MPE, and SMACv2 environments in Table 2.

There are 8 tasks in SMAC. For map 2s3z, each side has 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 5m_vs_6m, there are 5 allied marines against 6 marine enemies. In MMM2, there are 1 Medivac, 2 Marauders, and 7 Marines against 1 Medivac, 3 Marauders, and 8 Marines. In 3s5z_vs_3s6z, there are 3 Stalkers and 5 Zealots against 3 Stalkers and 6 Zealots. In corridor, 6 allied Zealots are against 24 Zerglings. In 6h_vs_8z, there are 6 Hydralisks against 8 Zealots. At each episode, a group of allied units is going to fight against the enemy units.

In MPE, we use a set of predator-prey tasks, where multiple slower cooperative agents must capture faster prey in a continuous two-dimensional toroidal space with obstacle landmarks. 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 its own position and velocity, the relative positions of other agents, the relative position and velocity of preys, and the relative positions of landmarks. Additionally, each agent has a view radius, which restricts the agents from receiving information about other entities (including all landmarks, other agents, and preys) that are outside the view.

In SMACv2, task scenarios are procedurally generated and require agents to generalize to previously unseen settings during evaluation. We experiment with three challenging tasks, including zerg_10_vs_10, protoss_10_vs_10, and terran_10_vs_10. In zerg_10_vs_10, there are 10 allied zerg units randomly generated to fight against 10 zerg enemy units, which are also randomly generated. In protoss_10_vs_10, 10 randomly generated protoss agents are controlled to fight against 10 protoss enemies. In terran_10_vs_10, 10 allied terran units fight against 10 terran enemy units.

Table 2: Summary of experimental tasks in SMAC, MPE, and SMACv2.

SMAC		
Task Name	**Allied Units**	**Enemy Units**
2s3z	2 Stalkers, 3 Zealots	2 Stalkers, 3 Zealots
10m_vs_11m	10 Marines	11 Marines
3s5z	3 Stalkers, 5 Zealots	3 Stalkers, 5 Zealots
5m_vs_6m	5 Marines	6 Marines
MMM2	1 Medivac, 2 Marauders, 7 Marines	1 Medivac, 3 Marauders, 8 Marines
3s5z_vs_3s6z	3 Stalkers, 5 Zealots	3 Stalkers, 6 Zealots
6h_vs_8z	6 Hydralisks	8 Zealots
corridor	6 Zealots	24 Zerglings
MPE		
Task Name	**Agents**	**Preys**
3a	3 Agents	1 Prey
6a	6 Agents	2 Preys
9a	9 Agents	3 Preys
SMACv2		
Task Name	**Allied Units**	**Enemy Units**
zerg_10_vs_10	10 Randomly Zerg Units	10 Randomly Zerg Units
protoss_10_vs_10	10 Randomly Protoss Units	10 Randomly Protoss Units
terran_10_vs_10	10 Randomly Terran Units	10 Randomly Terran Units

For the computing resources, the experiments are conducted with the NVIDIA GPUs. The version of PyTorch is 2.6.0. The operating system is Ubuntu. For a single instance of all the algorithms except ATM and EnSet-ATM, the amount of GPU memory is small, which is usually less than 1GB. For a single instance of ATM and EnSet-ATM based on transformer layers, the maximum GPU memory is about 24GB on the corridor, where a total of up to 30 units exist in this map. Here we provide the computational cost of EnSet-QMIX (ensemble size H=5) with a replay ratio of 10, the standard QMIX with a replay ratio of 1, and MARR with a replay ratio of 10 in Table 3.

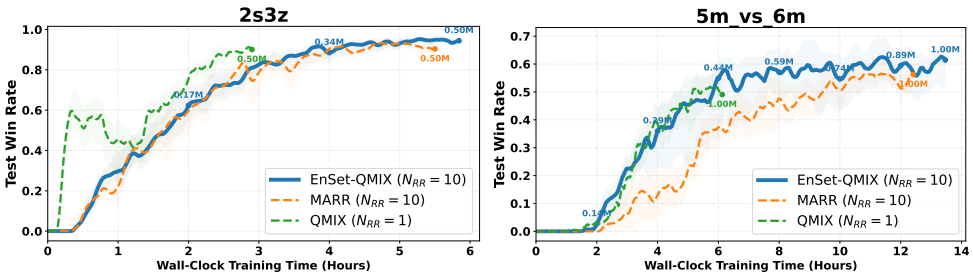
We see that, although replay ratio is improved from 1 to 10, the training time only increases roughly twice for EnSet. Using more computational time on GPU to save environment interactions is worthwhile, especially when the environment interaction or simulation is expensive.

Table 3: Training time of EnSet-QMIX, the standard QMIX, and MARR-QMIX. The unit is hour.

Scenario	EnSet-QMIX (RR=10)	QMIX (RR=1)	MARR-QMIX (RR=10)
2s3z (0.5M steps)	5.84	2.91	5.50
5m_vs_6m (1M steps)	13.47	6.13	12.42

Compared with MARR at the same replay ratio of 10, EnSet takes only slightly more training time. The main extra cost of EnSet comes from the H copies of the global Q-network, while Shrink & Perturb and the translation add negligible overhead. The cost of the centralized critic is thus H times that of the base algorithm. Each copy grows with the number of agents in the same way as the base critic does, so EnSet introduces no additional dependence on the number of agents. At execution time, only the agent networks are deployed, so the cost is identical to the base algorithm.

We further plot the performance as a function of the wall-clock training time on 2s3z and 5m_vs_6m in Figure 13, using the same runs as Table 3. The numbers on the curves mark the consumed environment steps. On the harder 5m_vs_6m, when QMIX at a replay ratio of 1 exhausts its budget of 1M environment steps, EnSet has consumed fewer than half of the environment steps yet already performs better. Compared with MARR at the same replay ratio of 10, EnSet takes nearly the same wall-clock time and performs better throughout training. On the easier 2s3z, QMIX at a replay ratio of 1 finishes earlier, while EnSet ends slightly higher, and all methods converge to a near-saturated level. The extra wall-clock time of EnSet thus buys a performance that the base algorithm cannot reach with the same budget of environment steps, and the advantage is more pronounced on the harder task.



(a) 2s3z. (b) 5m_vs_6m.

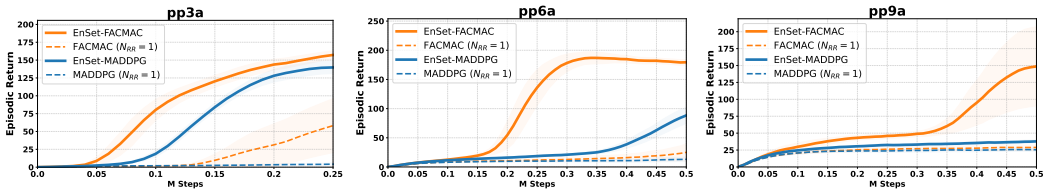
Figure 13: Test win rate as a function of the wall-clock training time. The numbers on the curves mark the consumed environment steps. Results are over 6 independent seeds, showing the median with shaded 25-75% percentiles.

E Dormant Neuron Rates in MPE

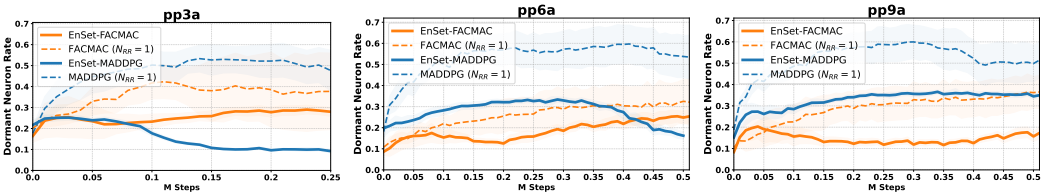
We provide the dormant neuron rates of EnSet and the standard MARL algorithms in Figure 14 in MPE. We see that, compared with the standard MADDPG and FACMAC, EnSet-based MADDPG and EnSet-based FACMAC have lower dormant neuron rates, respectively. At the same time, EnSet achieves higher performance with a replay ratio of 25 than the standard MARL algorithms with a default replay ratio of 1. This indicates EnSet has superior sample efficiency by stabilizing MARL training at high replay ratios by tackling the dormant neurons.

F Dormant Neuron Rates in SMACv2

We provide the dormant neuron rates of EnSet-QMIX and the standard QMIX in Figure 15 in SMACv2. We see that, compared with the standard QMIX, EnSet-based QMIX has lower dormant neuron rates as well as the higher performance with replay ratios of both 10 and 15.

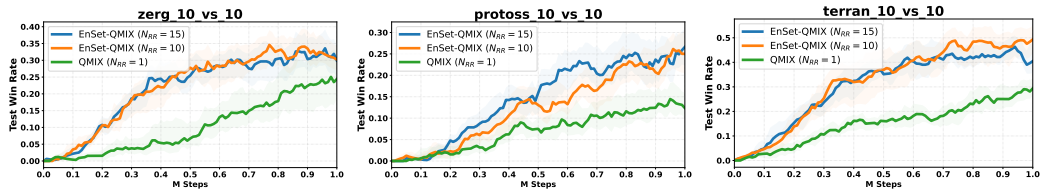


(a) Test reward on pp3a. (b) Test reward on pp6a. (c) Test reward on pp9a.

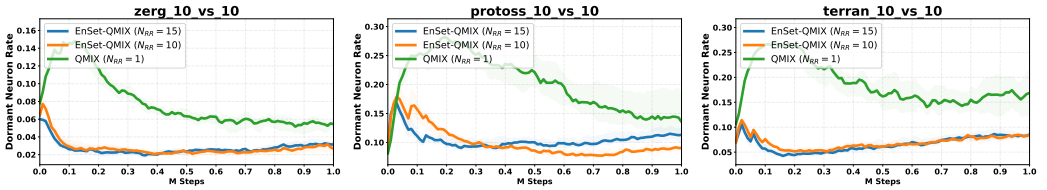


(d) Dormant neuron on pp3a. (e) Dormant neuron on pp6a. (f) Dormant neuron on pp9a.

Figure 14: EnSet boosts MARL in the MPE environment. EnSet-MADDPG and EnSet-FACMAC have a replay ratio of 25, while the standard MADDPG and FACMAC have a replay ratio of 1.



(a) zerg_10_vs_10. (b) protoss_10_vs_10. (c) terran_10_vs_10.



(d) zerg_10_vs_10. (e) protoss_10_vs_10. (f) terran_10_vs_10.

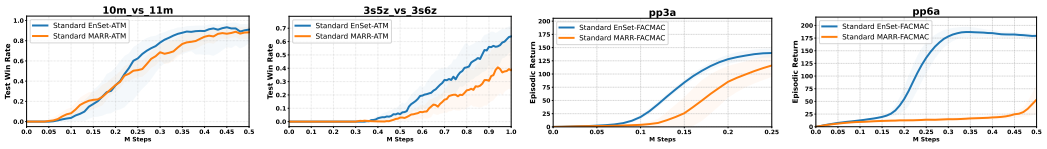
Figure 15: Benchmark experiments of EnSet for boosting MARL in SMACv2. EnSet-QMIX is trained with a replay ratio of 10 or 15, while QMIX is trained with a default replay ratio value of 1.

G More Comparisons of EnSet with MARR

In this section, we conduct an in-depth comparison with MARR using its official default settings, such as the number of environments for sampling. We first list the key differences between EnSet and MARR in Table 4. As we see, MARR employs parallel environments for scaling the replay ratio, while EnSet uses the default series environment in both SMAC and MPE. The interpolation factor α_r of 0.8 in multiagent Shrink & Perturb is the same in both EnSet and MARR, and the reset interval T_R is set at 2000 for both methods. EnSet integrates the global Q-network ensemble, while MARR does not consider it. The ensemble size of global Q-networks in EnSet is set at 5. For the local observation, we augment EnSet's agent observation with the random amplitude scaling by following MARR with the same hyperparameters. In EnSet, we apply the multiagent translation to diversify the global state s, which implies the global Q-value invariance. In contrast, the random amplitude scaling in MARR causes biases in the estimation of global Q-values. For example, increasing the allied unit health would increase the global Q-value, which departs from its true value to an overestimation. EnSet and MARR also have different replay ratios in SMAC, where EnSet uses 10 and MARR uses 50. The replay ratio is set at 25 for both EnSet and MARR in MPE.

Table 4: Summary of differences of EnSet and MARR in experiments.

Method	EnSet	MARR
Multiagent Shrink & Perturb	Yes	Yes
Global Q-network ensemble	Yes	No
Agent observation augmentation	Random amplitude scaling	Random amplitude scaling
Global state augmentation	Multiagent translation	Random amplitude scaling
Number of Environments in SMAC	1	8
Replay ratio in SMAC	10	50
Number of Environments in MPE	1	4
Replay ratio in MPE	25	25

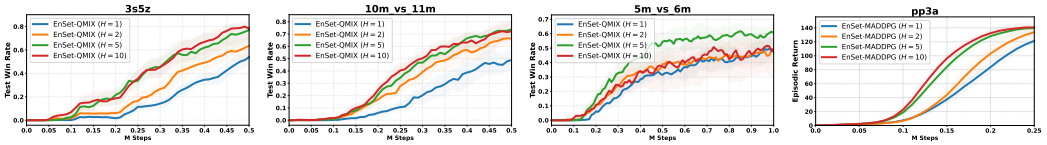


(a) Comparison on 10m_vs_11m. (b) Comparison on 3s5z_vs_3s6z. (c) Comparison on pp3a. (d) Comparison on pp6a.

Figure 16: The comparison of standard EnSet and MARR in SMAC as well as MPE.

Next, we compare EnSet and MARR with their default settings at the same environment interaction steps. We integrate EnSet and MARR into the best standard algorithms (i.e., ATM in SMAC and FACMAC in MPE). The results are shown in Figure 16. In both simple scenarios, 10m_vs_11m and pp3a, as well as difficult scenarios, 3s5z_vs_3s6z and pp6a, EnSet outperforms MARR within the same environment steps, regardless of different settings such as parallel/series environments and replay ratios. This indicates that EnSet achieves a higher sample efficiency than MARR.

H Different Global Q-network Ensemble Sizes



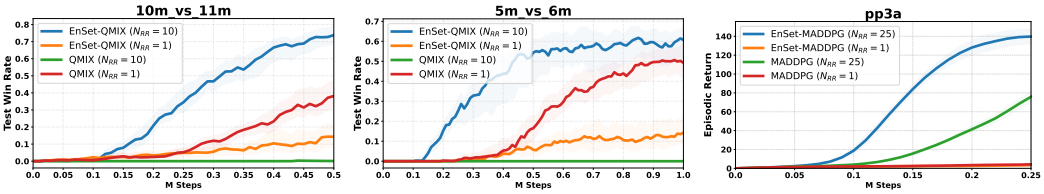
(a) 3s5z. (b) 10m_vs_11m. (c) 5m_vs_6m. (d) pp3a.

Figure 17: Experimental evaluation of the performance of different ensemble sizes in EnSet.

In this section, we study how the ensemble size of the global Q-network affects the performance of EnSet. In Figure 17, the ensemble size H of 1 performs worst in these cases, while H of 5 consistently performs well in both SMAC and MPE. At the same time, H of 10 also achieves superior performance in 3s5z and pp3a, further proving ensemble's effectiveness at high replay ratios.

I EnSet with a Standard Replay Ratio of 1

Although EnSet is specially designed for the high-replay-ratio training setting, we also wonder how EnSet performs under the standard replay ratio of 1. We provide results in Figure 18. EnSet with a replay ratio of 1 performs worse than the standard MARL algorithms with a replay ratio of 1, which is expected as network reset is employed to forget past experience, even if no severe dormant neuron phenomenon exists. On the other hand, EnSet at high replay ratios achieves a significant improvement over standard MARL algorithms with a replay ratio of 1. These results confirm that EnSet is specifically designed to address the challenge of training MARL at high replay ratios.



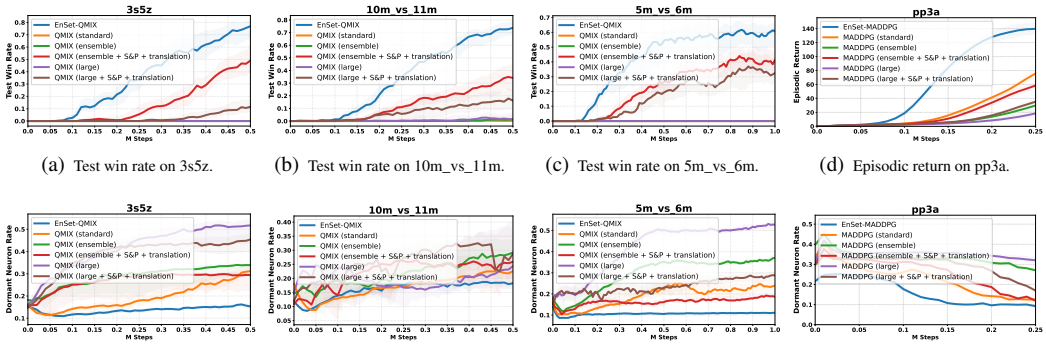
(a) 10m_vs_11m. (b) 5m_vs_6m. (c) pp3a.

Figure 18: Experimental evaluation of the performance of EnSet with a replay ratio of 1.

J Comparison of Ensemble with a Large Global Q-network

In this section, we examine whether the standard MARL algorithms with a single large global Q-network achieve effects similar to EnSet-based algorithms. We refer to the default model size of the global Q-network in the standard MARL algorithms as the standard size. At the same time, we set the model size of the global Q-network in the standard MARL algorithms to the same size as EnSet's global Q-network ensemble. We also enlarge the model size of the global Q-network to be twice the size of EnSet's global Q-network ensemble. The results of MARL algorithms with standard model size, model size the same as EnSet's global Q-network ensemble, and model size twice as EnSet's global Q-network ensemble are in Figure 19. The standard MARL algorithms with various model sizes fail to learn at high replay ratios, showing the ensemble reset's effectiveness.

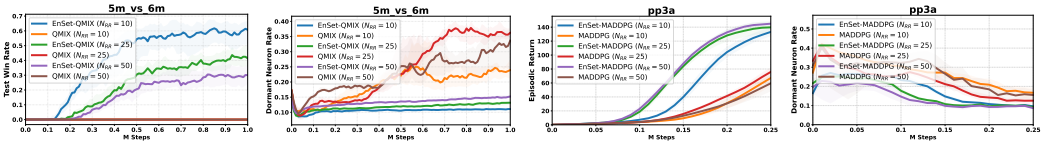
Figure 4 shows that the ensemble alone is not sufficient without Shrink & Perturb and the translation. We therefore equip the single global Q-network with the same Shrink & Perturb and translation as EnSet, for both the ensemble size and the large size. Both single networks then improve over their plain versions, but they remain far below EnSet. Since the reset and the augmentation are the same as in EnSet, the gap comes from the ensemble rather than the network capacity. Moreover, the large size performs worse than the ensemble size on all four scenarios, so a larger single network even hurts learning at high replay ratios. The dormant neuron rates are consistent with this. On 5m_vs_6m, the large single network reaches a dormant neuron rate of about 0.53. With Shrink & Perturb and the translation, its rate drops to about 0.29. In contrast, EnSet stays at about 0.11 throughout training.



(a) Test win rate on 3s5z. (b) Test win rate on 10m_vs_11m. (c) Test win rate on 5m_vs_6m. (d) Episodic return on pp3a.

(e) The dormant neuron rate of global Q-network on 3s5z. (f) The dormant neuron rate of global Q-network on 10m_vs_11m. (g) The dormant neuron rate of global Q-network on 5m_vs_6m. (h) The dormant neuron rate of global Q-network on pp3a.

Figure 19: The comparison results of EnSet and the standard MARL algorithms with a large global Q-network. 'standard' indicates the model size is the default size in the standard MARL algorithms. 'ensemble' indicates the model size equals the size of EnSet's ensembled global Q-networks. 'large' indicates the model size is twice the size of EnSet's ensembled global Q-networks. '+ S&P + translation' indicates that the single global Q-network is further equipped with the same multiagent Shrink & Perturb and translation as EnSet.



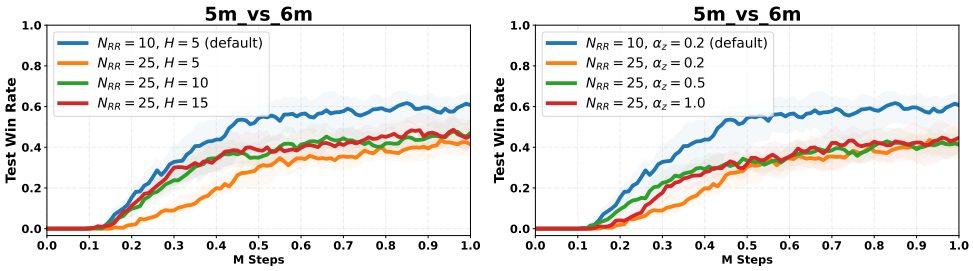
(a) Episode reward on 5m_vs_6m. (b) Dormant neurons on 5m_vs_6m. (c) Episode reward on pp3a. (d) Dormant neurons on pp3a.

Figure 20: Experimental results of EnSet under different replay ratios.

K Testing EnSet at Different Replay Ratios

In this section, we demonstrate how different replay ratios affect EnSet's performance. We experiment with EnSet using different replay ratios, such as 10, 25, and 50, for both QMIX on 5m_vs_6m and MADDPG on pp3a. The performance metrics and dormant neuron rates are shown in Figure 20. We see that high replay ratios lead to severe dormant neuron rates, and EnSet helps greatly reduce dormant neurons. On the other hand, the optimal replay ratio depends on environments and methods. A replay ratio of 10 is better than 25 and 50 in 5m_vs_6m, while replay ratios of 25 and 50 are better than 10 in pp3a. Nevertheless, EnSet always stabilizes MARL at different high replay ratios.

We further test whether a higher replay ratio requires a larger ensemble size H or a stronger translation α_z. On 5m_vs_6m at a replay ratio of 25, we vary $H \in \{5, 10, 15\}$ with $\alpha_z = 0.2$, and $\alpha_z \in \{0.2, 0.5, 1.0\}$ with $H = 5$, as shown in Figure 21. A larger H clearly speeds up learning, although it still stays below the default EnSet at a replay ratio of 10. A larger α_z brings a slight benefit at the beginning of training.



(a) Different H. (b) Different α_z.

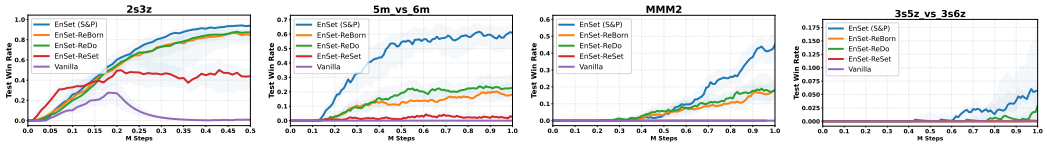
Figure 21: Test win rate of EnSet on 5m_vs_6m at a replay ratio of 25 with different H and α_z. Results are over 6 independent seeds, showing the median with shaded 25-75% percentiles.

L Introducing Other Network Reset Techniques to EnSet

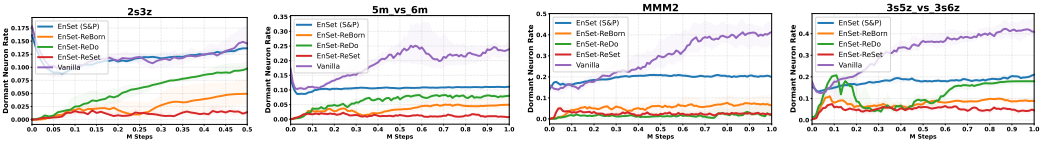
In this section, we replace the Shrink & Perturb in the standard EnSet with other MARL network reset techniques, such as ReBorn, ReDo, and ReSet. The results are shown in Figure 22. First, although all methods mitigate severe dormant neurons in the vanilla QMIX, Shrink & Perturb is the most compatible with EnSet, achieving the best performance among all reset methods. Second, although EnSet-based ReSet has low dormant neuron rates close to 0 in all maps, it performs poorly, further confirming that forcing a much lower dormant neuron rate may constrain network parameters and hurt network representation ability. These results show that the Shrink & Perturb is compatible with EnSet best among these reset techniques, which may benefit from its soft reset mechanism by balancing the network representation ability and learned knowledge [3].

M Introducing Other State Augmentation Techniques to EnSet

In this section, we replace the multiagent state translation in the standard EnSet with other MARL state augmentation techniques, such as multiagent state permutation [33] and random amplitude scaling [30]. The multiagent state permutation applies a permutation transform to unit features of agents in

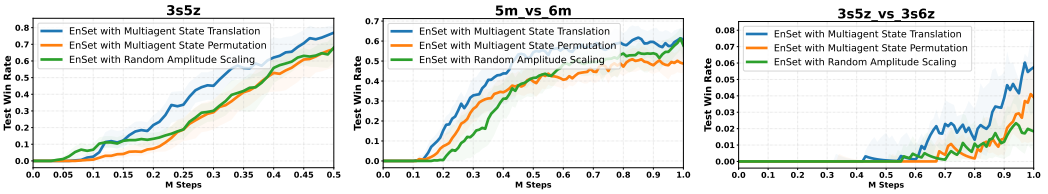


(a) Test win rate on 2s3z. (b) Test win rate on 5m_vs_6m. (c) Test win rate on MMM2. (d) Test win rate on 3s5z_vs_3s6z.



(e) Dormant neuron on 2s3z. (f) Dormant neuron on 5m_vs_6m. (g) Dormant neuron on MMM2. (h) Dormant neuron on 3s5z_vs_3s6z.

Figure 22: Introducing other network reset techniques to EnSet by replacing Shrink & Perturb (denoted as S&P) with ReBorn, ReDo, and ReSet in SMAC. Replay ratio is set at 10. The base algorithm is QMIX, and Vanilla indicates the standard QMIX with a replay ratio of 10.



(a) Test win rate on 3s5z. (b) Test win rate on 5m_vs_6m. (c) Test win rate on 3s5z_vs_3s6z.

Figure 23: Introducing other state augmentation techniques to EnSet by replacing multiagent state translation with state agent permutation and random amplitude scaling in SMAC. Replay ratio is 10.

the global state. The random amplitude scaling applies a random scale to the global state features, which is consistent across the current step and the next step. The results are shown in Figure 23. We see that multiagent state translation slightly outperforms the other two state augmentation techniques in the tested scenarios, indicating the multiagent translation invariance helps generate high-quality experiences in the high-replay-ratio setting.

N Statistical Significance Tests

We test whether the improvements of EnSet are statistically significant across seeds. All methods are trained with 6 independent seeds and the same budget of environment steps as in the main experiments. For each seed, we compute two statistics. FINAL is the mean of the last 5 test evaluations. AUC is the mean performance over the whole learning curve. We compare EnSet with the base algorithm by a two-sided Mann-Whitney U test. This non-parametric test makes no normality assumption and is robust to outlier seeds, which suits few-seed RL results. For each benchmark, we further average each seed's score over the scenarios and apply the same test. MPE returns are min-max normalized per scenario before averaging, since their scales differ across scenarios.

Table 5 shows that all aggregate comparisons are significant at the 0.05 level. Table 6 reports the complete per-scenario results. Not every per-scenario comparison is significant, for example QMIX and QPLEX on corridor, where the base algorithm fails regardless of EnSet.

Table 5: Significance tests aggregated over the scenarios of each benchmark. Each value is the median over seeds of the per-seed mean score across scenarios. MPE scores are min-max normalized per scenario. The p-values are from a two-sided Mann-Whitney U test, and bold indicates $p < 0.05$.

		FINAL			AUC		
Benchmark	Algorithm	EnSet	Base	p	EnSet	Base	p
SMAC (8 tasks)	QMIX	0.451	0.348	**0.0022**	0.258	0.187	**0.0022**
	ATM	0.707	0.488	**0.0043**	0.434	0.244	**0.0022**
	QPLEX	0.401	0.308	**0.0022**	0.227	0.187	**0.0043**
SMACv2 (3 tasks)	QMIX ($N_{RR} = 15$)	0.318	0.231	**0.0087**	0.238	0.127	**0.0043**
	QMIX ($N_{RR} = 10$)	0.345	0.231	**0.0080**	0.245	0.127	**0.0043**
MPE (3 tasks)	MADDPG	0.875	0.102	**0.0022**	0.851	0.105	**0.0022**
	FACMAC	0.894	0.223	**0.0022**	0.863	0.098	**0.0022**

Table 6: Per-scenario significance tests. Each value is the median over seeds of the test win rate in SMAC and SMACv2 and of the episode reward in MPE. The p-values are from a two-sided Mann-Whitney U test, and bold indicates $p < 0.05$. †The base algorithm has 5 seeds.

		FINAL			AUC		
Scenario	Algorithm	EnSet	Base	p	EnSet	Base	p
2s3z	QMIX	0.938	0.903	**0.0074**	0.693	0.610	**0.0152**
	ATM	0.988	0.959	0.0627	0.714	0.624	**0.0260**
	QPLEX	0.975	0.947	0.0750	0.702	0.722	0.1320
10m_vs_11m	QMIX	0.750	0.406	**0.0022**	0.402	0.157	**0.0022**
	ATM	0.894	0.309	**0.0152**	0.614	0.115	**0.0087**
	QPLEX	0.613	0.250	**0.0129**	0.338	0.077	**0.0022**
3s5z	QMIX	0.772	0.706	0.1797	0.414	0.401	1.0000
	ATM	0.938	0.822	0.1262	0.500	0.423	0.1797
	QPLEX	0.688	0.650	0.3358	0.300	0.379	0.0649
5m_vs_6m	QMIX	0.622	0.522	0.1797	0.458	0.244	**0.0087**
	ATM	0.663	0.744	0.0916	0.526	0.426	**0.0411**
	QPLEX	0.666	0.569	0.2290	0.462	0.264	0.1320
MMM2	QMIX	0.487	0.219	0.1474	0.150	0.039	0.0649
	ATM	0.747	0.566	0.2403	0.593	0.178	0.0649
	QPLEX	0.216	0.000	**0.0043**	0.046	0.003	**0.0049**
3s5z_vs_3s6z	QMIX	0.059	0.000	**0.0036**	0.018	0.001	**0.0050**
	ATM	0.641	0.056	**0.0200**	0.206	0.018	**0.0022**
	QPLEX	0.013	0.000	0.1120	0.005	0.000	0.0637
6h_vs_8z	QMIX	0.000	0.019	**0.0095**	0.001	0.005	**0.0050**
	ATM	0.225	0.091	0.1320	0.107	0.018	0.0776
	QPLEX	0.028	0.013	0.8700	0.007	0.007	1.0000
corridor	QMIX	0.000	0.000	1.0000	0.000	0.000	1.0000
	ATM	0.522	0.097	**0.0022**	0.301	0.030	**0.0022**
	QPLEX	0.000	0.000	1.0000	0.000	0.000	1.0000
zerg_10_vs_10	QMIX ($N_{RR} = 15$)	0.287	0.237	0.4673	0.234	0.110	**0.0043**
	QMIX ($N_{RR} = 10$)	0.300	0.237	0.4696	0.235	0.110	**0.0022**
protoss_10_vs_10	QMIX ($N_{RR} = 15$)	0.284	0.125	**0.0050**	0.158	0.080	**0.0022**
	QMIX ($N_{RR} = 10$)	0.281	0.125	**0.0047**	0.137	0.080	**0.0022**
terran_10_vs_10	QMIX ($N_{RR} = 15$)†	0.378	0.300	0.0530	0.319	0.166	**0.0043**
	QMIX ($N_{RR} = 10$)†	0.506	0.300	**0.0043**	0.359	0.166	**0.0043**
pp3a	MADDPG	148.6	3.3	**0.0022**	88.0	2.6	**0.0022**
	FACMAC	162.8	86.2	**0.0022**	118.4	21.2	**0.0022**
pp6a	MADDPG	104.1	11.9	**0.0022**	34.4	9.9	**0.0022**
	FACMAC	179.1	20.6	**0.0022**	122.3	12.8	**0.0022**
pp9a	MADDPG	39.4	25.2	**0.0022**	32.6	24.9	**0.0022**
	FACMAC	196.1	27.5	**0.0022**	80.4	27.6	**0.0022**

O Limitations

There are also some limitations of our work. First, EnSet is only suitable for off-policy MARL algorithms, while on-policy MARL algorithms are not tested. Second, the replay ratio is fixed during the training, which needs flexible adjustments to reduce the computational overheads.

P Broader Impacts

MARL is a powerful paradigm that can model real-world systems, such as autonomous driving, network optimization, and energy distribution. The proposed EnSet could improve the sample efficiency of MARL algorithms, thus increasing the practicality of MARL in realistic applications. However, when applied to real-world tasks, the learning process of MARL with EnSet still needs some exploration, which may lead to unsafe situations.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the abstract and introduction, we claim that we propose EnSet, combining the global Q-value network ensemble and global Q-value translation invariance, to stabilize MARL training at a high replay ratio.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We provide the limitation of our work in Appendix O.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: We provide the full set of assumptions and a complete (and correct) proof in Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We have described the tested tasks in the experimental section. We show the environments and tasks in Appendix D. We also give the detailed hyperparameters of EnSet in Appendix C. Our code is available at GitHub.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We give the detailed hyperparameters of EnSet in Appendix C. We give the descriptions of our algorithm in Algorithm 1. We believe the current information we provide in the paper is sufficient to reproduce the main experimental results. Our code is available at GitHub.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We have described our experimental setting as detailed as possible in the experimental section and the Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We use the 25-75% percentiles in the experimental results of SMAC and SMACv2 and the 95% confidence interval in the experiment results of MPE to demonstrate the statistical significance of the experiments. We further test the statistical significance of the improvements by a two-sided Mann-Whitney U test over 6 independent seeds in Appendix N.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The compute resources in our experiments are easily available, which are described in Appendix D. One NVIDIA GPU is sufficient to run the experiments of the tasks in the paper. One run usually lasts for several hours, and the time depends on the specific tasks. The training time of EnSet and the baselines is reported in Table 3, and the performance as a function of the wall-clock training time is shown in Figure 13.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have conformed with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We have provided the Broader Impacts section in Appendix P.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: Our paper poses no such risks.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We have cited all original owners of assets we used in the paper and ensured their licenses are properly respected.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release our code with a README and the Apache 2.0 license.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.