DUAL ENSEMBLED MULTIAGENT Q-LEARNING WITH HYPERNET REGULARIZER

Anonymous authors
Paper under double-blind review

ABSTRACT

Overestimation in the temporal-difference single-agent reinforcement learning has been widely studied, where the variance in value estimation causes overestimation of the maximal target value due to Jensen's inequality. Instead, overestimation in multiagent settings has received little attention though it can be even more severe. One kind of pioneer work extends ensemble methods from single-agent deep reinforcement learning to address the multiagent overestimation by discarding the large target values among the ensemble. However, its ability is limited by the ensemble diversity. Another kind of work softens the maximum operator in the Bellman equation to avoid large target values, but also leads to sub-optimal value functions. Unlike previous works, in this paper, we address the multiagent overestimation by analyzing its underlying causes in an estimation-optimization iteration manner. We show that the overestimation in multiagent value-mixing Q-learning not only comes from the overestimation of target Q-values but also accumulates in the online Q-network's optimization step. Therefore, first, we integrate the random ensemble and in-target minimization into the estimation of target Q-values to derive a lower update target. Second, we propose a novel hypernet regularizer on the learnable terms of the online global Q-network to further reduce overestimation. Experiments on various kinds of tasks demonstrate that the proposed method consistently addresses the overestimation problem while previous works fail.

1 INTRODUCTION

Overestimation is a serious challenge for reinforcement learning that stems from the maximum operation when bootstrapping the target Q-value (Thrun & Schwartz, 1993). This overestimation can be continually accumulated during learning, leading to sub-optimal policy updates and behaviors, causing instability and significantly impeding the quality of the learned policy of deep reinforcement learning (DRL) algorithms (Lan et al., 2020; Pan et al., 2021; Liang et al., 2022). A lot of representative works have been proposed to address the overestimation in single-agent DRL, including Double DQN (Hasselt et al., 2016), Averaged-DQN (Anschel et al., 2017), TD3 (Fujimoto et al., 2018), Minmax Q-learning (Lan et al., 2020), and MeanQ (Liang et al., 2022) etc. Although overestimation in single-agent DRL has been widely studied, overestimation in multiagent settings has received little attention though it can be even more severe (Pan et al., 2021; Gan et al., 2021).

To address the overestimation problem in the multiagent setting, Ackermann et al. (2019) introduce the TD3 technique to reduce the overestimation bias by using double centralized critics. Sarkar & Kalita (2021) extend the multiagent TD3 with a weighted critic update scheme to further stabilize learning. Recently, Wu et al. (2022) use an ensemble of the target multiagent Q-values to derive a lower update target by discarding the larger previously learned action values and averaging the retained ones. Besides the above ensemble methods, there are a few works focusing on the soft versions of the Bellman operator. For example, Gan et al. (2021) extend the soft Mellowmax operator into the field of multiagent reinforcement learning to tackle the overestimation. At the same time, Pan et al. (2021) use the softmax Bellman operator on the global Q-value's joint action space to avoid large target Q-values. To speed up the softmax computation on the large action space, Pan et al. (2021) approximate the softmax Bellman operator by sampling actions around the maximal joint action. Although the above works try to solve the multiagent overestimation problem by introducing the ensemble technique or softening the Bellman operator, they neglect the underlying

causes that lead to overestimation in the multiagent value-mixing Q-learning, thus are hard to fully handle the multiagent tasks where the environment contains high uncertainty such as the noise.

In this work, we detailedly analyze the overestimation in the multiagent value-mixing Q-learning, which shows that it not only comes from the overestimation of the target individual and global Q-values (Gan et al., 2021), but also accumulates in the online Q-network during the estimation-optimization iterations. To address the multiagent overestimation, we propose the **D**ual **E**nsembled **MU**ltiagent Q-learning with hypernet **RE**gularizer (DEMURE) algorithm. First, we integrate the random ensemble and in-target minimization (Chen et al., 2021) into the estimation of the target individual and global Q-values to derive a lower update target. Besides, to prevent the online global Q-network from accumulating the overestimation to an extreme during the iterative optimization, we propose a novel hypernet regularizer to regularize the hypernet weights and biases for further reducing multiagent overestimation. To validate the proposed method, we conduct the experiments on the classical multiagent particle environment and a noisy version of the StarCraft II micromanagement platform. The extensive experiments show that the proposed method consistently solves various tasks with the overestimation problem while outperforming previous methods.

2 BACKGROUND

2.1 OVERESTIMATION IN Q-LEARNING

Q-learning (Watkins & Dayan, 1992) learns the optimal value of each state-action via stochastically updating a tabular representation of Q by

$$Q(s,a) \leftarrow Q(s,a) + \alpha(r + \gamma \max_{a'} Q(s',a') - Q(s,a)) \tag{1}$$

to minimize the temporal-difference error between the estimates and the bootstrapped targets. Deep Q-Networks (DQN) (Mnih et al., 2015), which creatively combine deep learning and reinforcement learning to achieve human-level control in Atari 2600 games, learn a parametrized Q^{θ} value function by minimizing the squared error loss between Q^{θ} and the target value estimate as

$$L(s,a,r,s';\theta) = (r + \gamma \max_{a'} Q^{\bar{\theta}}(s',a') - Q^{\theta}(s,a))^2, \tag{2}$$

where $Q^{\bar{\theta}}$ is a lagging version of the current value function and is updated periodically. The max operator in the noisy target value estimate has been shown to introduce an overestimation bias (Thrun & Schwartz, 1993) due to Jensen's inequality

$$\mathbb{E}[\max_{a'} Q(s',a')] \geq \max_{a'} \mathbb{E}[Q(s',a')]. \tag{3}$$

Since the update is applied repeatedly through bootstrapping, it can iteratively increase the bias of the estimated Q-values before convergence, and introduce instability into temporal-difference learning algorithms (Liang et al., 2022). To reduce overestimation, Double DQN (Hasselt et al., 2016) is proposed to decompose the max operation in the target into action selection and action evaluation. Specifically, Double DQN evaluates the greedy policy according to the online network, while uses the target network to estimate its value. Similarly, TD3 (Fujimoto et al., 2018) proposes a clipped Double Q-learning variant for the actor-critic methods, which takes the minimum between the two critics' estimates to calculate the target Q-value. At the same time, Averaged-DQN (Anschel et al., 2017), Minmax Q-learning (Lan et al., 2020), REDQ (Chen et al., 2021) and MeanQ (Liang et al., 2022) exhibit the similar idea of using an ensemble of Q-value networks to reduce the overestimation with a lower approximation variance. On the other hand, Song et al. (2019) revisit the softmax Bellman operator and show that it can consistently outperform its max and double Q-learning counterparts on several Atari games despite its sub-optimal value function concern. Furthermore, Gan et al. (2021) propose a Soft Mellowmax operator for both the single-agent reinforcement learning and multiagent reinforcement learning, which has a provable performance guarantee while preserving the advantages of the standard Mellowmax operator (Asadi & Littman, 2017).

2.2 MULTIAGENT VALUE-MIXING Q-LEARNING

We use the Markov games as our setting, which are a multiagent extension of Markov Decision Processes (Littman, 1994). They are described by a state transition function, $T : S \times A_1 \times ... \times A_N \rightarrow$

$P(S)$, which defines the probability distribution over all possible next states, $P(S)$, given the current global state S and the action A_i produced by the i-th agent. Note that the reward is usually given based on the global state and actions of all agents $R_i : S \times A_1 \times ... \times A_N \to \mathbb{R}$. If all agents receive the same rewards, i.e. $R_1 = ... = R_N$, Markov games are fully-cooperative: a best-interest action of one agent is also a best-interest action of others (Matignon et al., 2012). The Markov games can be partially observable, in which each agent i receives a local observation $o_i : Z(S,i) \to O_i$. Thus, each agent learns a policy $\pi_i : O_i \to P(A_i)$, which maps each agent's observation to a distribution over its action set, to maximize its expected discounted returns, $J_i(\pi_i) = \mathbb{E}_{a_1 \sim \pi_1, ..., a_N \sim \pi_N, s \sim T}[\sum_{t=0}^{\infty} \gamma^t r_i(s_t, a_{1,t}, ..., a_{N,t})]$ with $\gamma \in [0,1)$ as the discounted factor. Many multiagent reinforcement learning algorithms have been proposed to solve tasks that can be modeled as partially observable Markov games.

The multiagent value-mixing algorithms represent the global Q-value as an aggregation of the individual Q-values such as VDN with the additive formation (Sunehag et al., 2018), QMIX with the non-linear formation (Rashid et al., 2018) and Qatten with the agent-wise linear formation (Yang et al., 2020). As the most representative method, QMIX performs the implicit credit assignment by learning a non-linear monotonic mixing network with the hypernetwork (Ha et al., 2017).

2.3 Overestimation in Multiagent Q-learning

In a single-agent setting, if the estimated action values contain independent noise uniformly distributed in $[-\varepsilon, \varepsilon]$ (for some $\varepsilon > 0$), we have the expected overestimation of the target value

$$\mathbb{E}[Z^s] = \mathbb{E}[r + \gamma \max_{a'} Q(s', a') - (r + \gamma \max_{a'} Q^*(s', a'))] \in [0, \gamma\varepsilon \frac{m-1}{m+1}], \tag{4}$$

where m is the action space size and Q^* is the target optimal Q-value (Thrun & Schwartz, 1993).

Overestimation in multiagent reinforcement learning is more sophisticated as it increases with the number of agents (Pan et al., 2021). Furthermore, Gan et al. (2021) show that the multiagent Q-learning algorithms with the monotonic value-mixing global Q-network such as VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018), and Qatten (Yang et al., 2020) obtain the overestimation raised by the maximization of individual Q-values with the noise over its action space.

Lemma 1. *Let Q_{tot} is a function of s and Q_i, for $i = 1, 2, ..., N$. And Q_i is a function of s and a_i, for $a_i \in A_i$. Assuming $l \leq \frac{\partial Q_{tot}}{\partial Q_i} \leq L, i = 1, 2, ..., N$ where $l \geq 0$ and $L > 0$, and $Q_i(s, a_i)$ is with an independent noise uniformly distributed in $[-\varepsilon, \varepsilon]$ on each action a_i given a state s. Then*

$$lN\mathbb{E}[Z_i^s] \geq \mathbb{E}[r + \gamma \max_{\mathbf{a}} Q_{tot}(s, \mathbf{Q_i}) - (r + \gamma \max_{\mathbf{a}} Q_{tot}(s, \mathbf{Q_i^*}))] \geq lN\mathbb{E}[Z_i^s], \tag{5}$$

where $\mathbb{E}[Z_i^s] = \mathbb{E}[\max_{a_i} Q_i(s, a_i) - \max_{a_i} Q_i^*(s, a_i)]$ and $\mathbf{Q_i^*}$ are the target optimal values.

The proof is provided in Appendix A. As Eq. (5) shows, when computing the target global Q-value, the overestimation is raised by the maximization of Q_i over its action space. However, previous works ignore the fact that the overestimation can also accumulate throughout the estimation-optimization iterations of Q_{tot}, and thus cannot fully tackle the multiagent overestimation problem. Next, we analyze the underlying causes of multiagent value-mixing Q-learning overestimation throughout the Q_{tot}'s estimation-optimization iterations and derive our method correspondingly.

3 DEMURE for Multiagent Overestimation

3.1 Multiagent Overestimation in Estimation-Optimization Iterations

Recall that the global Q-network approximates the computed target global Q-value. First, Eq. (5) indicates that the computed target Q_{tot} and Q_is have the overestimation raised by the maximization of Q_i over its action space. Once the overestimated target Q_{tot} is computed, the online network of the global Q-value approximates the target Q_{tot} value. Here we expand the formation of Q_{tot} as

$$Q_{tot}(s, Q_1, ..., Q_N) = f_{mix}(s, Q_1, ..., Q_N) + c(s), \tag{6}$$

where f_{mix} is the monotonic mixing function that $\frac{\partial Q_{tot}}{\partial Q_i} \geq 0$.

We analyze each term's behavior of the online global Q-value network in the optimization step. Surprisingly, we found that the term $\frac{\partial Q_{tot}}{\partial Q_i}$ exacerbates the overestimation of the target Q_{tot} by a squared multiple after the online Q_{tot} network's optimization.

Theorem 1. *Assuming the learned Q_{tot} network approximates the optimal function $Q^*_{tot}(\cdot)$ where $Q_{tot}(\cdot) = Q^*_{tot}(\cdot)$ and the learned Q_i network approximates the optimal function $Q^*_i(\cdot)$ where $Q_i(\cdot) = Q^*_i(\cdot)$ with $l \leq \frac{\partial Q_{tot}}{\partial Q_i} \leq L, i = 1, 2, ..., N$ where $l \geq 0$ and $L > 0$, then estimated target value y_{tot} will be biased from the optimal value y^*_{tot} with $\Delta y > 0$ given the individual Q-value is with an independent noise uniformly distributed in $[-\varepsilon, \varepsilon]$ on each action. If we continue to train the online Q_i network by the L_2 norm $L_{mix} = \|y_{tot} - Q_{tot}\|^2$, the updated network $\hat{Q}_i$ will be biased from optimal Q^*_i as*

$$\hat{Q}_i = Q^*_i + \Delta Q_i, \tag{7}$$

where $\Delta Q_i \geq 2\alpha l \Delta y$ for some $\alpha > 0$. After the feedforward even without considering the updated Q_{tot} network, the bias of the new Q_{tot} value $\hat{Q}_{tot}$ from the optimal value Q^*_{tot} will become

$$\hat{Q}_{tot} \geq Q^*_{tot} + 2\alpha N l^2 \Delta y. \tag{8}$$

Proof. We apply the gradient method to minimize L_{mix}, the independent Q_i is updated as follows,

$$\begin{aligned}
\hat{Q}_i &= Q^*_i - \alpha \frac{\partial (y_{tot} - Q_{tot}(s, Q_1, \ldots, Q_N))^2}{\partial Q_i} \\
&= Q^*_i - \alpha \frac{\partial (y^*_{tot} + \Delta y - Q_{tot}(s, Q_1, \ldots, Q_N))^2}{\partial Q_i} \\
&= Q^*_i + 2\alpha (y^*_{tot} + \Delta y - Q_{tot}(s, Q_1, \ldots, Q_N)) \frac{\partial Q_{tot}(s, Q_1, \ldots, Q_N)}{\partial Q_i}.
\end{aligned} \tag{9}$$

Compared with updating with the ground-truth target global Q-value y^*_{tot}, Q_i is biased with $\Delta Q_i \geq 2\alpha l \Delta y$. Such bias ΔQ_i will be propagated through the global Q-network's feedforward process to increase the overestimation of the new global Q-value $\hat{Q}_{tot}$ as

$$\begin{aligned}
\hat{Q}_{tot} &= Q_{tot}(s, \hat{Q}_1, \ldots, \hat{Q}_N) = Q_{tot}(s, Q_1 + \Delta Q_1, \ldots, Q_N + \Delta Q_N) \\
&\approx Q_{tot}(s, Q_1, \ldots, Q_N) + \sum_{i=1}^{N} \frac{\partial Q_{tot}(s, Q_1, \ldots, Q_N)}{\partial Q_i} \Delta Q_i \\
&= Q_{tot}(s, Q_1, \ldots, Q_N) + 2\alpha \Delta y \sum_{i=1}^{N} \left(\frac{\partial Q_{tot}(s, Q_1, \ldots, Q_N)}{\partial Q_i}\right)^2 \\
&= Q^*_{tot} + 2\alpha \Delta y \sum_{i=1}^{N} \left(\frac{\partial Q_{tot}(s, Q_1, \ldots, Q_N)}{\partial Q_i}\right)^2 \geq Q^*_{tot} + 2\alpha N l^2 \Delta y,
\end{aligned} \tag{10}$$

where the second approximation comes from Taylor expansion. Thus, Q_i's overestimation causes more severe overestimation for Q_{tot} even if we only update the online Q_i network in one optimization step. Furthermore, in the repeated estimation-optimization iterations, such bias can accumulate. □

We could find that, when forwarding the updated biased Q_i to compute the new global Q-value, the term $\frac{\partial Q_{tot}}{\partial Q_i}$ exacerbates the overestimation of the Q_{tot} value by a squared multiple. Moreover, Pan et al. (2021) empirically show that $\frac{\partial Q_{tot}}{\partial Q_i}$ continually increases while learning, which makes the overestimation of Q_{tot} much more severe. Based on the above reasons, we are motivated to regularize $\frac{\partial Q_{tot}}{\partial Q_i}$ to prevent the severe overestimation of Q_{tot}. Meanwhile, another learnable term $c(s)$ in Eq. 6 can also accumulate overestimation when updating network parameters in the iterative estimation-optimization loop, which is further demonstrated in the experiments such as in Figure 6(d). Thus, we also regularize the term $c(s)$ in case that $c(s)$ increases extremely large to cause Q_{tot}'s severe overestimation. In summary, we are motivated to control the overestimated Q_{tot} and Q_i in the target Q_{tot} value's estimation, as well as to regularize $\frac{\partial Q_{tot}}{\partial Q_i}$ and $c(s)$ in the online Q_{tot} network's optimization.

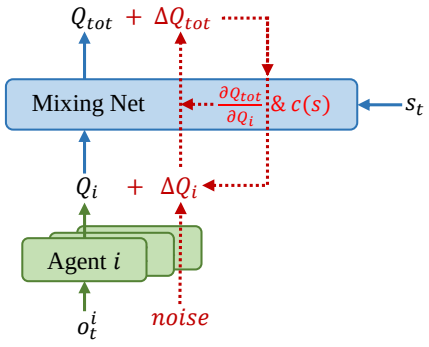


Figure 1: Overestimation analysis.

3.2 DUAL ENSEMBLED MULTIAGENT Q-LEARNING

Using ensemble to reduce the overestimation in Q-learning is widely studied in single-agent DRL (Anschel et al., 2017; Peer et al., 2021; Lee et al., 2021; Chen et al., 2021; Liang et al., 2022). The statistics of the ensemble can be used to assess model uncertainty or to produce a lower-variance estimate compared to a single estimator (Liang et al., 2022). These benefits have been utilized in DRL for alleviating estimation bias by pessimistically estimating the target as the minimum over ensemble predictions (Chen et al., 2021; Liang et al., 2022).

In this paper, we integrate REDQ (Chen et al., 2021), a state-of-the-art ensemble method in the single-agent DRL domain, into the multiagent value-mixing Q-learning algorithms. REDQ uses an in-target minimization across a random subset of Q-functions from the ensemble of Q-networks to control the overestimation. We incorporate this idea into the computation of the target Q-values to address the overestimated target Q_i and Q_{tot}. Next, we explain the details of the dual ensemble.

Algorithm 1 Dual Ensembled Multiagent Q-learning

1: Initialize individual Q-value network parameters $\theta_{1,1},...,\theta_{1,K},...,\theta_{N,1},...,\theta_{N,K}$, global Q-value network parameters $\phi_1,\phi_2,...,\phi_H$ and empty replay buffer D. Set target parameters $\bar{\theta}_{i,k} \leftarrow \theta_{i,k}$ for $i=1,2,...,N$, $k=1,2,...,K$ and $\bar{\phi}_h \leftarrow \phi_h$ for $h=1,2,...,H$.
2: **for** Episode $1,2,3...$ **do**
3: Each agent i takes action $a_{i,t} \sim \pi_{\theta_i}(\cdot|o_{i,t})$. Step into new state s_{t+1}. Receive new reward r_t and new observation $o_{i,t+1}$.
4: Add data to buffer: $D \leftarrow D \cup \{(s_t,\mathbf{o}_t,\mathbf{a}_t,r_t,s_{t+1},\mathbf{o}_{t+1})\}$.
5: Sample a mini-batch $B=\{(s,\mathbf{o},\mathbf{a}_t,r,s',\mathbf{o}')\}$ from D.
6: Sample a set $\mathbb{K}$ of N_K distinct indices from $\{1,2,...,K\}$.
7: Sample a set $\mathbb{H}$ of N_H distinct indices from $\{1,2,...,H\}$.
8: Compute the Q target y_{tot} (same for all of the N critics):

$$y_{tot} = r+\gamma(\min_{h\in\mathbb{H}} Q_{tot}^{\bar{\phi}_h}(s',Q_1(o'_1,a'_1),...,Q_N(o'_N,a'_N)),$$

$$Q_i(o'_i,a'_i) = \max_{a'_i}\min_{k\in\mathbb{K}} Q_i^{\bar{\theta}_{i,k}}(o'_i,a'_i).$$

9: **for** $h=1,...,H$ **do**
10: Update $\phi_h,\theta_{1,1},...,\theta_{N,K}$ with gradient descent

$$\nabla_{\phi_h,\theta_{1,1},...,\theta_{N,K}} L_{mix}(\phi_h,\theta_{1,1},...,\theta_{N,K}), \text{ where}$$

$$L_{mix}(\phi_h,\theta_{1,1},...,\theta_{N,K}) = \frac{1}{|B|}\sum_{(s,\mathbf{o},\mathbf{a},r,s',\mathbf{o}')\in B}[Q_{tot}^{\phi_h}(s,Q_1(o_1,a_1),...,Q_N(o_N,a_N))-y_{tot}]^2, \text{ and}$$

$$Q_i(o_i,a_i) = mean_{k=1}^{K} Q_i^{\theta_{i,k}}(o_i,a_i).$$

11: **end for**
12: Update target networks $\bar{\phi}_1 \leftarrow \phi_1$, ..., $\bar{\phi}_H \leftarrow \phi_H$ and $\bar{\theta}_{1,1} \leftarrow \theta_{1,1}$, ..., $\bar{\theta}_{N,K} \leftarrow \theta_{N,K}$ every C times.
13: **end for**

When computing the target global Q-value, we have

$$\begin{aligned} y_{tot} &= r+\gamma\max_{\mathbf{a}'} Q_{tot}^{\bar{\phi}}(\mathbf{Q_i})(s',\mathbf{a}') \\ &= r+\gamma Q_{tot}^{\bar{\phi}}(\max_{a'_1} Q_1^{\bar{\theta}_1},...,\max_{a'_N} Q_N^{\bar{\theta}_N})(s',\mathbf{a}') \\ &= r+\gamma Q_{tot}^{\bar{\phi}}(s',\max_{a'_1} Q_1^{\bar{\theta}_1}(o'_1,a'_1),...,\max_{a'_N} Q_N^{\bar{\theta}_N}(o'_N,a'_N)), \end{aligned}$$

where $\bar{\phi}$ and $\bar{\theta}$ are the parameters of the target global Q-value network and target individual Q-value networks respectively. As shown in Eq. (5), the overestimation in the target Q_i causes the overestimation of the target Q_{tot}. Therefore, we first apply the random in-target minimization technique of

REDQ to the target Q_i's estimation. Then the target individual Q-value is computed as

$$Q_i = \min_{k \in \mathbb{K}} Q_i^{\bar{\theta}_{i,k}}, \tag{11}$$

where $\mathbb{K}$ is a subset with size N_K randomly sampled from $\{1,2,...,K\}$ and $\bar{\theta}_{i,k}$ is agent i's target individual Q-network. Similarly, the target global Q-values estimation is also computed with the random in-target minimization technique to further reduce the multiagent overestimation as

$$Q_{tot}^{\bar{\phi}} = \min_{h \in \mathbb{H}} Q_{tot}^{\bar{\phi}_h}, \tag{12}$$

where $\mathbb{H}$ is a subset with size N_H randomly sampled from $\{1,2,...,H\}$ and $\bar{\phi}_h$ is the target centralized global Q-network. As shown in Theorem 1 of REDQ (Chen et al., 2021), by changing the subset size of the random minimization, we are able to control the overestimation of target Q_{tot} and Q_i.

The dual ensembled multiagent Q-learning algorithm is shown in Algorithm 1. **Line 1** initializes the empty replay buffer, the online and target individual Q-value networks for each agent, as well as the online and target global Q-value networks. **Line 3-4** interact with the environment based on agent policies and store the transition into the replay buffer. **Line 5** samples a mini-batch of transitions from the replay buffer for updating. **Line 6-7** samples the indices for selecting instances from Q_i network ensemble and Q_{tot} network ensemble respectively. **Line 8** computes the target Q_{tot} value with in-target minimization across the selected network subsets. **Line 9-11** update all online Q_i and Q_{tot} networks with respect to the loss function L_{mix}. **Line 12** updates the network parameters of all target individual and global Q-value networks by copying from their online versions periodically.

3.3 HYPERNET REGULARIZER

The analysis and empirical findings indicate that $\frac{\partial Q_{tot}}{\partial Q_i}$ and c can accumulate and exacerbate the multiagent overestimation in the estimation-optimization iterations. To tackle this issue, we propose a novel hypernet regularizer to restrict the learnable terms in the online global Q-network. Specifically, we propose to use the L1 regularization to restrict $\frac{\partial Q_{tot}}{\partial Q_i}$ and c as

$$L_{hyper} = \sum \mathbf{W} + \sum |\mathbf{B}| + |c|, \tag{13}$$

where $\mathbf{W} \geq 0$ and $\mathbf{B}$ are the hypernet weights and biases from the hypernets respectively. The term c is also from a hypernet. The proof is shown in Appendix C.

Then the final loss function for the proposed DEMURE becomes

$$L = L_{mix} + \alpha_{hyper} \frac{1}{|B|} \frac{1}{H} \sum_{(s,\mathbf{o},\mathbf{a},r,s',\mathbf{o}') \in B} \sum_{h=1}^{H} L_{hyper}(s|\phi_h), \tag{14}$$

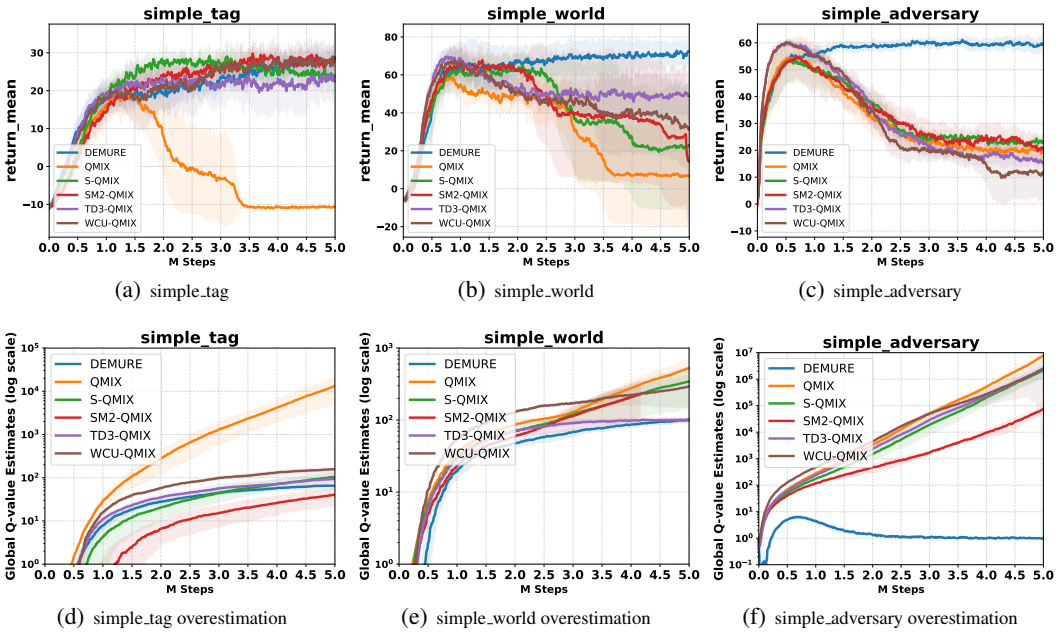
where α_{hyper} is the coefficient of the hypernet regularization loss.

4 EXPERIMENTS

In this section, we conduct the experiments in the multiagent particle environment (MPE) (Lowe et al., 2017) and in the noisy version of the StarCraft multiagent challenge (SMAC) (Samvelyan et al., 2019) environment to validate the effectiveness of DEMURE when facing the multiagent overestimation problem compared with various baselines. We also conduct the ablation study and analysis to validate the effectiveness of dual ensemble and hypernet regularizer techniques.

4.1 EXPERIMENTS ON MPE

We first conduct the experiments in the multiagent particle environment including *simple_tag*, *simple_world*, and *simple_adversary*. The *simple_tag* is a predator-prey task where 3 slower predators coordinate to capture a faster prey. The *simple_world* involves 4 slower agents to catch 2 faster adversaries that desire to eat food. The *simple_adversary* involves 2 cooperating agents and 1 adversary. The agents need to reach a single target landmark from a total of two landmarks while the adversary is unaware of the target. We follow the setting in RES (Pan et al., 2021) that these tasks



(a) simple_tag (b) simple_world (c) simple_adversary

(d) simple_tag overestimation (e) simple_world overestimation (f) simple_adversary overestimation

Figure 2: Results on different Multiagent Particle Environment scenarios.

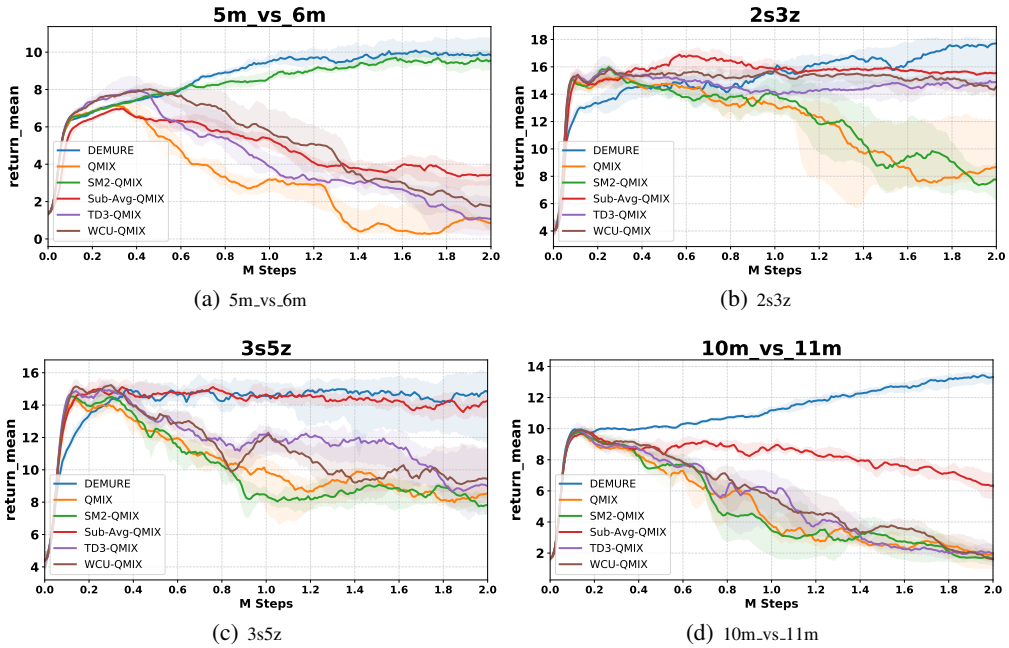
are fully cooperative and the adversaries are pre-trained by MADDPG (Lowe et al., 2017) for 10^4 episodes. The training and evaluation settings are also kept unchanged.

We compare the proposed method with QMIX (Rashid et al., 2018), S-QMIX (Pan et al., 2021) (For a fair comparison, we use RES without the N-step return regularization and focus on the one-step temporal-difference technique.), SM2-QMIX (Gan et al., 2021), TD3-QMIX (Ackermann et al., 2019) and WCU-QMIX (Sarkar & Kalita, 2021). We implement the proposed method based on the pymarl framework (Samvelyan et al., 2019), and the detailed implementation of each baseline is provided in Appendix B. For each method, we run 5 independent trails with different random seeds and the resulting plots include the mean performance as well as the shaded 95% confidence interval. Besides, we plot the estimated Q_{tot} of each method in the log scale to show the overestimation status.

As shown in Figure 2, the proposed method learns stably and achieves superior performance on all three tasks. At the same time, QMIX fails on all tasks as it gets the most severe overestimation. While some baselines such as TD3-QMIX and WCU-QMIX could control the overestimation in the *simple_tag* and *simple_world*, they cannot tackle the *simple_adversary* scenario where the overestimation on this task is much severer than the other two tasks. Meanwhile, although the soft Bellman operator (S-QMIX) and soft Mellowmax operator (SM2-QMIX) could tackle the *simple_tag* task, they fail on the two other tasks. Overall, the proposed method demonstrates its capability to address the multiagent overestimation problem and stabilizes the learning process.

4.2 Experiments on Noise SMAC

Then we conduct the experiments on the well-known StarCraft II platform, which has become a commonly-used benchmark for evaluating state-of-the-art multiagent reinforcement learning algorithms. Here we use a noisy version of SMAC, where the noise is added into the sensors of each agent's observation and the global state (see Appendix for the detailed noise setting). The noise increases the variances of the individual Q-value and global Q-value, and thus raises the overestimation problem for the multiagent Q-learning algorithms. We conduct the experiment on 4 tasks including *5m_vs_6m*, *2s3z*, *3s5z*, and *10m_vs_11m*. In *5m_vs_6m*, there are 5 allied marines against 6 marine enemies. For map *2s3z*, both sides have 2 Stalkers and 3 Zealots. For map *3s5z*, both sides have 3 Stalkers and 5 Zealots. In *10m_vs_11m*, there are 10 allied marines against 11 ma-



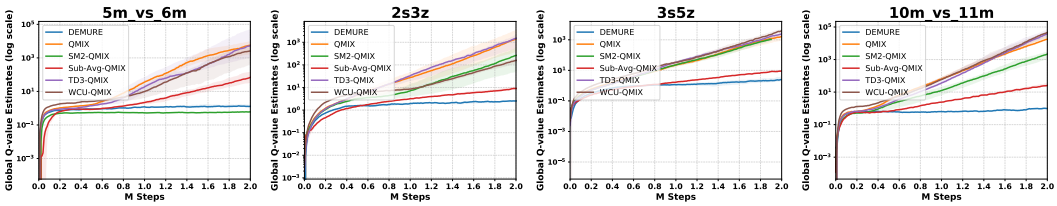
(a) 5m_vs_6m (b) 2s3z

(c) 3s5z (d) 10m_vs_11m

Figure 3: Results on different StarCraft Multi-Agent Challenge scenarios.

rine enemies. Training and evaluation schedules such as the testing episode number and training hyperparameters are kept unchanged (Samvelyan et al., 2019). The version of StarCraft II is 4.6.2.

We compare the proposed method with QMIX, SM2-QMIX, TD3-QMIX, WCU-QMIX and Sub-Avg-QMIX (Wu et al., 2022). Here we omit S-QMIX as it collapses while training. The reason is that its softmax implementation does not well support the action space with varying legal actions at each timestep. Instead, we add a new baseline Sub-Avg-QMIX as it exhibits excellent performance in the standard SMAC tasks (Wu et al., 2022). Results are averaged over 6 independent training trails with different random seeds, and the resulting plots include the median performance as well as the shaded 25-75% percentiles. We also report the estimated Q_{tot}. As shown in Figure 3 and Figure 4, in the noisy setting, QMIX suffers from severe overestimation and cannot learn stably in the noisy environment. At the same time, in these four maps, the proposed method successfully addresses the overestimation to stabilize learning while other baselines cannot consistently tackle all the tasks.



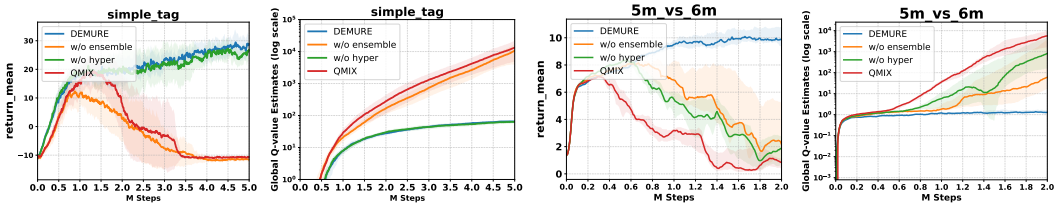
(a) Q_{tot} on 5m_vs_6m (b) Q_{tot} on 2s3z (c) Q_{tot} on 3s5z (d) Q_{tot} on 10m_vs_11m

Figure 4: Overestimation on different StarCraft Multi-Agent Challenge scenarios.

4.3 Ablation Study

In this section, we perform the ablation study of each technique of our method. We compare DEMURE, DEMURE without the ensemble (w/o ensemble), and DEMURE without the hypernet regularizer (w/o hyper). If DEMURE is without both techniques, it degenerates to vanilla QMIX. Figure 5 shows the results. As we can see, each component is essential for addressing the multiagent overestimation problem. Especially, in *5m_vs_6m*, neither the dual ensemble nor the hypernet

regularizer addresses the overestimation alone. Only the combined techniques, which jointly address the underlying causes, can avoid the overestimation and successfully stabilize learning. The ablation study validates the importance of each technique of the proposed method.

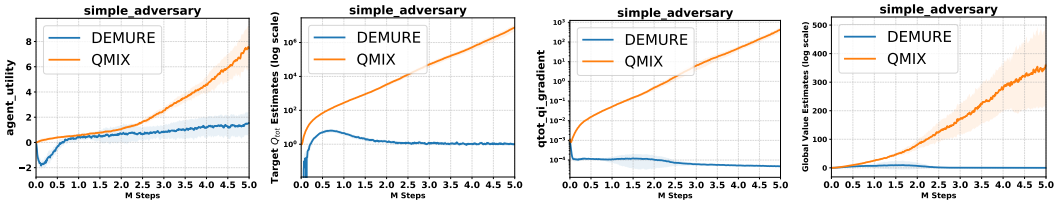


(a) simple_tag ablation (b) Q_{tot} on simple_tag ablation (c) 5m_vs_6m ablation (d) Q_{tot} on 5m_vs_6m ablation

Figure 5: Results of the ablation study.

4.4 Experimental Analysis of Overestimation's Causes

As the proposed method is designed to control each overestimation cause, we report Q_i and Q_{tot} in the target Q-value estimation as well as $\frac{\partial Q_{tot}}{\partial Q_i}$ and c in the online Q-network optimization to visually show how it works. Here we run the *simple_adversary* task as its overestimation is the most severe among all tasks and the results are shown in Figure 6(a)-6(d). We could see that each term in QMIX has higher values than the proposed method. Especially, $\frac{\partial Q_{tot}}{\partial Q_i}$ contributes much to the overestimation problem of the vanilla QMIX as the value of $\frac{\partial Q_{tot}}{\partial Q_i}$ is much larger than the proposed method by orders of magnitude. Differently, the proposed method successfully prevents Q_i, $\frac{\partial Q_{tot}}{\partial Q_i}$, c and thus Q_{tot} from becoming extreme large to avoid the multiagent overestimation problem.



(a) Q_i on simple_adversary (b) Q_{tot} on simple_adversary (c) $\frac{\partial Q_{tot}}{\partial Q_i}$ on simple_adversary (d) c on simple_adversary

Figure 6: Experimental analysis of each cause.

5 Conclusion

In this paper, we propose DEMURE to address the multiagent overestimation problem. The analysis of the multiagent value-mixing Q-learning shows that the overestimation falls into the overestimated target Q_i and Q_{tot} in the target Q_{tot} value's estimation, as well as the learnable terms in the online Q_{tot} network's optimization. Motivated by this analysis, we propose to utilize the dual ensemble and hypernet regularizer to address these underlying causes that previous works neglect. Extensive experiments show that the proposed method successfully addresses the multiagent overestimation and achieves superior performance on the MPE and the noisy version of SMAC environments when compared with various baselines. Furthermore, the ablation study and experimental analysis demonstrate the effectiveness and importance of each technique of the proposed method.

For future work, on the one hand, it is potential to further scale the proposed method into realistic large-scale multiagent settings where noise is common and learning stability is required for tasks. On the other hand, reducing the network parameters of the ensemble model while keeping the ensemble diversity could also bring a lot of benefits. To achieve it, our method is promising to be combined with advanced techniques in deep learning such as the dropout and network pruning.

REFERENCES

Johannes Ackermann, Volker Gabler, Takayuki Osa, and Masashi Sugiyama. Reducing Overestimation Bias in Multi-Agent Domains Using Double Centralized Critics. In *Proceedings of NeurIPS Deep RL Workshop*, 2019.

Oron Anschel, Nir Baram, and Nahum Shimkin. Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 176–185, 2017.

Kavosh Asadi and Michael L. Littman. An Alternative Softmax Operator for Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 243–252, 2017.

Gleb Beliakov, Humberto Bustince Sola, and Tomasa Calvo Snchez. *A Practical Guide to Averaging Functions*. Springer Publishing Company, Incorporated, 2015.

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.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). In *Proceedings of the 4th International Conference on Learning Representations*, 2016.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 1587–1596, 2018.

Yaozhong Gan, Zhe Zhang, and Xiaoyang Tan. Stabilizing Q Learning Via Soft Mellowmax Operator. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 7501–7509, 2021.

David Ha, Andrew M. Dai, and Quoc V. Le. HyperNetworks. In *International Conference on Learning Representations*, 2017.

Hado van Hasselt, Arthur Guez, and David Silver. Deep Reinforcement Learning with Double Q-Learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pp. 2094–2100. AAAI Press, 2016.

Qingfeng Lan, Yangchen Pan, Alona Fyshe, and Martha White. Maxmin Q-learning: Controlling the Estimation Bias of Q-learning. In *International Conference on Learning Representations*, 2020.

Kimin Lee, Michael Laskin, Aravind Srinivas, and Pieter Abbeel. SUNRISE: A Simple Unified Framework for Ensemble Learning in Deep Reinforcement Learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 6131–6141, 2021.

Litian Liang, Yaosheng Xu, Stephen Mcaleer, Dailin Hu, Alexander Ihler, Pieter Abbeel, and Roy Fox. Reducing Variance in Temporal-Difference Value Estimation via Ensemble of Deep Networks. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pp. 13285–13301, 2022.

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

Ryan Lowe, YI WU, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Proceedings of the 31th Advances in Neural Information Processing Systems*, pp. 6379–6390, 2017.

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.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Ling Pan, Tabish Rashid, Bei Peng, Longbo Huang, and Shimon Whiteson. Regularized Softmax Deep Multi-Agent Q-Learning. In *Advances in Neural Information Processing Systems*, 2021.

Oren Peer, Chen Tessler, Nadav Merlis, and Ron Meir. Ensemble Bootstrapping for Q-Learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139, pp. 8454–8463, 2021.

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*, pp. 4292–4301, 2018.

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*, pp. 2186–2188, 2019.

Tamal Sarkar and Shobhanjana Kalita. A Weighted Critic Update Approach to Multi Agent Twin Delayed Deep Deterministic Algorithm. In *2021 IEEE 18th India Council International Conference*, pp. 1–6, 2021.

Zhao Song, Ron Parr, and Lawrence Carin. Revisiting the Softmax Bellman Operator: New Benefits and New Perspective. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 5916–5925, 2019.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2085–2087, 2018.

Sebastian Thrun and A. Schwartz. Issues in using function approximation for reinforcement learning. In *Proceedings of 4th Connectionist Models Summer School*, 1993.

Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

Haolin Wu, Jianwei Zhang, Zhuang Wang, Yi Lin, and Hui Li. Sub-AVG: Overestimation reduction for cooperative multi-agent reinforcement learning. *Neurocomputing*, 474:94–106, 2022.

Yaodong Yang, Jianye Hao, Ben Liao, Kun Shao, Guangyong Chen, Wulong Liu, and Hongyao Tang. Qatten: A General Framework for Cooperative Multiagent Reinforcement Learning. *CoRR*, abs/2002.03939, 2020.

A PROOF OF OVERESTIMATION IN VALUE-MIXING Q-LEARNING

Gan et al. (2021) show that if the assumption that $l \leq \frac{\partial Q_{tot}}{\partial Q_i} \leq L, i = 1, 2, ..., N$ where $l \geq 0$ and $L > 0$ satisfies, multiagent Q-learning algorithms with the monotonic value-mixing global Q-network such as VDN (Sunehag et al., 2018), QMIX (Rashid et al., 2018), and Qatten (Yang et al., 2020) obtain the overestimation

$$LN\mathbb{E}[Z_i^s] \geq \mathbb{E}[r + \gamma \max_{\mathbf{a}} Q_{tot}(s, \mathbf{Q_i}) - (r + \gamma \max_{\mathbf{a}} Q_{tot}(s, \mathbf{Q_i^*}))] \geq lN\mathbb{E}[Z_i^s], \tag{15}$$

where $\mathbb{E}[Z_i^s] = \mathbb{E}[\max_{a_i} Q_i(s, a_i) - \max_{a_i} Q_i^*(s, a_i)]$ and $\mathbf{Q_i^*}$ are the target values.

Proof.

$$\begin{aligned}
&\mathbb{E}[r + \gamma \max_{\mathbf{a}} Q_{tot}(s, \mathbf{Q_i}) - (r + \gamma \max_{\mathbf{a}} Q_{tot}(s, \mathbf{Q_i^*}))] \\
&= \gamma \mathbb{E}[Q_{tot}(s, \max_{\mathbf{a}} \mathbf{Q_i}) - Q_{tot}(s, \max_{\mathbf{a}} \mathbf{Q_i^*})] \\
&= \gamma \mathbb{E}[Q_{tot}(s, \max_{a_1} Q_1, ..., \max_{a_N} Q_N) \\
&\quad - Q_{tot}(s, \max_{a_1} Q_1^*, ..., \max_{a_N} Q_N^*)] \\
&\geq \gamma \mathbb{E}[\sum_{i}^{N} l(\max_{a_i} Q_i(s, a_i) - \max_{a_i} Q_i^*(s, a_i))] \\
&= \gamma lN\mathbb{E}[\max_{a_i} Q_i(s, a_i) - \max_{a_i} Q_i^*(s, a_i)] \\
&= lN\mathbb{E}[Z_i^s],
\end{aligned} \tag{16}$$

where the estimated Q_i is assumed with an independent noise uniformly distributed in $[-\varepsilon, \varepsilon]$ on each action a_i given s. Similarly, we can also get $\mathbb{E}[r + \gamma \max_{\mathbf{a}} Q_{tot}(s, \mathbf{Q_i}) - (r + \gamma \max_{\mathbf{a}} Q_{tot}(s, \mathbf{Q_i^*}))] \leq LN\mathbb{E}[Z_i^s]$. □

B THE IMPLEMENTATION DETAILS OF BASELINES

B.1 S-QMIX

S-QMIX use the softmax Bellman operator to compute the estimation of the target global Q-value

$$\text{softmax}_{\beta, \mathbf{U}}(Q_{tot}(s, \cdot)) = \sum_{\mathbf{u} \in \mathbf{U}} \frac{e^{\beta Q_{tot}(s, \mathbf{u})}}{\sum_{\mathbf{u}' \in \mathbf{U}} e^{\beta Q_{tot}(s, \mathbf{u}')}} Q_{tot}(s, \mathbf{u}), \tag{17}$$

where $\beta \geq 0$ is the inverse temperature parameter. However, computation of Equation (17) in the multiagent setting can be computationally intractable as the size of the joint action space grows exponentially with the number of agents. Therefore, Pan et al. (2021) use an alternative joint action set $\hat{\mathbf{U}}$ to replace the joint action space $\mathbf{U}$. First, the maximal joint action $\hat{\mathbf{u}}$ is obtained by $\hat{\mathbf{u}} = \arg\max_{\mathbf{u}} Q(s, \mathbf{u})$. Next, for each agent i, K joint actions are considered by changing only agent i's action while keeping the other agents' actions $\mathbf{u}_{-i}$ fixed and the resulting action set of agent i is $U_i = \{(u_i, \hat{\mathbf{u}}_{-i}) | u_i \in U\}$. Finally, the joint action subspace $\hat{\mathbf{U}} = U_i \cup \cdots \cup U_N$ is obtained and used to calculate the softmax version of the global Q-value.

B.2 SM2-QMIX

SM2-QMIX uses the soft Mellowmax operator to compute the estimation of the target individual Q-value and thus avoids the explosion problem of the joint action space in S-QMIX.

$$sm_{\omega} Q_i(s, \cdot) = \frac{1}{\omega} \log[\sum_{a \in A} \frac{e^{\alpha Q_i(s,a)}}{\sum_{a' \in A} e^{\alpha Q_i(s,a')}} e^{\omega Q_i(s,a)}], \tag{18}$$

where $\omega > 0$ and $\alpha \in \mathbb{R}$, which can be viewed as a particular instantiation of the weighted quasi-arithmetic mean (Beliakov et al., 2015).

B.3 TD3-QMIX

TD3-QMIX takes the minimum between the two critics' estimations to calculate the target global Q-value.

$$\begin{aligned} y_{tot} &= r + \gamma \min_{h\in\{1,2\}} Q_{tot}^{\bar{\phi}_h}(s', Q_1(o'_1, a'_1), ..., Q_N(o'_N, a'_N)), \\ Q_i(o'_i, a'_i) &= \max_{a'_i} Q_i^{\bar{\theta}_i}(o'_i, a'_i). \end{aligned} \tag{19}$$

B.4 WCU-QMIX

WCU-QMIX propose a weighted critic updating scheme of TD3. It updates the critic networks with the loss function that is calculated using the weighted Q-values obtained from the two critic networks. The target is calculated using the rewards and the minimum of the target Q-values.

$$\begin{aligned} L(\phi_h, \theta_1, ..., \theta_N) &= \frac{1}{|B|}\sum_b (y_b - (wQ_{tot}^{\phi_h}(s_b, Q_1, ..., Q_N) + (1-w)Q_{tot}^{\phi_p}(s_b, Q_1, ..., Q_N))|_{p\neq h}), \\ Q_i &= \max_{a_i} Q_i^{\theta_i}(o_{i,b}, a_i). \end{aligned} \tag{20}$$

B.5 SUB-AVG-QMIX

The Sub-Avg-QMIX keeps multiple target networks to maintain various action values of different periods, and discards the larger action values of them to eliminate the excessive overestimation error. Thereby, Sub-Avg-QMIX gets an overall lower maximum action value, then obtain an overall lower update target. Specifically, Sub-Avg-QMIX discards the action values above the average. Here we apply the Sub-Avg operator in the mixing network as it shows the better performance compared with the version of applying the Sub-Avg operator in the agent network (Wu et al., 2022).

$$y_{tot} = r + \gamma \max_{\mathbf{a}'} \left(\frac{\sum_{k=1}^{K} c_k Q_{tot}^{\bar{\theta}_{i,t-k+1}}(s', \mathbf{a}')}{\sum_{k=1}^{K} c_k}\right). \tag{21}$$

with

$$c_k = \max(0, sign(\bar{Q}_{tot}^{\bar{\phi}}(s', \mathbf{a}') - Q_{tot}^{\bar{\phi}_{t-k+1}}(s', \mathbf{a}'))), \tag{22}$$

where c_k determines whether the global action value should be preserved if it is below average or discarded otherwise. $\bar{Q}_{tot}^{\bar{\phi}}(s', \mathbf{a}')$ is the average of the last K global action values.

C PROOF OF HYPERNET REGULARIZER WITH $\frac{\partial Q_{tot}}{\partial Q_i}$ AND c

First, we show the relation between $\frac{\partial Q_{tot}}{\partial Q_i}$ and the hypernet weights and biases. For the most representative QMIX (Rashid et al., 2018), the global Q-value is calculated as follows

$$Q_{tot} = f_{mix}(s, Q_1, ..., Q_N) + c(s) = elu(\mathbf{Q}_i^{1\times N}\mathbf{W}_1^{N\times L_h} + \mathbf{B}_1^{1\times L_h})\mathbf{W}_2^{L_h\times 1} + c_1^{1\times 1}, \tag{23}$$

where $\mathbf{W}_1^{N\times L_h}$, $\mathbf{W}_2^{L_h\times 1}$, $\mathbf{B}_1^{1\times L_h}$ and $c_1^{1\times 1}$ are weights and biases generated from the corresponding hypernets, and L_h is the hidden unit number. Then

$$\begin{aligned} \frac{\partial Q_{tot}}{\partial Q_i} &= \frac{\partial elu(\mathbf{Q}_i^{1\times N}\mathbf{W}_1^{N\times L_h} + \mathbf{B}_1^{1\times L_h})\mathbf{W}_2^{L_h\times 1} + c_1^{1\times 1}}{\partial Q_i} \\ &= \frac{\partial elu(Q_i^{1\times 1}\mathbf{W}_{1,i}^{1\times L_h} + B_1^{1\times L_h})\mathbf{W}_2^{L_h\times 1}}{\partial Q_i} \\ &= \sum_{\substack{l_h=1 \\ Q_i w_{1,i,l_h} + b_{1,l_h} \geq 0}}^{L_h} w_{1,i,l_h} w_{2,l_h} + \sum_{\substack{l_h=1 \\ Q_i w_{1,i,l_h} + b_{1,l_h} < 0}}^{L_h} \alpha w_{1,i,l_h} w_{2,l_h} e^{Q_i w_{1,i,l_h} + b_{1,l_h}} \\ &\leq \sum_{\substack{l_h=1 \\ Q_i w_{1,i,l_h} + b_{1,l_h} \geq 0}}^{L_h} w_{1,i,l_h} w_{2,l_h} + \sum_{\substack{l_h=1 \\ Q_i w_{1,i,l_h} + b_{1,l_h} < 0}}^{L_h} \alpha w_{1,i,l_h} w_{2,l_h}, \end{aligned} \tag{24}$$

where $w_{1,i,l_h} \geq 0$ and $w_{2,l_h} \geq 0$, and elu is the Exponential Linear Unit activation function, and $\alpha > 0$ is a scalar (Clevert et al., 2016). Therefore, if we use

$$L_{hyper} = \sum \mathbf{W} + \sum |\mathbf{B}| + |c| = \sum w_1 + \sum w_2 + \sum |b_1| + |c_1| \tag{25}$$

as the regularization term in the loss function, we can restrict both the terms of $\frac{\partial Q_{tot}}{\partial Q_i}$ and c.

Furthermore,

$$\frac{\partial}{\partial Q_i}(\frac{\partial Q_{tot}}{\partial Q_i}) = \sum_{\substack{l_h=1 \\ Q_i w_{1,i,l_h} + b_{1,l_h} < 0}}^{L_h} \alpha w_{1,i,l_h}^2 w_{2,l_h} e^{Q_i w_{1,i,l_h} + b_{1,l_h}} \geq 0. \tag{26}$$

Therefore, in QMIX, $\frac{\partial Q_{tot}}{\partial Q_i}$ increases with Q_i if $Q_i w_{1,i,l_h} + b_{1,l_h} < 0$ for some l_h and the overestimation can be bootstrapped to an extreme, which is also observed from the experiments (Pan et al., 2021).

For the linear value-mixing algorithms such as Qatten (Yang et al., 2020), the global Q-value is calculated as follows

$$Q_{tot} = \sum_{i=1}^{N} w_i Q_i + c, \tag{27}$$

where $w_i \geq 0$, the following simply holds as

$$\frac{\partial Q_{tot}}{\partial Q_i} = w_i. \tag{28}$$

The hypernet regularizer to restrict both the terms of $\frac{\partial Q_{tot}}{\partial Q_i}$ and c is

$$L_{hyper} = \sum \mathbf{W} + \sum |\mathbf{B}| + |c| = \sum w + |c|. \tag{29}$$

D NOISY SMAC SETTINGS

In this noisy SMAC environment, we add a random noise for each feature of both the observation and global state, which is common in the realistic tasks. The random noise is uniformly distributed and its range is $[0, 0.02)$. Although the noise is small, it dramatically raises the overestimation problem for QMIX and seriously impedes the quality of the learned policies.

E HYPERPARAMETER SETTINGS

Table 1: Hyperparamters of algorithms on MPE.

DEMIX	simple tag	simple world	simple adversary
H	3	10	10
N_H	3	6	4
K	1	1	10
N_K	1	1	4
α_{hyper}	0.002	0.02	0.05
S-QMIX	simple tag	simple world	simple adversary
β	0.05	0.5	0.005
SM2-QMIX	simple tag	simple world	simple adversary
α	0.1	10.0	0.1
ω	5.0	0.05	0.5
WCU-QMIX	simple tag	simple world	simple adversary
w	0.75	0.75	0.75

As different tasks have different levels of overestimation, we adjust the hyperparameters of each method on each task. To make a fair comparision, we perform the grid search for all baselines around their tuned default values which perform best in their original papers. Specifically, for S-QMIX, we search $\beta \in \{50.0, 5.0, 0.5, 0.05, 0.005\}$. For SM2-QMIX, we search $(\alpha, \omega) \in$

Table 2: Hyperparamters of algorithms on SMAC.

DEMIX	5m_vs_6m	2s3z	3s5z	10m_vs_11m
H	3	3	10	4
N_H	2	2	6	3
K	1	1	1	1
N_K	1	1	1	1
α_{hyper}	0.002	0.002	0.001	0.01
SM2-QMIX	5m_vs_6m	2s3z	3s5z	10m_vs_11m
α	1.0	10.0	10.0	1.0
ω	0.5	0.05	5.0	5.0
WCU-QMIX	5m_vs_6m	2s3z	3s5z	10m_vs_11m
w	0.75	0.75	0.75	0.75
Sub-Avg-QMIX	5m_vs_6m	2s3z	3s5z	10m_vs_11m
K	10	3	3	3

$\{(10.0,5.0),(10.0,0.5),(10.0,0.05),(1.0,5.0),(1.0,0.5),(1.0,0.05),(0.1,5.0),(0.1,0.5),(0.1,0.05)\}$. For WCU-QMIX, we search $w \in \{0.25,0.5,0.75\}$. Here are the tuned hyperparameter parameters of each method on each task of MPE as shown in Table 1.

We also use the grid search on eahc of the SMAC tasks. For SM2-QMIX, we search $(\alpha,\omega) \in \{(10.0,5.0),(10.0,0.5),(10.0,0.05),(1.0,5.0),(1.0,0.5),(1.0,0.05),(0.1,5.0),(0.1,0.5),(0.1,0.05)\}$. For WCU-QMIX, we search $w \in \{0.25,0.5,0.75\}$. For Sub-Avg-QMIX, we search $K \in \{3,5,10\}$. Here are the methods' tuned hyperparameter parameters on each task of SMAC as shown in Table 2.