

Dual Ensembled Multiagent Q-Learning with Hypernet Regularizer

Yaodong Yang
Department of CSE, CUHK
Hong Kong, China
yydapple@gmail.com

Guangyong Chen
Zhejiang Lab
Hangzhou, China
gychen@zhejianglab.com

Hongyao Tang
Mila, University of Montreal
Montreal, Canada
tang.hongyao@quebec.mila

Furui Liu
Zhejiang Lab
Hangzhou, China
liufurui@zhejianglab.com

Danruo Deng
Department of CSE, CUHK
Hong Kong, China
drdeng@link.cuhk.edu.hk

Pheng Ann Heng
Department of CSE, CUHK
Hong Kong, China
pheng@cse.cuhk.edu.hk

ABSTRACT

Overestimation in single-agent reinforcement learning has been extensively studied. In contrast, overestimation in the multiagent setting has received comparatively little attention although it increases with the number of agents and leads to severe learning instability. Previous works concentrate on reducing overestimation in the estimation process of target Q-value. They ignore the follow-up optimization process of online Q-network, thus making it hard to fully address the complex multiagent overestimation problem. To solve this challenge, in this study, we first establish an iterative estimation-optimization analysis framework for multiagent value-mixing Q-learning. Our analysis reveals that multiagent overestimation not only comes from the computation of target Q-value but also accumulates in the online Q-network's optimization. Motivated by it, we propose the Dual Ensembled Multiagent Q-Learning with Hypernet Regularizer algorithm to tackle multiagent overestimation from two aspects. First, we extend the random ensemble technique into the estimation of target individual and global Q-values to derive a lower update target. Second, we propose a novel hypernet regularizer on hypernetwork weights and biases to constrain the optimization of online global Q-network to prevent overestimation accumulation. Extensive experiments in MPE and SMAC show that the proposed method successfully addresses overestimation across various tasks¹.

KEYWORDS

Multiagent Reinforcement Learning; Q-value Overestimation

ACM Reference Format:

Yaodong Yang, Guangyong Chen, Hongyao Tang, Furui Liu, Danruo Deng, and Pheng Ann Heng. 2025. Dual Ensembled Multiagent Q-Learning with Hypernet Regularizer. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025)*, Detroit, Michigan, USA, May 19 – 23, 2025, IFAAMAS, 9 pages.

¹Code is available at GitHub.

Corresponding author: Guangyong Chen.



This work is licensed under a Creative Commons Attribution International 4.0 License.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Overestimation is a critical challenge for reinforcement learning that stems from the maximum operation when bootstrapping the target Q-value [23]. This Q-value overestimation can be continually accumulated during learning, leading to sub-optimal policy updates and behaviors, causing instability and crippling the quality of learned policy [10]. In single-agent deep reinforcement learning (DRL), a lot of representative works have been proposed to address the overestimation problem, including Double DQN [8], Averaged-DQN [2], TD3 [4], Minmax Q-learning [10], and MeanQ [11] etc. Although overestimation in single-agent DRL has been widely studied, overestimation in multiagent reinforcement learning (MARL) has received comparatively little attention although it increases with the number of agents and could cause severe learning instability to harm agent policy [18].

To mitigate overestimation in MARL, a few preceding works extend ensemble methods from single-agent DRL to reduce Q-value overestimation by discarding large target values in the ensemble. For example, Ackermann et al. [1] introduce the TD3 technique to reduce the overestimation bias by using double centralized critics. Recently, Wu et al. [27] use an ensemble of 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-based methods, another category of works softens the maximum operator in the Bellman equation to avoid updating with large target values. For example, Gan et al. [5] extend the soft Mellowmax operator into the field of MARL to alleviate the multiagent overestimation issue. At the same time, Pan et al. [18] use the softmax Bellman operator on the joint action space to avoid large target Q-values and speed up the softmax computation by sampling actions around the maximal joint action. However, the above works focus on reducing the overestimation of MARL in the estimation process of the target Q-value. They ignore the follow-up optimization process of the online Q-network where overestimation can accumulate, thus making it hard to fully solve the complex multiagent overestimation problem.

To address this challenge, in our work, we first establish an iterative estimation-optimization framework to analyze the overestimation phenomenon in multiagent value-mixing Q-learning. Through the analysis, we found that multiagent overestimation not only comes from the overestimation of target Q-values [5], but also accumulates in the online Q-network after optimization with

the overestimated update target. Inspired by this finding, we propose the **Dual Ensembled MultiAgent Q-learning with hypernet Regularizer (DEMAR)** algorithm to address multiagent overestimation from two aspects. First, DEMAR extends a random ensemble technique [3] into the estimation of target individual and global Q-values to derive a lower update target. Second, to prevent the accumulation of overestimation, we propose a novel hypernet regularizer to regularize the weights and biases from hypernetworks to constrain the optimization of online global Q-network. Extensive experiments in the classical multiagent particle environment and a noisy version of the StarCraft multiagent challenge environment show that DEMAR successfully stabilizes learning on various multiagent tasks suffering from overestimation while outperforming other baselines of the multiagent overestimation problem.

2 BACKGROUND

2.1 Overestimation in Q-learning

Q-learning [26] learns the optimal value of each state-action via updating a tabular representation of the Q-value function by

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

to minimize the temporal difference error between estimates and bootstrapped targets with learning rate α and discount factor γ . DQN [17] learns a parametrized value function Q^θ by minimizing the squared error loss between Q^θ and the target value estimation as

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

where Q^θ is a lagging version of the current state-action value function and is updated periodically from Q^θ . When the target value estimation is with noise, the max operator has been shown to lead to an overestimation bias [23] due to Jensen's inequality as

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

Since the update is applied repeatedly through bootstrapping, it iteratively increases the bias of estimated Q-values and introduces instability into learning.

2.2 Randomized Ensembled Double Q-Learning

Using an ensemble of Q-networks to reduce overestimation is widely studied in the field of single-agent DRL [2, 3, 11]. To reduce overestimation of the target Q-value, Double DQN [8] decomposes the max operation in the target into action selection and action evaluation, and TD3 [4] extends this idea into actor-critic methods. Meanwhile, Averaged-DQN [2] and MeanQ [11] both employ an ensemble of multiple Q-networks to reduce overestimation with a lower estimation variance.

Recently, Randomized Ensembled Double Q-Learning (REDQ) [3] adopts an in-target minimization across a subset \mathbb{H} of $N_{\mathbb{H}}$ Q-functions, which is randomly sampled from an ensemble of H Q-functions, to derive a lower update target. Based on SAC [7], the Q target of REDQ is computed as

$$y = r + \gamma(\min_{h \in \mathbb{H}} Q^h(s', a') - \alpha_{sac} \log \pi(s', a')), \tilde{a}' \sim \pi(\cdot | s'), \quad (4)$$

where h is the index of Q-function and α_{sac} is the coefficient of the entropy term in SAC. As indicated by Theorem 1 of REDQ [3], by

adjusting H and $N_{\mathbb{H}}$, the overestimation can be flexibly controlled. For example, by increasing the subset size $N_{\mathbb{H}}$ with some fixed ensemble size H , the estimation bias of the Q-value can be adjusted from above zero (overestimation) to under zero (underestimation).

2.3 Multiagent Value-Mixing Q-learning

We use Markov games, the multiagent extension of Markov Decision Processes [12], as our setting. Markov games are described by a state transition function, $T : S \times A_1 \times \dots \times A_N \rightarrow P(S)$, which defines the probability distribution over all possible next states, $P(S)$, given the current global state S and the action A_i produced by the i -th agent. The reward is usually given based on the global state and actions of all agents $R_i : S \times A_1 \times \dots \times A_N \rightarrow \mathbb{R}$. If all agents receive the same rewards, i.e. $R_1 = \dots = R_N$, Markov games are fully-cooperative [16]: a best-interest action of one agent is also a best-interest action of other agents. Markov games can be partially observable and each agent i receives a local observation $o_i : O(S, i) \rightarrow O_i$. Thus, each agent learns a policy $\pi_i : O_i \rightarrow P(A_i)$, which maps each agent's observation to a probability distribution over its action set, to maximize its expected discounted returns, $J_i(\pi_i) = \mathbb{E}_{a_1 \sim \pi_1, \dots, a_N \sim \pi_N, s \sim T} [\sum_{t=0}^{\infty} \gamma^t r_i(s_t, a_{1,t}, \dots, a_{N,t})]$, where γ is the discounted factor and is in the range of $[0, 1)$.

To solve Markov games, multiagent value-mixing Q-learning algorithms [19, 29] represent the global Q-value as an aggregation of individual Q-values with different forms. As the most representative method, QMIX [19] learns a monotonic mixing network to transform individual Q-values into the global Q-value as $Q_{tot} = f_{mix}(s, Q_1(s, a_1), \dots, Q_N(s, a_N))$. To ensure the monotonicity between global and individual Q-values that $\frac{\partial Q_{tot}}{\partial Q_i} \geq 0$, QMIX utilizes the hypernetworks [6] to produce non-negative weights and biases for mixing Q_i s. A hypernetwork HPN is a network such as a multilayer perceptron. It takes the global state s as the input to output non-negative weights and biases for the mixing network f_{mix} . For example, we can build Q_{tot} with a linear form as $f_{mix} = \sum_{i=1}^N w_i Q_i + b$ where the weights $w_i \geq 0$ and bias b are produced from the hypernetworks $\mathbf{w} = HPN_1(s)$ and $b = HPN_2(s)$ [29]. The specific form of f_{mix} in QMIX is described in Appendix B. Recently, many works [14, 15, 24] aim to address the monotonicity limitation of value-mixing MARL. The representation limitation from monotonicity causes inaccurate multiagent utilities (higher utilities on some joint-action pairs as overestimation and lower utilities on other pairs as underestimation) to lead to sub-optimal policies. Differently, the multiagent overestimation causes all joint-action multiagent Q-values larger than the ground truth to cripple policy quality. In this paper, we focus on the multiagent overestimation problem which receives much less attention nowadays.

2.4 Overestimation in Multiagent Q-learning

In the single-agent setting, if estimated action values contain independent noise uniformly distributed in $[-\epsilon, \epsilon]$ ($\epsilon > 0$), the expected overestimation of target values becomes

$$\mathbb{E}[Z^s] = \mathbb{E}[r + \gamma \max_{a'} Q(s', a') - (r + \gamma \max_{a'} Q^*(s', a'))] \in [0, \gamma \epsilon \frac{m-1}{m+1}], \quad (5)$$

where Q^* is the target optimal Q-value and m is the action space size [23]. Overestimation in MARL is more sophisticated as it increases

with the number of agents [18]. Furthermore, Gan et al. [5] prove that the overestimation in multiagent value-mixing algorithms is raised from the maximization of individual Q-values with noise over its action space, which is shown below.

Lemma 2.1 (Gan et al. [5]). *Let Q_{tot} be a function of s and Q_i for $i = 1, 2, \dots, N$ where Q_i is a function of s and a_i for $a_i \in A_i$. Assuming $l \leq \frac{\partial Q_{tot}}{\partial Q_i} \leq L, i = 1, 2, \dots, N$ where $l \geq 0, L > 0$, and $Q_i(s, a_i)$ is with an independent noise uniformly distributed in $[-\epsilon, \epsilon]$ on each a_i given state s . Then*

$$\begin{aligned} \mathbb{E}[Z_i^s] &\leq \mathbb{E}[r + \gamma \max_{a'} Q_{tot}(s', Q(s', a')) - \\ &\quad (r + \gamma \max_{a'} Q_{tot}(s', Q^*(s', a')))] \leq L N \mathbb{E}[Z_i^s], \end{aligned} \quad (6)$$

where $Q(s', a') = (Q_1(s', a'_1), \dots, Q_N(s', a'_N))$ are individual Q-values, $Q^*(s', a') = (Q_1^*(s', a'_1), \dots, Q_N^*(s', a'_N))$ are the target optimal individual Q-values, and $\mathbb{E}[Z_i^s] = \mathbb{E}[\max_{a'_i} Q_i(s', a'_i) - \max_{a'_i} Q_i^*(s', a'_i)] \geq 0$. The proof is in Appendix A for completeness.

Eq. (6) shows that, when computing target global Q-values, overestimation is raised by maximizing Q_i over its action space. However, previous works ignore the fact that the overestimation could also be accumulated when optimizing the Q-network, thus not fully tackling multiagent overestimation. Next, we show how multiagent overestimation is raised and accumulated during learning.

3 DEMAR FOR MULTIAGENT OVERESTIMATION

In this section, first, we conduct an in-depth analysis of the overestimation in multiagent value-mixing Q-learning during the estimation-optimization iterations. Next, motivated by the findings of the analysis, we propose the Dual Ensembled Multiagent Q-Learning with Hypernet Regularizer algorithm as shown in Figure 1.

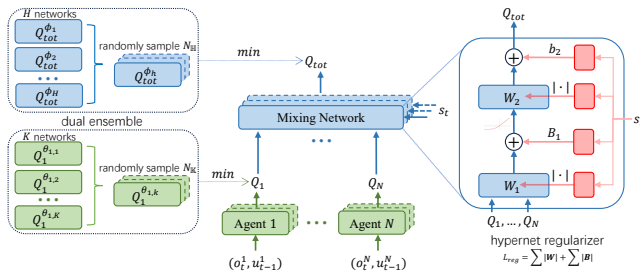


Figure 1: The framework of DEMAR. The left part involves the dual ensembled multiagent Q-learning while the right part shows the hypernet regularizer on the global Q-network.

3.1 Analysis of Multiagent Overestimation

Multiagent value-mixing Q-learning learns in estimation-optimization iterations. At each iteration, the target global Q-value is estimated first. Then the online global Q-network is optimized based on the estimated target global Q-value. Following multiagent value-mixing Q-learning [19, 22, 29] which approximates the global Q-value Q_{tot} by mixing individual Q-values Q_i , we expand Q_{tot} as

$$Q_{tot} = Q_{tot}(s, Q_1, \dots, Q_N) = f_{mix}(s, Q_1(s, a_1), \dots, Q_N(s, a_N)), \quad (7)$$

where f_{mix} is monotonic mixing network by enforcing $\frac{\partial Q_{tot}}{\partial Q_i} \geq 0$.

First, we analyze the estimation process of the target global Q-value. Lemma 2.1 indicates that the overestimation of the target Q_{tot} results from overestimated target Q_i s by monotonicity in multiagent value-mixing Q-learning. And the overestimation of the target Q_i results from independent noise on each action-value of Q_i by maximizing Q_i over its action space. After an overestimated target Q_{tot} is obtained, the online global Q-network is trained to fit this overestimated update target. Therefore, the overestimation of the target Q_{tot} and Q_i contribute to the multiagent overestimation.

Second, we analyze the optimization process of the online global Q-network. The update target of online global Q-network is computed as $y_{tot} = r + \gamma \max_{a'} Q_{tot}(s', a')$ with the overestimated target $Q_{tot}(s', a')$. We found that, after the online global Q-network's optimization with y_{tot} , $\frac{\partial Q_{tot}}{\partial Q_i}$ impacts the overestimation of the global Q-network with a quadratic term as below.

Theorem 3.1. *Assume that the current Q_{tot} network approximates the optimal $Q_{tot}^*(\cdot)$ as $Q_{tot}(\cdot) = Q_{tot}^*(\cdot)$ and the current Q_i network approximates the optimal $Q_i^*(\cdot)$ as $Q_i(\cdot) = Q_i^*(\cdot)$. By Lemma 2.1, if $l \leq \frac{\partial Q_{tot}}{\partial Q_i} \leq L$ for $i = 1, 2, \dots, N$ where $l \geq 0, L > 0$ and the estimated individual Q-value is with an independent noise uniformly distributed in $[-\epsilon, \epsilon]$ on each action, then the estimated update target y_{tot} will be biased from the optimal target value y_{tot}^* as $y_{tot} = y_{tot}^* + \Delta y$ where the estimation bias $\Delta y > 0$. If we continue to train the current Q_i network by the loss $L_{mix} = \|y_{tot} - Q_{tot}\|^2$, the updated network \hat{Q}_i will be biased from optimal Q_i^* as*

$$\hat{Q}_i = Q_i^* + \Delta Q_i, \quad (8)$$

where $\Delta Q_i \geq 2\alpha\Delta y$ for some learning rate $\alpha > 0$. 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}^* becomes

$$\hat{Q}_{tot} = Q_{tot}^* + 2\alpha\Delta y \sum_{i=1}^N \left(\frac{\partial Q_{tot}}{\partial Q_i} \right)^2. \quad (9)$$

PROOF. From Lemma 2.1, we have an overestimated target value $y_{tot} = y_{tot}^* + \Delta y$ where $\Delta y > 0$. As we assume that the learned Q_{tot} network approximates the optimal function, we have $Q_{tot}(s, Q_1, \dots, Q_N) = y_{tot}^*$. We apply the gradient method to minimize L_{mix} , the independent Q_i is updated as follows,

$$\begin{aligned} \hat{Q}_i &= Q_i^* - \alpha \frac{\partial L_{mix}}{\partial Q_i} \\ &= Q_i^* - \alpha \frac{\partial (y_{tot} - Q_{tot}(s, Q_1, \dots, Q_N))^2}{\partial Q_i} \\ &= Q_i^* - \alpha \frac{\partial (y_{tot}^* + \Delta y - Q_{tot}(s, Q_1, \dots, Q_N))^2}{\partial Q_i} \\ &= Q_i^* + 2\alpha(y_{tot}^* + \Delta y - Q_{tot}(s, Q_1, \dots, Q_N)) \frac{\partial Q_{tot}(s, Q_1, \dots, Q_N)}{\partial Q_i} \\ &= Q_i^* + 2\alpha\Delta y \frac{\partial Q_{tot}(s, Q_1, \dots, Q_N)}{\partial Q_i}. \end{aligned} \quad (10)$$

Compared with updating with the ground-truth target global Q-value y_{tot}^* , Q_i is biased with $\Delta Q_i = 2\alpha\Delta y \frac{\partial Q_{tot}}{\partial Q_i} \geq 2\alpha\Delta y$. Such a 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

$$\begin{aligned}
\hat{Q}_{tot} &= Q_{tot}(s, \hat{Q}_1, \dots, \hat{Q}_N) = Q_{tot}(s, Q_1 + \Delta Q_1, \dots, Q_N + \Delta Q_N) \\
&\approx Q_{tot}(s, Q_1, \dots, Q_N) + \sum_{i=1}^N \frac{\partial Q_{tot}(s, Q_1, \dots, Q_N)}{\partial Q_i} \Delta Q_i \\
&= Q_{tot}(s, Q_1, \dots, Q_N) + 2\alpha \Delta y \sum_{i=1}^N \left(\frac{\partial Q_{tot}(s, Q_1, \dots, Q_N)}{\partial Q_i} \right)^2 \\
&= Q_{tot}^* + 2\alpha \Delta y \sum_{i=1}^N \left(\frac{\partial Q_{tot}(s, Q_1, \dots, Q_N)}{\partial Q_i} \right)^2 \\
&\geq Q_{tot}^* + 2\alpha N l^2 \Delta y,
\end{aligned} \tag{11}$$

where the second approximation comes from the first-order Taylor expansion of the multivariate function. 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 repeated estimation-optimization iterations, such a bias accumulates to increase the multiagent Q-value overestimation. \square

As Eq. (9) shows, when forwarding the updated biased \hat{Q}_i to compute a new global Q-value \hat{Q}_{tot} , $\frac{\partial Q_{tot}}{\partial Q_i}$ impacts the global Q-value's overestimation with a quadratic term. Besides, Pan et al. [18] empirically show that $\frac{\partial Q_{tot}}{\partial Q_i}$ in QMIX continually increases with overestimation. Therefore, we also need to regularize $\frac{\partial Q_{tot}}{\partial Q_i}$ in Q_{tot} 's optimization. In conclusion, the overall overestimation analysis motivates us to control the overestimated Q_{tot} and Q_i in the target Q-value estimation, as well as to regularize $\frac{\partial Q_{tot}}{\partial Q_i}$ in the online Q_{tot} network's optimization.

3.2 Dual Ensembled Multiagent Q-Learning

Motivated by the above analysis, first, we are going to control the overestimated Q_{tot} and Q_i in the estimation process of target Q_{tot} . Similar to works in single-agent DRL, here we introduce the idea of the ensemble into the estimation process of target Q-values to derive a lower update target. We extend REDQ [3], a state-of-the-art ensemble method with theoretical guarantee and impressive performance from single-agent DRL, into the multiagent value-mixing Q-learning algorithms. As shown in Section 2.2, REDQ uses an in-target minimization across a random subset of Q-functions from the ensemble of Q-networks to reduce overestimation. However, REDQ is specially built on the single-agent SAC [7] algorithm. It cannot be directly applied to the multiagent value-mixing Q-learning which involves the process of mixing individual Q_i s into Q_{tot} . In this study, we carefully design a dual ensembled algorithm based on the random ensemble technique from REDQ to control both the overestimated target Q_i and Q_{tot} during the mixing process. Next, we explain the details.

The global target value of multiagent value-mixing Q-learning [19] is computed as

$$\begin{aligned}
y_{tot} &= r + \gamma \max_{a'} Q_{tot}^{\bar{\phi}}(s', Q(s', a'_i)) \\
&= r + \gamma Q_{tot}^{\bar{\phi}}(s', \max_{a'} Q(s', a'_i)) \\
&= r + \gamma Q_{tot}^{\bar{\phi}}(s', \max_{a'_1} Q_1^{\bar{\theta}_1}(o'_1, a'_1), \dots, \max_{a'_N} Q_N^{\bar{\theta}_N}(o'_N, a'_N)),
\end{aligned} \tag{12}$$

where $\bar{\phi}$ and $\bar{\theta}$ are the parameters of the target global Q-network and target individual Q-networks respectively. As analyzed in Section 3.1, the overestimation in target Q_i results in the overestimation of target Q_{tot} . Therefore, we first apply the minimization operation in the random ensemble of target Q_i networks to reduce target Q_i 's overestimation. Thus, the target Q_i of agent i is computed as

$$Q_i^{\bar{\theta}_i} = \min_{k \in \mathbb{K}} Q_i^{\bar{\theta}_{i,k}}, \tag{13}$$

where \mathbb{K} is a subset with size $N_{\mathbb{K}}$ randomly sampled from $\{1, 2, \dots, K\}$ and $\bar{\theta}_{i,k}$ are the parameters of the agent i 's k th target individual Q-network. After getting the target Q_i , we are able to compute the target Q_{tot} . Similarly, we use the minimization operation again in the random ensemble of target Q_{tot} networks to reduce the overestimation of target global Q-value as

$$Q_{tot}^{\bar{\phi}} = \min_{h \in \mathbb{H}} Q_{tot}^{\bar{\phi}_h}, \tag{14}$$

where \mathbb{H} is a subset with size $N_{\mathbb{H}}$ randomly sampled from $\{1, 2, \dots, H\}$ and $\bar{\phi}_h$ are the parameters of the h th target global Q-network. With the proposed dual ensembled Q-learning technique, we derive a lower update target for the online global Q-network. In addition, as indicated by Theorem 1 of REDQ [3], we are able to flexibly control the overestimation of the target Q_{tot} and Q_i by changing their Q-network ensemble sizes and the random subset sizes respectively. Next, we are going to prevent the overestimation accumulation in the optimization process of the global Q-network.

3.3 Hypernet Regularizer

The analysis in Section 3.1 indicates that, in the optimization step of the online Q_{tot} network, $\frac{\partial Q_{tot}}{\partial Q_i}$ impacts the multiagent overestimation with a quadratic term. To tackle this issue, we propose a novel hypernet regularizer to constrain this term in the online global Q-network's optimization. Specifically, we use the L1 sparse regularization on hypernetwork weights and biases to constrain $\frac{\partial Q_{tot}}{\partial Q_i}$ as

$$L_{reg} = \sum |\mathbf{W}_f| + \sum |\mathbf{B}_f|, \tag{15}$$

where \mathbf{W}_f and \mathbf{B}_f are hypernetwork weights and biases in f_{mix} produced from separate hypernetworks. The proof of using the hypernet regularizer to constrain $\frac{\partial Q_{tot}}{\partial Q_i}$ is given in Appendix B.

The final loss function for DEMAR becomes

$$L(\phi_h, \theta) = L_{mix}^h + \alpha_{reg} L_{reg}^h, \tag{16}$$

where α_{reg} is the coefficient of hypernet regularization. h is the index of Q_{tot} network, ϕ_h are the network parameters of h th Q_{tot} network, and θ are the network parameters of all Q_i networks.

DEMAR is completely described in Algorithm 1. **Line 1** initializes the empty replay buffer, the online and target individual Q-value

Algorithm 1 Dual Ensembled Multiagent Q-Learning with Hypernet Regularizer (DEMAR)

```

1: Initialize individual Q-value network parameters  $\theta_{1,1}, \dots, \theta_{1,K}, \dots, \theta_{N,1}, \dots, \theta_{N,K}$ , global Q-value network parameters  $\phi_1, \phi_2, \dots, \phi_H$  and empty replay buffer  $D$ . Set target network parameters  $\bar{\theta}_{i,k} \leftarrow \theta_{i,k}$  for  $i = 1, 2, \dots, N, k = 1, 2, \dots, K$  and  $\bar{\phi}_h \leftarrow \phi_h$  for  $h = 1, 2, \dots, H$ .
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 state  $s_{t+1}$ . Receive reward  $r_t$  and observe  $o_{i,t+1}$ .
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  $B = \{(s, \mathbf{o}, \mathbf{a}, r, s', \mathbf{o}')\}$  from  $D$ .
6:   Sample a set  $\mathbb{K}$  of  $N_{\mathbb{K}}$  distinct indices from  $\{1, 2, \dots, K\}$ .
7:   Sample a set  $\mathbb{H}$  of  $N_{\mathbb{H}}$  distinct indices from  $\{1, 2, \dots, H\}$ .
8:   Compute the Q target  $y_{tot}$  which is the same for all  $H$  critics (denote  $\bar{\theta}_{i,1}, \dots, \bar{\theta}_{i,K}$  as  $\bar{\theta}_i$ ):

$$y_{tot} = r + \gamma (\min_{h \in \mathbb{H}} Q_{tot}^{\bar{\phi}_h}(s', Q_1^{\bar{\theta}_1}(o'_1, a'_1), \dots, Q_N^{\bar{\theta}_N}(o'_N, a'_N)), \text{ where } Q_i^{\bar{\theta}_i}(o'_i, a'_i) = \max_{a'_i} \min_{k \in \mathbb{K}} Q_i^{\bar{\theta}_{i,k}}(o'_i, a'_i).$$

9:   for  $h = 1, \dots, H$  do
10:    Update  $\phi_h, \theta_{1,1}, \dots, \theta_{N,K}$  (denote  $\theta_{1,1}, \dots, \theta_{N,K}$  as  $\theta$ ) with gradient descent:

$$\nabla_{\phi_h, \theta} L(\phi_h, \theta), \text{ where}$$


$$L(\phi_h, \theta) = \frac{1}{|B|} \sum_{(s, \mathbf{o}, \mathbf{a}, r, s', \mathbf{o}') \in B} ([Q_{tot}^{\phi_h}(s, Q_1^{\theta_1}(o_1, a_1), \dots, Q_N^{\theta_N}(o_N, a_N)) - y_{tot}]^2 + \alpha_{reg} L_{reg}^h(s)), \text{ and } Q_i^{\theta_i}(o_i, a_i) = \text{mean}_{k=1}^K Q_i^{\theta_{i,k}}(o_i, a_i).$$

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

```

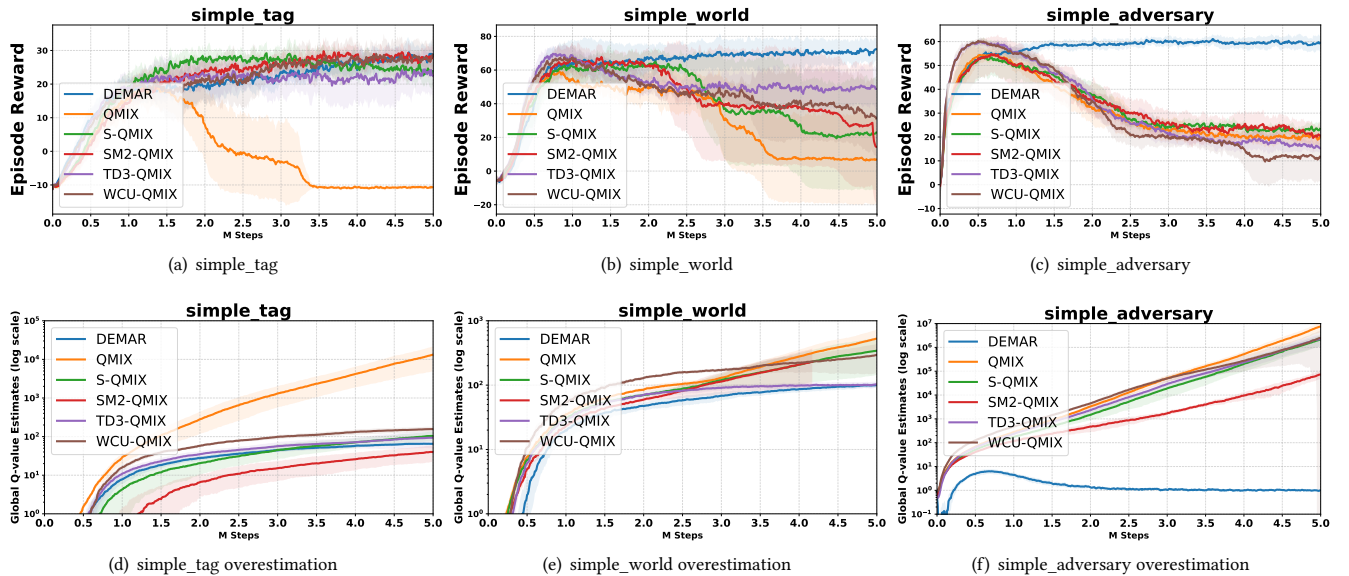


Figure 2: Results on different MPE scenarios. Figure 2(a)-2(c) show the learning performance of each method on MPE tasks. Figure 2(d)-2(f) show the estimated global Q-value of each method in the log scale on MPE tasks.

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. **Line 6-7** sample the indices for selecting instances from the 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. Specifically, for the computation of $\max_{a'_i} \min_{k \in \mathbb{K}} Q_i^{\bar{\theta}_{i,k}}(o'_i, a'_i)$, the $\min_{k \in \mathbb{K}} Q_i^{\bar{\theta}_{i,k}}$ is first computed for

each action in agent i ' action space by finding the minimal value in the ensemble subset, then the $\max_{a'_i}$ operation is executed. **Line 9-11** update all online Q_i and Q_{tot} networks by the loss L . Note that the online Q_i is calculated by averaging the ensemble of Q_i values with given actions instead of randomly picking up one value from the Q_i ensemble, which enjoys the benefit of reducing value variance. **Line 12** updates all target Q_i and Q_{tot} networks by copying network parameters from their online versions periodically. With the dual ensembled Q-learning and hypernet regularizer, DEMAR is able to control the multiagent overestimation. Additionally, by

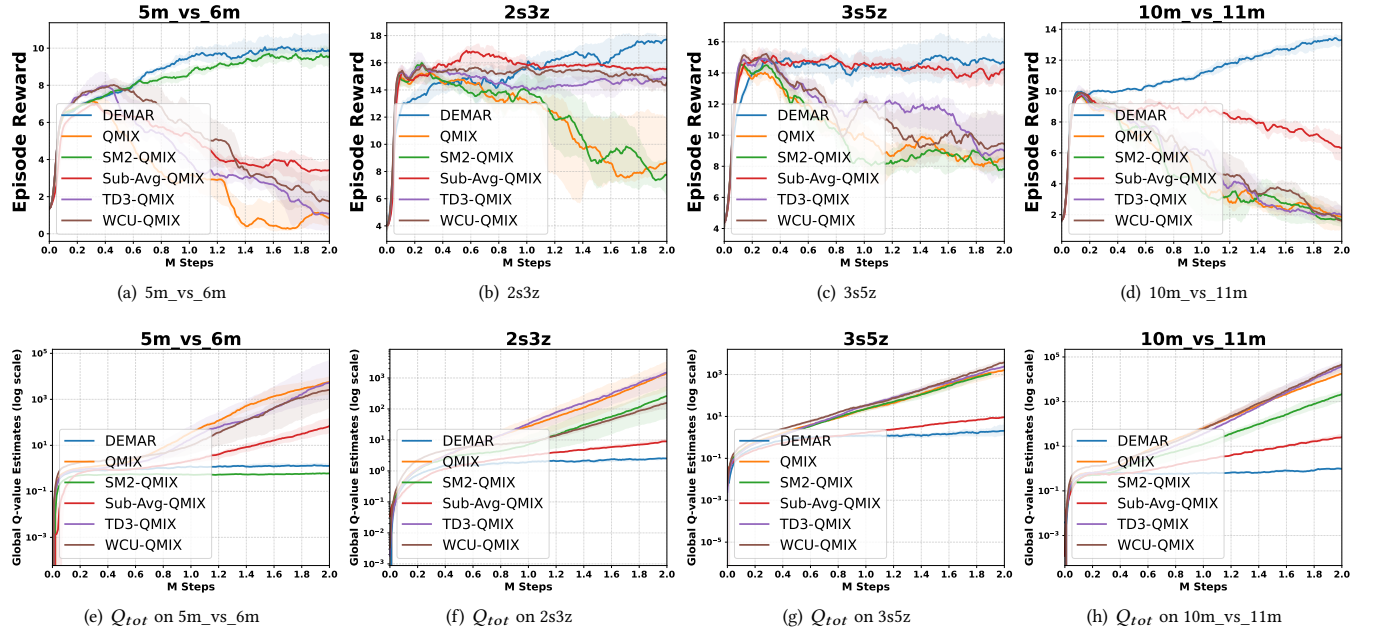


Figure 3: Results on different noisy SMAC scenarios. Figure 3(a)-3(d) show the learning performance of each method on SMAC tasks. Figure 3(e)-3(h) show the estimated global Q-value of each method in the log scale on SMAC tasks.

setting $H = N_{\mathbb{H}} = K = N_{\mathbb{K}} = 1$ and $\alpha_{reg} = 0$, DEMAR degenerates to vanilla QMIX.

4 EXPERIMENTS

In this section, we first conduct experiments in the multiagent particle environment (MPE) [13] and a noisy version of the StarCraft multiagent challenge (SMAC) [20] that both suffer from overestimation. Then we perform ablation studies to validate the proposed dual ensemble Q-learning and hypernet regularizer of DEMAR separately. Next, we experimentally examine the overestimation terms as analyzed in Section 3.1 and compare the estimated Q-value of DEMAR with its true Q-value. Finally, we extend DEMAR to other advanced MARL approaches to test the generality of DEMAR.

4.1 Experiments in MPE

We first evaluate DEMAR in MPE including the *simple_tag*, *simple_world*, and *simple_adversary* scenarios. *Simple_tag* is a predator-prey task where 3 slower predators coordinate to capture a faster prey. *Simple_world* involves 4 slower agents to catch 2 faster adversaries that desire to eat food. *Simple_adversary* involves 2 cooperating agents and 1 adversary where agents need to reach a specified target landmark of two landmarks while the adversary is unaware of the target. We compare DEMAR with several baselines including QMIX [19], S-QMIX [18], SM2-QMIX [5], TD3-QMIX [1] and WCU-QMIX [21]. All methods are implemented in the pymarl framework [20] and use one-step return. The details of each baseline are provided in Appendix C. We follow the training and evaluation settings of Pan et al. [18]. For the hyperparameter tuning, we conduct a grid search for each baseline. To reduce the load of the hyperparameter tuning for DEMAR, we adopt a heuristic sequential searching to

tune the hyperparameters of DEMAR, which is provided in Appendix E with the hyperparameter settings in MPE. For each method, we run 5 independent trials with different random seeds and the resulting plots include the mean performance as well as the shaded 95% confidence interval in Figure 2(a)-2(c). Besides, we plot the estimated Q_{tot} in the log scale to show the overestimation status of each method in Figure 2(d)-2(f).

As shown in Figure 2, DEMAR prevents overestimation and stabilizes the learning on all three MPE tasks. QMIX fails on all tasks as it gets the most severe overestimation as shown in Figure 2(d)-2(f). Meanwhile, although WCU-QMIX, S-QMIX, and SM2-QMIX tackle the *simple_tag* task, they fail on the two other tasks. While TD3-QMIX controls the overestimation in *simple_tag* and *simple_world*, it cannot tackle *simple_adversary* where the overestimation on this task is the most severe. Overall, DEMAR is the only method that successfully stabilizes the learning on all tasks in MPE.

4.2 Experiments in Noisy SMAC

Next, we conduct the experiments on a noisy version of the commonly used MARL benchmark SMAC [20]. In this noisy SMAC environment, the interference signal noises are added to the sensors of each agent’s observation and the global state (details in Appendix D). The noise increases the variances of the individual Q-value and global Q-value, and thus raises the overestimation problem for multiagent Q-learning algorithms, making the noisy SMAC environment a good testbed for MARL algorithms designed to address the multiagent overestimation problem. We perform the experiments 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. In the 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 marine enemies. We train multiple agents to control allies respectively while a built-in handcrafted AI controls the enemies. Training and evaluation schedules such as the testing setting and training hyperparameters are kept unchanged [20]. The version of StarCraft II is 4.6.2.

We compare DEMAR with QMIX, SM2-QMIX, TD3-QMIX, WCU-QMIX, and Sub-Avg-QMIX [27]. Here we add the Sub-Avg-QMIX which performs well on standard SMAC tasks [27]. We omit S-QMIX's results as it collapses while training. Similarly, we conduct the grid hyperparameter searching for baselines and use the sequential hyperparameter searching for DEMAR. The hyperparameter settings in SMAC are in Appendix E. Results are averaged over 6 independent training trials with different random seeds, and the resulting plots include the median performance as well as the shaded 25-75% percentiles. Here we report the episode reward as the performance metric instead of the test winrate because the test winrate is almost zero for some baselines in most tasks. The results are shown in Figure 3(a)-3(d). We also report the estimated Q_{tot} in the log scale in Figure 3(e)-3(h).

Figure 3 shows that, in the noisy SMAC, QMIX suffers from severe overestimation and cannot learn stably on all tasks. Meanwhile, although Sub-Avg-QMIX controls the overestimation and stabilizes learning on *2s3z* and *3s5z*, it fails on *5m_vs_6m* and *10m_vs_11m*. At the same time, SM2-QMIX performs well on *5m_vs_6m* by successfully controlling overestimation on this task but fails on the other three tasks. Both TD3-QMIX and WCU-QMIX fail to control the overestimation on most tasks. While other baselines cannot consistently tackle all the tasks, DEMAR successfully reduces the overestimation to stabilize the learning in these four maps, which validates DEMAR's effectiveness.

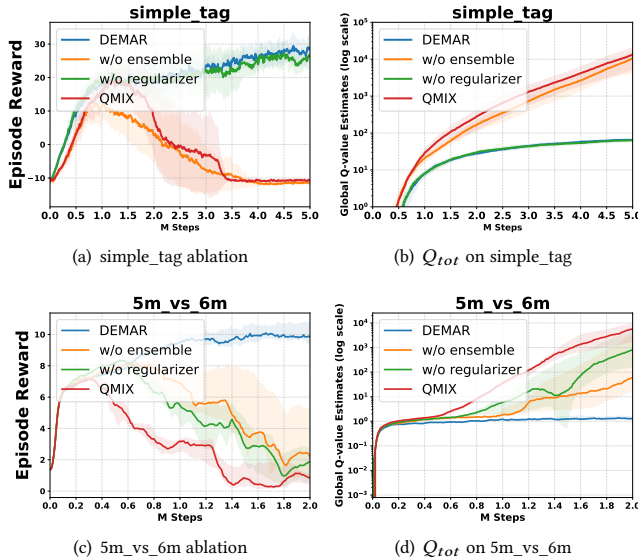


Figure 4: Results of the ablation study on *simple_tag* and *5m_vs_6m*. The w/o ensemble indicates DEMAR without the dual ensembled Q-learning. The w/o regularizer represents DEMAR without the hypernet regularizer.

4.3 Ablation Study of DEMAR

In this section, we perform the ablation study to validate each component of DEMAR. We compare DEMAR, DEMAR without the dual ensembled Q-learning (w/o ensemble), and DEMAR without the hypernet regularizer (w/o regularizer). When DEMAR is without both techniques, it degenerates to vanilla QMIX. Figure 4 shows the ablation results on *simple_tag* from MPE and *5m_vs_6m* from SMAC. As we can see, both techniques of DEMAR are essential to address the multiagent overestimation problem. Especially, in *5m_vs_6m*, neither the dual ensembled Q-learning nor the hypernet regularizer addresses the overestimation alone. Only the combined techniques, which jointly control the overestimation terms, can avoid overestimation and successfully stabilize learning. We also conduct the ablation study with different numbers of ensemble networks and different coefficients of the hypernet regularization. The additional experimental results are provided in Appendix F.

4.4 Experiment Study of Overestimation Terms

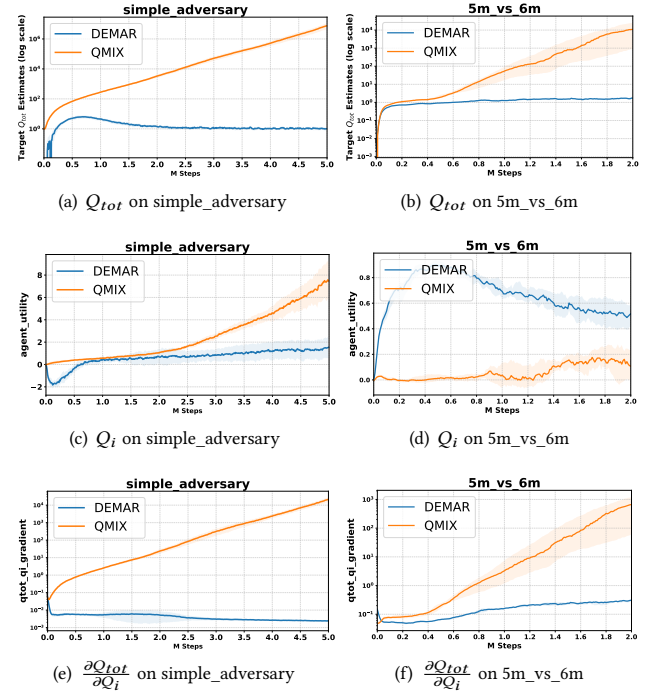


Figure 5: Results of analyzed overestimation terms including Q_{tot} , Q_i , and $\frac{\partial Q_{tot}}{\partial Q_i}$ on both *simple_adversary* and *5m_vs_6m*.

As DEMAR is designed to control each analyzed overestimation term, we experimentally examine Q_i and Q_{tot} in the target Q-value estimation as well as $\frac{\partial Q_{tot}}{\partial Q_i}$ in the online Q-network optimization to see whether DEMAR works as expected. Here we use the *simple_adversary* task because its overestimation is the most severe among all tasks. The values of each analyzed overestimation term are plotted in Figure 5(a)-5(e). We see that each overestimation term

in QMIX has higher values than DEMAR. Especially, $\frac{\partial Q_{tot}}{\partial Q_i}$ contributes much to the overestimation of QMIX as the value of $\frac{\partial Q_{tot}}{\partial Q_i}$ in QMIX is much larger than in DEMAR by orders of magnitude. Overall, the experimental results here correspond to our analysis in Theorem 3.1. By controlling each analyzed overestimation term, DEMAR successfully prevents severe multiagent overestimation.

4.5 Comparing True and Estimated Q-values

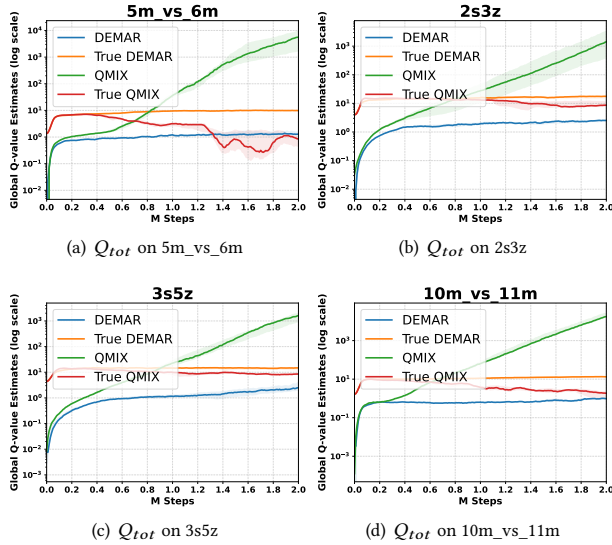


Figure 6: Comparison between True Q-values and Estimated Q-values. All Q-value curves are plotted in the log scale.

In this section, we compare the true Q-values and estimated Q-values of DEMAR as well as QMIX. We estimate true values by summing up the discounted returns of the following transitions starting from the sampled state. The results are shown in Figure 6. We could see that the estimated Q-value by DEMAR is closer to its true Q-value than QMIX. At the same time, the estimated Q-value by QMIX increases rapidly and its gap to its true Q-value also becomes larger as training continues.

4.6 Extending DEMAR to Other MARL Methods

To test the generality of DEMAR, we extend DEMAR to other advanced MARL algorithms such as ASN [25], UPDeT [9], and ATM [28]. ASN explicitly represents action semantics between agents and characterizes different actions' influence on other agents using neural networks to significantly improve the performance of MARL algorithms [25]. UPDeT utilizes a transformer-based model to enable multiple tasks transferring in MARL through the transformer's strong generalization abilities [9]. ATM proposes a transformer-based working memory mechanism to address partial observability in multiagent scenarios [28]. The results of extending DEMAR into ASN, UPDeT, and ATM are shown in Figure 7. As we can see, DEMAR helps ASN, UPDeT, and ATM avoid severe overestimation and stabilizes the learning process, which validates the generality of DEMAR. By the way, the global Q-value estimation curve of UPDeT

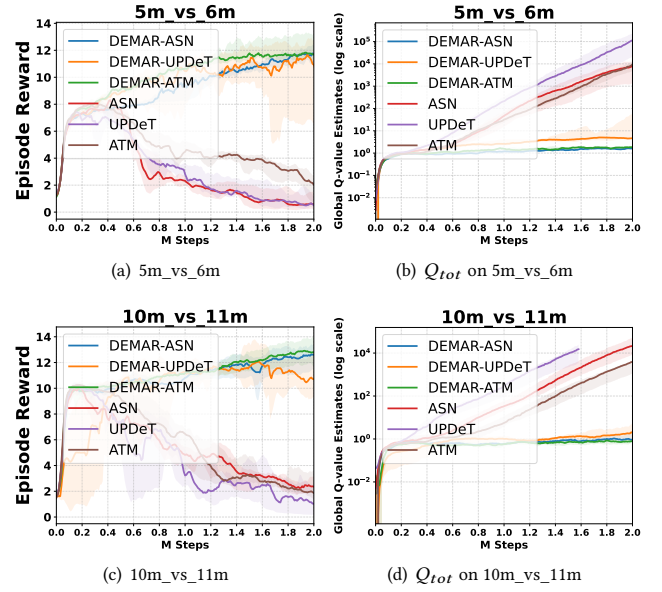


Figure 7: Results of UPDeT, ASN, and ATM with DEMAR.

on *10m_vs_11m* is clipped in Figure 7(d). During running UPDeT on *10m_vs_11m*, the global Q-value estimation in one trial increased too much to become NaN (too large to represent) while learning. Then the mean value of global Q-value estimation of UPDeT on *10m_vs_11m* also becomes NaN and is clipped when plotting.

5 CONCLUSION

In this study, we propose DEMAR to address the challenging multiagent overestimation problem. For the first time, we establish an iterative estimation-optimization analysis framework to systematically analyze the overestimation in multiagent value-mixing Q-learning. We found that the multiagent overestimation not only comes from the overestimation of target individual and global Q-values but also accumulates in the online Q-network's optimization. Motivated by this analysis finding, we propose DEMAR with dual ensembled Q-learning and hypernet regularizer to address these analyzed overestimation terms correspondingly. Extensive experiments in MPE and the noisy SMAC demonstrate that DEMAR successfully controls the multiagent overestimation.

For future work, on the one hand, there is a high potential to apply DEMAR to real-world multiagent scenarios where environmental noises are common and learning stability is a prerequisite. And it would be helpful to develop evaluation techniques for MARL in real-world applications. On the other hand, extending DEMAR to policy-based MARL algorithms is also promising.

ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (62376254, 32341018), a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 14201321), and a grant from Hong Kong Innovation and Technology Fund (Project No. MHP/092/22).

REFERENCES

- [1] Johannes Ackermann, Volker Gabler, Takayuki Osa, and Masashi Sugiyama. 2019. Reducing Overestimation Bias in Multi-Agent Domains Using Double Centralized Critics. In *Proceedings of NeurIPS Deep RL Workshop*.
- [2] Oron Anschel, Nir Baram, and Nahum Shimkin. 2017. Averaged-DQN: Variance Reduction and Stabilization for Deep Reinforcement Learning. In *Proceedings of the 34th International Conference on Machine Learning*. 176–185.
- [3] Xinyue Chen, Che Wang, Zijian Zhou, and Keith W. Ross. 2021. Randomized Ensembled Double Q-Learning: Learning Fast Without a Model. In *International Conference on Learning Representations*.
- [4] Scott Fujimoto, Herke van Hoof, and David Meger. 2018. Addressing Function Approximation Error in Actor-Critic Methods. In *Proceedings of the 35th International Conference on Machine Learning*, Vol. 80. 1587–1596.
- [5] Yaozhong Gan, Zhe Zhang, and Xiaoyang Tan. 2021. Stabilizing Q Learning Via Soft Mellowmax Operator. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 7501–7509.
- [6] David Ha, Andrew M. Dai, and Quoc V. Le. 2017. HyperNetworks. In *International Conference on Learning Representations*.
- [7] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 1861–1870. <https://proceedings.mlr.press/v80/haarnoja18b.html>
- [8] Hado van Hasselt, Arthur Guez, and David Silver. 2016. Deep Reinforcement Learning with Double Q-Learning. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. AAAI Press, 2094–2100.
- [9] Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. 2021. UPDeT: Universal Multi-agent RL via Policy Decoupling with Transformers. In *International Conference on Learning Representations*. <https://openreview.net/forum?id=v9c7hr9ADKx>
- [10] Qingfeng Lan, Yangchen Pan, Alona Fyshe, and Martha White. 2020. Maxmin Q-learning: Controlling the Estimation Bias of Q-learning. In *International Conference on Learning Representations*.
- [11] Litian Liang, Yaosheng Xu, Stephen McAleer, Dailin Hu, Alexander Ihler, Pieter Abbeel, and Roy Fox. 2022. Reducing Variance in Temporal-Difference Value Estimation via Ensemble of Deep Networks. In *Proceedings of the 39th International Conference on Machine Learning*, Vol. 162. 13285–13301.
- [12] Michael L. Littman. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine Learning Proceedings*. Elsevier, 157–163.
- [13] Ryan Lowe, YI WU, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Proceedings of the 31st Advances in Neural Information Processing Systems*. 6379–6390.
- [14] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. 2019. MAVEN: Multi-Agent Variational Exploration. In *Advances in Neural Information Processing Systems*, Vol. 32.
- [15] Anuj Mahajan, Mikayel Samvelyan, Lei Mao, Viktor Makovychuk, Animesh Garg, Jean Kossaifi, Shimon Whiteson, Yuke Zhu, and Animashree Anandkumar. 2021. Tesseract: Tensorised Actors for Multi-Agent Reinforcement Learning. In *Proceedings of the 38th International Conference on Machine Learning*, Vol. 139. PMLR, 7301–7312.
- [16] Laetitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. 2012. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. *The Knowledge Engineering Review* 27, 1 (2012), 1–31.
- [17] 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. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.
- [18] Ling Pan, Tabish Rashid, Bei Peng, Longbo Huang, and Shimon Whiteson. 2021. Regularized Softmax Deep Multi-Agent Q-Learning. In *Advances in Neural Information Processing Systems*.
- [19] Tabish Rashid, Mikayel Samvelyan, Christian Schröder de Witt, Gregory Farquhar, Jakob N. Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*. 4292–4301.
- [20] 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. 2019. The StarCraft Multi-Agent Challenge. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. 2186–2188.
- [21] Tamal Sarkar and Shobhanjana Kalita. 2021. A Weighted Critic Update Approach to Multi Agent Twin Delayed Deep Deterministic Algorithm. In *2021 IEEE 18th India Council International Conference*. 1–6.
- [22] 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. 2018. 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*. 2085–2087.
- [23] Sebastian Thrun and A. Schwartz. 1993. Issues in Using Function Approximation for Reinforcement Learning. In *Proceedings of 4th Connectionist Models Summer School*.
- [24] Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. 2021. QPLEX: Duplex Dueling Multi-Agent Q-Learning. In *International Conference on Learning Representations*.
- [25] Weixun Wang, Tianpei Yang, Yong Liu, Jianye Hao, Xiaotian Hao, Yujing Hu, Yingfeng Chen, Changjie Fan, and Yang Gao. 2020. Action Semantics Network: Considering the Effects of Actions in Multiagent Systems. In *International Conference on Learning Representations*. <https://openreview.net/forum?id=ryg48p4tPH>
- [26] Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-learning. *Machine Learning* 8, 3–4 (1992), 279–292.
- [27] Haolin Wu, Jianwei Zhang, Zhuang Wang, Yi Lin, and Hui Li. 2022. Sub-AVG: Overestimation reduction for cooperative multi-agent reinforcement learning. *Neurocomputing* 474 (2022), 94–106.
- [28] Yaodong Yang, Guangyong Chen, Weixun Wang, Xiaotian Hao, Jianye HAO, and Pheng-Ann Heng. 2022. Transformer-based Working Memory for Multiagent Reinforcement Learning with Action Parsing. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). <https://openreview.net/forum?id=pd6ipu3jDw>
- [29] Yaodong Yang, Jianye Hao, Ben Liao, Kun Shao, Guangyong Chen, Wulong Liu, and Hongyao Tang. 2020. Qatten: A General Framework for Cooperative Multiagent Reinforcement Learning. *CoRR abs/2002.03939* (2020).