Qatten: A General Framework for Cooperative Multiagent Reinforcement Learning

Yaodong Yang[1], **Jianye Hao**[1]*, **Ben Liao**[2] **Kun Shao**[3], **Guangyong Chen**[4], **Wulong Liu**[3], **Hongyao Tang**[1]
[1]College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Tencent Quantum Lab
[3]Huawei Noah's Ark Lab, China
[4]Shenzhen Institutes of Advanced Technology

Abstract

In many real-world tasks, multiple agents must learn to coordinate with each other given their private observations and limited communication ability. Deep multiagent reinforcement learning (Deep-MARL) algorithms have shown superior performance in such challenging settings. One representative class of work is multiagent value decomposition, which decomposes the global shared multiagent Q-value Q_{tot} into individual Q-values Q^i to guide individuals' behaviors, i.e. VDN imposing an additive formation and QMIX adopting a monotonic assumption using an implicit mixing method. However, most of the previous efforts impose certain assumptions between Q_{tot} and Q^i and lack theoretical groundings. Besides, they do not explicitly consider the agent-level impact of individuals to the whole system when transforming individual Q^i's into Q_{tot}. In this paper, we theoretically derive a general formula of Q_{tot} in terms of Q^i, based on which we can naturally implement a multi-head attention formation to approximate Q_{tot}, resulting in not only a refined representation of Q_{tot} with an agent-level attention mechanism, but also a tractable maximization algorithm of decentralized policies. Extensive experiments demonstrate that our method outperforms state-of-the-art MARL methods on the widely adopted StarCraft benchmark across different scenarios, and attention analysis is further conducted with valuable insights.

1 Introduction

The cooperative multiagent reinforcement learning problem has achieved lots of attention in the last decade, where a system of agents learns towards coordinated policies to optimize the accumulated global rewards [1, 6, 13]. Complex tasks such as the coordination of autonomous vehicles [2], optimizing the productivity of a factory in distributed logistics [21] and energy distribution, are often modeled as the cooperative multi-agent learning problems and have great application prospects with enormous commercial value. One natural way to address cooperative MARL problems is the centralized approach, which views the multiagent system (MAS) as a whole and solves it as a single-agent learning task. In such settings, existing reinforcement learning (RL) techniques can be leveraged to learn joint optimal policies based on agents' joint observations and common rewards [19]. However, the centralized approach usually scales not well, since the joint action space of agents grows exponentially as the number of agents increases. Furthermore, the partial observation limitation and communication constraints also necessitate the learning of decentralized policies, which conditions only on the local action-observation history of each agent [4].

---

*Corresponding author. Email: jianye.hao@tju.edu.cn

arXiv:2002.03939v2 [cs.MA] 9 Jun 2020

Another choice is the decentralized approach that each agent learns its policy. Letting individual agents learn concurrently based on the global reward (aka. independent learners) [19] is the simplest option. However, it is shown to be difficult in even simple two-agent, single-state stochastic coordination problems. One main reason is that the global reward signal brings the non-stationarity that agents cannot distinguish between the stochasticity of the environment and the exploitative behaviors of other co-learners [9], and thus may mistakenly update their policies. To mitigate this issue, decentralized policies can be learned in the centralized training with decentralized execution (CTDE) paradigm. CTDE learns a fully centralized value function based on the joint action and state information and then uses it to guide the optimization of decentralized policies, an approach taken by counterfactual multi-agent (COMA) policy gradients [4], as well as work by [6]. However, the centralized critic of COMA suffers difficulty in evaluating global Q-values from the joint state-action space especially when there are more than a handful of agents and is hard to give an accurate multiagent baseline [16].

Different from the above methods, Value Decomposition Network (VDN) [18] represents Q_{tot} as a sum of individual Q-values that condition only on individual observations and actions. Decentralized policies arise simply from each agent, which selects actions greedily based its Q^i. However, VDN supposes a strict additive assumption between Q_{tot} and Q^i, and ignores any extra state information available during training. Later QMIX [14] is proposed to overcome VDN's limitations. QMIX employs a network that estimates joint action-values as a non-linear combination of per-agent values that condition on local observations. Besides, QMIX enforces that Q_{tot} is monotonic in Q^i, which allows computationally tractable maximization of the joint action-value in off-policy learning. But QMIX performs an implicit mixing of Q^i while regarding the mixing process as a black-box. Besides, when mixing individual Q^i's to Q_{tot}, QMIX uses weights directly produced from global features instead of accurately modeling the individual impact to the whole system at a per-agent level. Recently, QTRAN [17] is proposed to guarantee optimal decentralization inheriting the additive assumption while avoiding representation limitations introduced by VDN and QMIX. However, the constraints of QTRAN on the optimization problem involved are computationally intractable. Authors have to relax these constraints by two penalties thus deviating QTRAN from exact solutions [10].

In this paper, we theoretically derive a decomposing formulation of Q_{tot} by Q^i for cooperative agents. Following the theoretical results, we propose a practical multi-head attention based Q-value mixing network (Qatten) to approximate the global Q-value. Qatten employs the key-value memory operation to explicitly measure the importance of each individual to the global system for transforming individual Q^i's to Q_{tot} within a multi-head attention structure. Experiments on the challenging MARL benchmark show that our method obtains the best performance compared with other popular MARL algorithms. The attention analysis further shows how Qatten measures agents with different weights for mixing Q^i's into Q_{tot} according to each agent's individual properties.

The remainder of this paper is organized as follows. We first introduce the Markov games and attention mechanism in Section 2. Then in Section 3, we explain the explicit formula for local behavior of cooperative MARL and derive the mathematical relation between Q_{tot} and Q^i. Next, in Section 4, we present the framework of Qatten in detail. Furthermore, we validate Qatten in the challenging StarCraft II platform and give the mixing weight analysis in Section 5. Finally, conclusions and future work are provided in Section 6.

2 Background

2.1 Markov Games

Markov games is a multi-agent extension of Markov Decision Processes [8]. They are defined by a set of states, S, action sets for each of N agents, $A^1, ..., A^N$, a state transition function, $T : S \times A^1 \times ... \times A^N \to P(S)$, which defines the probability distribution over possible next states, given the current state and actions for each agent, and a reward function for each agent that also depends on the global state and actions of all agents, $R^i : S \times A^1 \times ... \times A^N \to \mathbb{R}$. We specially consider the partial observed Markov games, in which each agent i receives a local observation $o^i : Z(S, i) \to O^i$. 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. Each agent learns a policy that maximizes its expected discounted returns, $J^i(\pi^i) = \mathbb{E}_{a^1 \sim \pi^1, ..., a^N \sim \pi^N, s \sim T} [\sum_{t=0}^{\infty} \gamma^t r_t^i(s_t, a_t^1, ..., a_t^N)]$, where $\gamma \in [0, 1]$ is the discounted factor. If all agents receive the same rewards ($R^1 = ... = R^N = R$), Markov games becomes fully-cooperative: a best-interest action of one agent is also a best-interest action of others [11]. We consider the

fully-cooperative partially observed Markov games in this paper. When agents make decisions, the history record of agent i's action-observation, τ^i, is usually used to replace the agent i's observation o^i and an Recurrent Neural Network (RNN) is added in the agent's local policy or Q-value function.

For multiagent value decomposition methods such as VDN and QMIX, an important concept for such methods is decentralisability, also called Individual-Global-Max (IGM) [17], which asserts that $\exists Q^i$, such that the following holds:

$$\arg\max_{\vec{a}} Q^*(\vec{\tau}, \vec{a}) = (\arg\max_{a^i} Q^1(\tau^1, a^1)...\arg\max_{a^N} Q^N(\tau^N, a^N)), \tag{1}$$

where $\vec{\tau}$ is the joint agent action-observation histories.

2.2   Attention Mechanism

In recent years, the attention mechanism [20] has been widely used in various research fields and more and more works rely on the idea of the attention to deal with challenges in MAS. For example, MAAC [7] uses the self-attention to learn the critic for each agent by selectively paying attention to information from other agents while TarMAC [3] allows targeted continuous communication between agents via a sender-receiver soft attention mechanism and multiple rounds of collaborative reasoning. An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query V_Q, keys V_K^j, values, and output are all vectors. The output is computed as a weighted sum of values, where each weight w_j assigned to each value is computed by a compatibility function of the query with the corresponding key.

$$w_j = \frac{exp(f(V_Q, V_K^j))}{\sum_k exp(f(V_Q, V_K^k))}, \tag{2}$$

where $f(V_Q, V_K^j))$ is the user-defined function to measure the importance of the corresponding value and the scaled dot-product is a common one. In practice, a multi-head structure is usually employed to allow the model to focus on information from different representation sub-spaces.

3   Theoretical Analysis of Global and Individual Q-values

In this section, we first theoretically derive the expansion formula of each individual Q-value function and then investigate the non-linear combination of multiple individual Q-values into the global one under the multiagent value decomposition framework, which naturally inspires the development of our Qatten model as shown in the next section.

Following the general framework of MARL, the Q-value $Q_{tot}(s, \vec{a})$ is a function on the state vector s and joint actions $\vec{a} = (a^1, ..., a^N)$. By applying the Implicit Function Theorem, Q_{tot} can also be viewed as a function in terms of individual Q-value Q^i:

$$Q_{tot} = Q_{tot}(s, Q^1, Q^2, ..., Q^n), \text{ where } Q^i = Q^i(s, a^i) \approx Q^i(\tau^i, a^i). \tag{3}$$

Without any loss of generality, we assume that there is no 'independent' agent unrelated to the whole group, since an independent agent i should not be considered as a group member, or it can be simply treated as an independent agent to optimize its policy. This means mathematically that the variation of individual Q-value Q^i will have an impact (negative or positive) on the global Q-value Q_{tot}, that is $\frac{\partial Q_{tot}}{\partial Q^i}$ cannot be *identically* zero (whereas admitting *some* zero points is allowed),

$$\frac{\partial Q_{tot}}{\partial Q^i} \neq 0.$$

To simplify our analysis, we investigate the local behaviors of Q_{tot} and Q^i nearly a maximum point $\vec{a}$ in the action space assuming the state s is fixed. Since the gradient $\frac{\partial Q_{tot}}{\partial a^i}$ vanishes at the optimum point $\vec{a}_o$, we have

$$\frac{\partial Q_{tot}}{\partial a^i} = \frac{\partial Q_{tot}}{\partial Q^i} \frac{\partial Q^i}{\partial a^i} = 0. \tag{4}$$

We conclude that

$$\frac{\partial Q^i}{\partial a^i}(a_o) = 0. \tag{5}$$

Consequently, we have local expansion of $Q^i(a^i)$ as follows,

$$Q^i(a^i) = \alpha_i + \beta_i(a^i - a^i_o)^2 + o((a^i - a^i_o)^2). \tag{6}$$

where α_i and β_i are constants. Thereafter, we can theoretically derive that the non-linear dependence of the global Q-value Q_{tot} on individual Q-value Q^i (near a maximal point $\vec{a}_o$) as shown in the following Theorem 1,.

Theorem 1. *Assume that the action space is continuous and there is no independent agent. Then there exist constants $c(s), \lambda_i(s)$ (depending on state s), such that the local expansion of Q_{tot} admits the following form*

$$Q_{tot}(s, \vec{a}) \approx c(s) + \sum_{i,h} \lambda_{i,h}(s) Q^i(s, a^i), \tag{7}$$

where $\lambda_{i,h}$ is a linear functional of all partial derivatives $\frac{\partial^h Q_{tot}}{\partial Q^{i_1} ... \partial Q^{i_h}}$ of order h, and decays super-exponentially fast in h.

A rigorous proof and analysis of Theorem 1 can be found in Appendix A, we briefly cover the proof idea here to keep our presentation intact. At first glance, this theorem may be questioned since Q_{tot} as a functional of all the Q^is absolutely contains non-linear terms, such as $Q^i Q^j$. To address this issue, we multiple $Q^i Q^j$ directly and find that we can reformulate $Q^i Q^j$ with first-order terms Q^i and Q^j. In the multiplication result of $Q^i Q^j$, $\alpha_i \alpha_j$ is a constant term; The term $\alpha_i \beta_j(a^j - a^j_o)^2 = \alpha_i(Q^j - \alpha_j)$ and thus contributes a linear term $\alpha_i Q^j$ and a constant term $-\alpha_i \alpha_j$, and so is the term $\beta_i(a^i - a^i_o)^2 \alpha_j$; Finally, the fourth-order term $\beta_i(a^i - a^i_o)^2 \beta_j(a^j - a^j_o)^2$ is negligible. In this simple computation, the major part of the second-order term $Q^i Q^j$ is the same as the major part of a linear term Q^i and Q^j, other non-linear higher-order terms follow the similar approximating process. Thus, we could just use individual Q-values to effectively approach the global Q-value.

Eq. (7) appears to be linear, yet contains the non-linear information since the head coefficient $\lambda_{i,h}$ is a function of all partial derivatives of order h, and corresponds to all cross terms $Q^{i_1} ... Q^{i_h}$ of order h (e.g. $\lambda_{i,2}$ corresponds to second order non-linearity $Q^i Q^j$). When we implement Theorem 1 to design our multiagent value decomposition network in realistic applications, we propose to adopt the attention mechanism as the universal function approximator [22] to approximate the coefficients $\lambda_{i,h}(s)$ (upto a normalization factor). The Universal Approximation Theorem for Transformer proved in [22] mathematically justifies our choice, where attention mechanism is shown to be the main force of the Transformer's approximation ability. Next, we introduce our practical framework in detail based on the decomposition formation and validate its effectiveness in the experiment section.

4 The Practical Implementation of Qatten

In this section, we propose the Q-value Attention network (Qatten), a practical deep Q-value decomposition network following the above decomposition formation in Eq. (7). Figure 1 illustrates the overall architecture, which consists of agents' recurrent Q-value networks representing each agent's individual value function $Q^i(\tau^i, a^i)$ and the refined attention based value-mixing network to model the agent-level individual impacts while transforming individual Q-values to Q_{tot}. Besides Q^is, the global information (including s and $\vec{u^i}$) is also fed into the attention-based mixing network.

We start from the decomposition formation of Q_{tot} and Q^i. Letting the outer sum over h in Eq.(7), we have

$$Q_{tot} \approx c(s) + \sum_{h=1}^{H} \sum_{i=1}^{N} \lambda_{i,h}(s) Q^i. \tag{8}$$

For each h, the inner weighted sum operation can be implemented using the differentiable key-value memory model [5, 12] to approximate the coefficients and establish the relations from the individuals to the global. As mentioned in the previous section, differentiable key-value memory model admits powerful function approximation capability [22]. This is different from MAAC [7], which uses
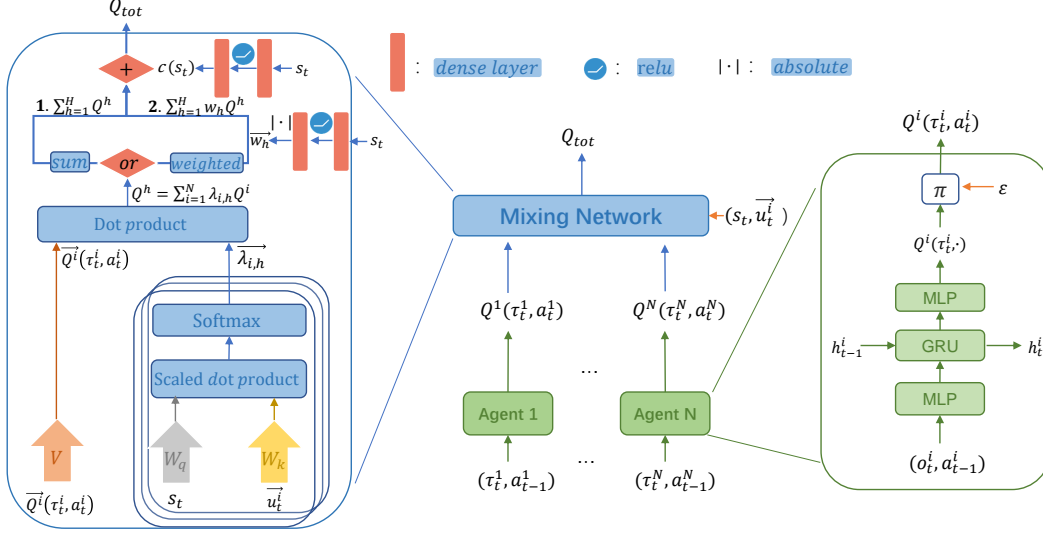
4

Figure 1: The overall architecture of Qatten. The right is agent i's recurrent deep Q-network, which receives the action-observation history record τ^i (last hidden states h_{t-1}^i, current local observations o_t^i and last action a_{t-1}^i). The left is the mixing network of Qatten, which mixes $\vec{Q}^i(\tau_t^i, a_t^i)$ together with s_t and $\vec{u_t^i}$. In general, s is the global state and u^i is the agent i's individual features like its position.

the self-attention to learn the critic for each agent by selectively paying attention to information from other agents. Here we do not perform self-attention among each pair of agents, but use this mechanism to help the whole system to model each individual agent's impact at a per-agent level. Specifically, we pass the similarity value between the global state's embedding vector $e_s(s)$ and the individual features' embedding vector $e_i(u^i)$ into a softmax.

$$\lambda_{i,h} \propto \exp(e_i^T W_{k,h}^T W_{q,h} e_s), \tag{9}$$

where $W_{q,h}$ transforms e_s into the global query and $W_{k,h}$ transforms e_i into the individual key. The e_s and e_i could be obtained by a one or two-layer embedding transformation for s and u^i. Next, for the outer sum over h, we use multiple attention heads to implement the approximations of different orders of partial derivatives. By summing up the head Q-values Q^h from different heads, we get

$$Q_{tot} \approx c(s) + \sum_{h=1}^H Q^h, \text{ where } Q^h = \sum_{i=1}^N \lambda_{i,h} Q^i. \tag{10}$$

H is the number of attention heads. Lastly, the first term $c(s)$ in Eq. (7) could be learned by a neural network with the global state s as the input.

Qatten naturally holds the monotonicity and then achieves IGM property between Q_{tot} and Q^i.

$$\frac{\partial Q_{tot}}{\partial Q^i} \geq 0, \forall i \in \{1, 2, ..., N\}, \tag{11}$$

Thus, Qatten allows tractable maximization of the joint action-value in off-policy learning and guarantees consistency between the centralized and decentralized policies.

Weighted Head Q-value In previous descriptions, we directly add the Q-value contribution from different heads. To relax the weight boundary limitation imposed by the self-attention and increase Qatten's representation ability, we could assign weights w_h for the Q-values from different heads. To ensure monotonicity, we retrieve these head Q-value weights with an absolute activation function from a two-layer feed-work network f^{NN}, which adjusts head Q-values based on global states s.

$$Q_{tot} \approx c(s) + \sum_{h=1}^H w_h \sum_{i=1}^N \lambda_{i,h} Q^i, \tag{12}$$

where $w_h = |f^{NN}(s)|_h$. Compared with directly summing up head Q-values, the weighted head Q-values relaxes the upper bound and lower bound of Q_{tot} imposed by attention. Note that Eq. (12) still satisfies the formation of Eq. (7) as

$$Q_{tot} \approx c(s) + \sum_{h=1}^{H} w_h \sum_{i=1}^{N} \lambda_{i,h} Q^i = c(s) + \sum_{h,i} w_h \lambda_{i,h} Q^i = c(s) + \sum_{i,h} \hat{\lambda}_{i,h} Q^i, \text{ for each } i \text{ and } h, \hat{\lambda}_{i,h} = w_h \lambda_{i,h}.$$

5 Experimental Evaluation

5.1 Settings

In this section, we evaluate Qatten in the StarCraft II decentralized micromanagement tasks and use StarCraft Multi-Agent Challenge (SMAC) environment [15] as our testbed, which has become a common-used benchmark for evaluating state-of-the-art MARL approaches such as COMA [4], QMIX [14] and QTRAN [17]. We train multiple agents to control allied units respectively to beat the enemy, while the enemy units are controlled by a built-in handcrafted AI. All the results are averaged over 5 runs with different seeds. Training and evaluation schedules such as the testing episode number and training hyper-parameters are kept the same as QMIX in SMAC. For the attention part, the embedding dim for the query (global state s) and key (agent individual features u^i) is 32, and the head number is set to 4. More details are provided in the Appendix.

Table 1: Maps in hard and super hard scenarios.

Name	Ally Units	Enemy Units	Type	Challenge
5m_vs_6m	5 Marines	6 Marines	Asymmetric, Homogeneous	Focusing fire
3s_vs_5z	3 Stalkers	5 Zealots	Asymmetric, Heterogeneous	Kite enemy
2c_vs_64zg	2 Colossi	64 Zerglings	Asymmetric, Heterogeneous	Large action space
bane_vs_bane	4 Banelings, 20 Zerglings	4 Banelings, 20 Zerglings	Symmetric, Heterogeneous	Baneling blasts properly
3s5z_vs_3s6z	3 Stalkers, 5 Zealots	3 Stalkers, 6 Zealots	Asymmetric, Heterogeneous	Medivac absorbs fire
MMM2	1 Medivac, 2 Marauders, 7 Marines	1 Medivac, 3 Marauders, 8 Marines	Asymmetric, Heterogeneous	Circuitous tactics

According to the map characters and learning performance of algorithms, SMAC divides these maps into three levels: easy scenarios, hard scenarios, and super hard scenarios. All map scenarios are of different agent number or agent type. The easy scenarios include 2s_vs_1sc, 2s3z, 3s5z, 1c3s5z and 10m_vs_11m. Here we briefly introduce the hard scenarios and super hard scenarios in Table 1 (the first four maps are hard and the last two are super hard) and more map details are in the Appendix.

5.2 Validation

5.2.1 Easy Scenarios

First, we validate Qatten on the easy scenarios. Table 2 shows the median test win rate of different algorithms. It shows that Qatten achieves competitive performance with QMIX and other popular methods. Qatten could master these easy tasks and also works well in heterogeneous and asymmetric settings. COMA's poor performance may result from the sample inefficient on-policy learning and the unstable baseline because of its naive critic structure [7]. Except for 2s_vs_1sc and 2s3z, IDL's win percentage is quite low as directly using global rewards to update policies brings the non-stationary, which becomes severe when the number of agents increases. QTRAN also performs not well, as the practical relaxations could impede the exactness of its updating [10].

5.2.2 Hard Scenarios and Super Hard Scenarios

Next, we test Qatten on the hard scenarios. Results are presented in Figure 2(a)-2(d) as 25%-75% percentile is shaded. To summarize, these scenarios reflect different challenges, and existing approaches fail to achieve consistent performance while Qatten could consistently gain almost the best performance. For example, in bane_vs_bane which consists of 24 agents, Qatten performs much

Table 2: Median performance of the test win percentage in easy scenarios.

Scenario	Qatten	QMIX	COMA	VDN	IDL	QTRAN
2s_vs_1sc	**100**	**100**	97	**100**	**100**	**100**
2s3z	**97**	**97**	34	**97**	75	83
3s5z	**94**	**94**	0	84	9	13
1c3s5z	**97**	94	23	84	11	67
10m_vs_11m	**97**	94	5	94	19	59

(a) Map 5m_vs_6m   (b) Map 3s_vs_5z   (c) Map bane_vs_bane

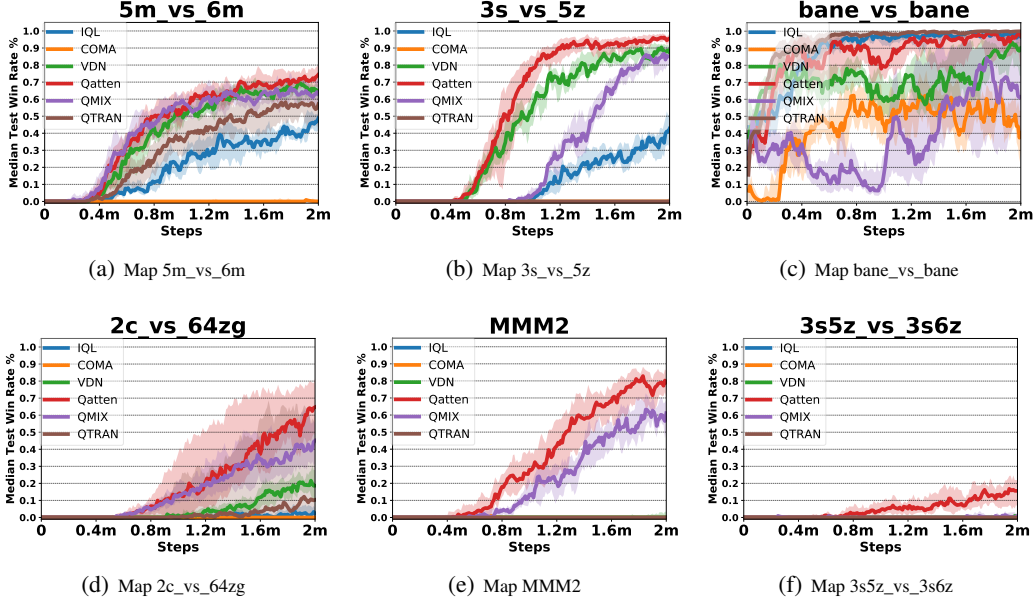(d) Map 2c_vs_64zg   (e) Map MMM2   (f) Map 3s5z_vs_3s6z

Figure 2: Median win percentage on the hard scenarios (a-d) and super hard scenarios (e-f).

better than QMIX which cannot learn steadily. The reason is that the global Q-value heavily depends on the 4 Banelings among 24 agents as they are vital to winning the battle, thus Qatten only needs to concentrate on 4 Banelings while QMIX cannot easily find the appropriate formation for Q_{tot} and Q_i. In the four maps, only Qatten consistently beats all other approaches, which validates the effectiveness of its attention based decomposition structure of Q_{tot} and Q^i.

Finally, we test Qatten on the super hard scenarios as shown in Figure 2(e)-2(f) where 25%-75% percentile is shaded. Due to the difficulty and complexity, we augment Qatten with weighted head Q-values. In MMM2, Qatten masters the winning strategy by heading Medivac to absorb damage and then retreating it and exceeds QMIX by a large margin. The 3s5z_vs_3s6z is another more challenging scenario and all existing approaches fail. In contrast, our approach Qatten can win approximately 16% after 2 million steps of training.

5.3 Ablation Study

We investigate the influence of weighted head Q-values using three difficult scenarios as demonstrated in previous experiments. For clarity, the basic Qatten without weighted head Q-value is called Qatten-base while Qatten with weighted head Q-value is called Qatten-weighted. Figure 3 shows the ablation results and 25%-75% percentile is shaded. As we can see, the weighted head Q-values can leverage the performance of Qatten-base on these difficult scenarios. It shows that this mechanism may capture sophisticated relations between Q_{tot} and Q^i with w_h adjusting head weights flexibly to mitigate the boundedness imposed by the attention. In addition, in MMM2, u^i is replaced by (u^i, Q^i). By the attention's softmax activation, it could enhance the non-linearity of Q_{tot} and Q^i.

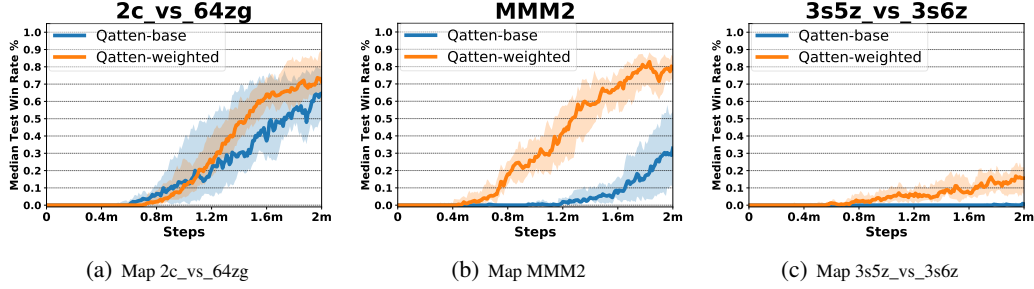(a) Map 2c_vs_64zg (b) Map MMM2 (c) Map 3s5z_vs_3s6z

Figure 3: Ablation study of Qatten on three difficult scenarios.

5.4 Attention Analysis

We further visualize the attention weights ($\lambda_{i,h}$) on each step during a battle. We choose two representative maps 5m_vs_6m and 3s5z_vs_3s6z for illustration and the attention weight heat map of each head is shown in Figure 5.4(a) and Figure 5.4(c) respectively. For 5m_vs_6m, the most important trick to win is to avoid being killed and focusing fire to kill the enemy. From Figure 5.4(a), we see allies have similar attention weights, which means that Qatten indicates an almost equally divided Q_{tot} for agents. This is because each ally marine plays similar roles in this homogeneous scenario. This may also explain the fact that VDN could slightly outperform QMIX when the weights of each Q^i is roughly equal. We also find some tendencies of Qatten. One main tendency is that the only alive unit (agent 0) receives the highest weight in the later steps of the episode. With the attention mechanism adjusting weights of Q^i at a minor difference, Qatten performs better than VDN which assumes the weights of agents' individual Q-values are equal and static during the entire battle.

The weight difference is more apparent on the super hard map 3s5z_vs_3s6z as shown in Figure 5.4(c). VDN and QMIX both fail at this challenging scenario while Qatten has the ability to approximate the sophisticated relations between Q^i and Q_{tot} at the agent level to better perceive each agent's role. For example, at the beginning of the battle, one kind of agents receive more attention than another kind under certain attention heads according to their type features as there exists two kinds of agents. A more detailed analysis based on the game background is in the Appendix C.

(a) 5m_vs_6m Attention Heat Map (b) 5m_vs_6m Agent Health (c) 3s5z_vs_3s6z Attention Heat Map
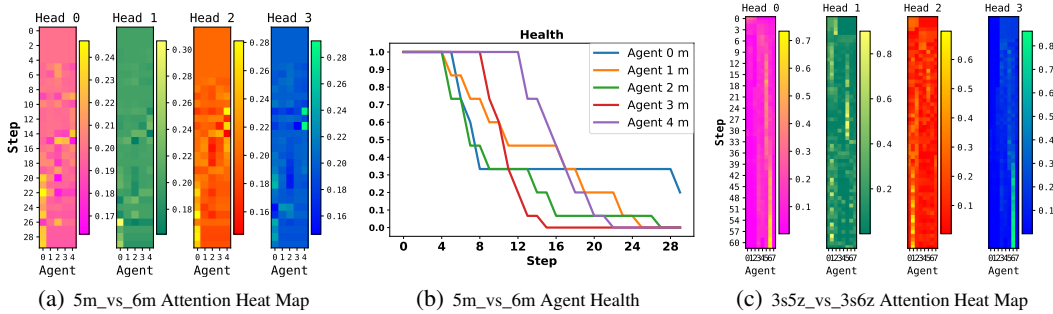
Figure 4: Attention weights on 5m_vs_6m and 3s5z_vs_3s6z. Steps increase from top to the bottom in the attention heat maps. Horizontal ordination indicates the agent id under each head.

6 Conclusion and Future Work

In this paper, we propose the novel Q-value Attention network for the multiagent Q-value decomposition problem. We first perform theoretical analysis of global and individual Q-values and derive their general decomposition formula. Following the decomposition formation, we introduce multi-head attention to establish the mixing network by explicit modeling the individual impact to the whole system at the agent level when transforming individual Q-values to Q_{tot}. Experiments on the chal-

lenging StarCraft MARL benchmark show that our method obtains the best performance on almost all maps and the attention analysis gives the intuitive explanations about the attention weights.

For future works, improving Qatten by combining with explicit exploration mechanisms on difficult MARL tasks is a straightforward direction. Besides, incorporating recent progresses of attention to adapt Qatten into large-scale settings where hundreds of agents exist is also promising.

References

[1] L. Busoniu, R. Babuska, and B. De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics*, 38(2):156–172, 2008.

[2] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9(1):427–438, 2012.

[3] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. TarMAC: Targeted multi-agent communication. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1538–1546, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

[4] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual Multi-Agent Policy Gradients. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.

[5] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural Turing Machines. *arXiv:1410.5401 [cs]*, December 2014. arXiv: 1410.5401.

[6] Jayesh K. Gupta, Maxim Egorov, and Mykel Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *Proceedings of the 16th International Conference on Autonomous Agents and MultiAgent Systems*, pages 66–83, 2017.

[7] Shariq Iqbal and Fei Sha. Actor-Attention-Critic for Multi-Agent Reinforcement Learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2961–2970, Long Beach, California, USA, June 2019. PMLR.

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

[9] 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*, pages 6379–6390. Curran Associates, Inc., 2017.

[10] Anuj Mahajan, Tabish Rashid, Mikayel Samvelyan, and Shimon Whiteson. MAVEN: Multi-Agent Variational Exploration. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7611–7622. Curran Associates, Inc., 2019.

[11] 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.

[12] Junhyuk Oh, Valliappa Chockalingam, Satinder Singh, and Honglak Lee. Control of memory, active perception, and action in minecraft. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, ICML'16, pages 2790–2799. JMLR.org, 2016.

[13] Gregory Palmer, Karl Tuyls, Daan Bloembergen, and Rahul Savani. Lenient multi-agent deep reinforcement learning. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 443–451, 2018.

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

[15] 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 *arXiv:1902.04043 [cs, stat]*, February 2019. arXiv: 1902.04043.

[16] Christian Schroeder de Witt, Jakob Foerster, Gregory Farquhar, Philip Torr, Wendelin Boehmer, and Shimon Whiteson. Multi-Agent Common Knowledge Reinforcement Learning. In *Proceedings of the Advances in Neural Information Processing Systems 32*, pages 9924–9935. Curran Associates, Inc., 2019.

[17] Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. QTRAN: Learning to Factorize with Transformation for Cooperative Multi-Agent Reinforcement Learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5887–5896, Long Beach, California, USA, June 2019. PMLR.

[18] 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*, AAMAS '18, pages 2085–2087, Richland, SC, 2018. International Foundation for Autonomous Agents and Multiagent Systems.

[19] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *In Proceedings of the 10th International Conference on Machine Learning*, pages 330–337. Morgan Kaufmann, 1993.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 5998–6008. Curran Associates, Inc., 2017.

[21] Wang Ying and Dayong Sang. Multi-agent framework for third party logistics in e-commerce. *Expert Systems with Applications*, 29(2):431–436, 2005.

[22] Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

A Proofs

Proof of Theorem 1:

Proof. We expand Q_{tot} in terms of Q^i

$$Q_{tot} = constant + \sum_i \mu_i Q^i + \sum_{ij} \mu_{ij} Q^i Q^j + ... + \sum_{i_1...i_k} \mu_{i_1...i_k} Q^{i_1}...Q^{i_k} + ... \tag{13}$$

where

$$\mu_i = \frac{\partial Q_{tot}}{\partial Q^i}, \mu_{ij} = \frac{1}{2}\frac{\partial^2 Q_{tot}}{\partial Q^i \partial Q^j},$$

and in general

$$\mu_{i_1...i_k} = \frac{1}{k!}\frac{\partial^k Q_{tot}}{\partial Q^{i_1}...\partial Q^{i_k}}.$$

Recall that for each individual Q value, we have local expansion

$$Q^i(a^i) = \alpha_i + \beta_i(a^i - a^i_o)^2 + o((a^i - a^i_o)^2).$$

Now we apply the equation above to the second order term in Equation 13:

$$\sum_{ij} \mu_{ij} Q^i Q^j$$

$$= \sum_{ij} \mu_{ij}(\alpha_i + \beta_i(a^i - a^i_o)^2)(\alpha_j + \beta_j(a^j - a^j_o)^2) + o(\|a - a_o\|^2)$$

$$= \sum_{ij} \mu_{ij}\alpha_i\alpha_j + 2\sum_{ij} \mu_{ij}\alpha_j\beta_i(a^i - a^i_o)^2 + o(\|a - a_o\|)^2$$

$$= \sum_{ij} \mu_{ij}\alpha_i\alpha_j + 2\sum_{ij} \mu_{ij}\alpha_j(Q^i - \alpha_i) + o(\|a - a_o\|)^2$$

$$= -\sum_{ij} \mu_{ij}\alpha_i\alpha_j + 2\sum_{ij} \mu_{ij}\alpha_j Q^i + o(\|a - a_o\|)^2$$

Therefore, we will take

$$\lambda_{i,2} = 2\sum_j \mu_{ij}\alpha_j.$$

In general, we have

$$\mu_{i_1...i_k} = \frac{1}{k!}\frac{\partial^k Q_{tot}}{\partial Q^{i_1}...\partial Q^{i_k}}$$

and

$$\sum_{i_1,...,i_k} \mu_{i_1...i_k} Q^{i_1}...Q^{i_k} = -(k-1)\sum_{i_1,...,i_k} \mu_{i_1...i_k} + k\sum_{i_1,...,i_k} \mu_{i_1...i_k}\alpha_{i_1}...\alpha_{i_{k-1}} Q^{i_k} + o(\|a - a_o\|^2)$$

so we take

$$\lambda_{i,k} = k\sum_{i_1,...,i_k} \mu_{i_1...i_{k-1}i}\alpha_{i_1}...\alpha_{i_{k-1}}.$$

The convergence of the series $\sum_k \lambda_{i,k}$ only requires mild conditions, e.g. boundedness or even small growth of partial derivatives $\frac{\partial^k Q_{tot}}{\partial Q^{i_1}...\partial Q^{i_k}}$ in terms of k. □

We remark when we employ attention mechanism to approximate λ_i, $\lambda_{i,h}$ is modelled as single-head attention. Since $\lambda_{i,h}$ decays fast in h, we stop the series at H (number of heads) for feasible computations in the practical implementation.

B Experimental Settings

We follow the settings of SMAC [15], which could be referred in the SMAC paper. For clarity and completeness, we present these environment details again.

B.1 States and Observations

At each time step, agents receive local observations within their field of view. This encompasses information about the map within a circular area around each unit with a radius equal to the sight range, which is set to 9. The sight range makes the environment partially observable for agents. An agent can only observe others if they are both alive and located within its sight range. Hence, there is no way for agents to distinguish whether their teammates are far away or dead. If one unit (both for allies and enemies) is dead or unseen from another agent's observation, then its unit feature vector is reset to all zeros. The feature vector observed by each agent contains the following attributes for both allied and enemy units within the sight range: distance, relative x, relative y, health, shield, and unit type. If agents are homogeneous, the unit type feature will be omitted. All Protos units have shields, which serve as a source of protection to offset damage and can regenerate if no new damage is received. Lastly, agents can observe the terrain features surrounding them, in particular, the values of eight points at a fixed radius indicating height and walkability.

The global state is composed of the joint unit features of both ally and enemy soldiers. Specifically, the state vector includes the coordinates of all agents relative to the centre of the map, together with unit features present in the observations. Additionally, the state stores the energy/cooldown of the allied units based the unit property, which represents the minimum delay between attacks/healing. All features, both in the global state and in individual observations of agents, are normalized by their maximum values. In Qatten, the agent i's individual features u^i include its own position (relative coordination to the central point), health, cooldown, the possible shield (if available) and the possible type feature (in heterogeneous scenarios), which is the part of the global state.

B.2 Action Space

The discrete set of actions which agents are allowed to take consists of move[direction], attack[enemy id], stop and no-op. Dead agents can only take no-op action while live agents cannot. Agents can only move with a fixed movement amount 2 in four directions: north, south, east, or west. To ensure decentralization of the task, agents are restricted to use the attack[enemy id] action only towards enemies in their shooting range. This additionally constrains the ability of the units to use the built-in attack-move macro-actions on the enemies that are far away. The shooting range is set to be 6 for all agents. Having a larger sight range than a shooting range allows agents to make use of the move commands before starting to fire. The unit behavior of automatically responding to enemy fire without being explicitly ordered is also disabled. As healer units, Medivacs use heal[agent_id] actions instead of attack[enemy_id].

B.3 Rewards

At each time step, the agents receive a joint reward equal to the total damage dealt on the enemy units. In addition, agents receive a bonus of 10 points after killing each opponent, and 200 points after killing all opponents for winning the battle. The rewards are scaled so that the maximum cumulative reward achievable in each scenario is around 20.

B.4 Training Configurations

The training time is ranging from about 8 hours to 18 hours on these maps (GPU Nvidia RTX 2080 and CPU AMD Ryzen Threadripper 2920X), which is based on the agent numbers and map features of each map. The number of the total training steps is about 2 million and every 10 thousand steps we train and test the model. When training, a batch of 32 episodes are retrieved from the replay buffer which contains the most recent 5000 episodes. We use ε-greedy policy for exploration. The starting exploration rate is set to 1 and the end exploration rate is 0.05. Exploration rate decays linearly at the first 50 thousand steps. We keep the default configurations of environment parameters.

B.5 Mixing Network Hyper-parameters

We adopt the Python MARL framework (PyMARL) [15] on the github to develop our algorithm. The hyper-parameters of training and testing configurations are the same as in SMAC [15] and could be referred in the source codes. Here we list the parameters of Qatten's mixing network in Table 3.

Table 3: The network configurations of Qatten's mixing network.

Qatten mixing network configurations	Value
Query embedding layer number	2
Unit number in query embedding layer 1	64
Activation after query embedding layer 1	Relu
Unit number in query embedding layer 2	32
Key embedding layer number	1
Unit number in key embedding layer 1	32
Head weight layer number	2
Unit number in head weight layer 1	64
Activation after head weight layer 1	Relu
Unit number in head weight layer 2	4
Attention head number	4
Constant value layer number	2
Unit number in constant value layer 1	32
Activation after constant value layer 1	Relu
Unit number in constant value layer 2	1

B.6 Win Percentage Table of All Maps

We here give the median win rates of all methods on the scenarios presented in our paper. In MMM2 and 3s5z_vs_3s6z, we augment Qatten with the weighted head Q-values. Other maps are reported by the win rates based on the original Qatten (Qatten-base). We could see Qatten obtains the best performance on almost all the map scenarios.

Table 4: Median performance of the test win percentage.

Scenario	Qatten	QMIX	COMA	VDN	IDL	QTRAN
2s_vs_1sc	**100**	**100**	97	**100**	**100**	**100**
2s3z	**97**	**97**	34	**97**	75	83
3s5z	**94**	**94**	0	84	9	13
1c3s5z	**97**	94	23	84	11	67
5m_vs_6m	**74**	63	0	63	49	57
3s_vs_5z	**96**	85	0	87	43	0
bane_vs_bane	97	62	40	90	97	**100**
2c_vs_64zg	**65**	45	0	19	2	10
MMM2	**79**	61	0	0	0	0
3s5z_vs_3s6z	**16**	1	0	0	0	0

In the hard scenarios, 5m_vs_6m requires the precise control such as focusing fire and consistent walking to win. In 3s_vs_5z, since Zealots counter Stalkers, the only winning strategy is to kite the enemy around the map and kill them one after another. In 2c_vs_64zg, 64 enemy units make the action space of the agents the largest among all scenarios. In bane_vs_bane, the strategy of winning is to correctly use the Banelings to destroy as many enemies as possible. For super hard scenarios, the key winning strategy of MMM2 is that the Medivac heads to enemies first to absorbing fire and then retreats to heal the right ally with the least health. As the most difficult scenarios, 3s5z_vs_3s6z may require a better exploration mechanism while training.

C   Extended Attention Analysis

For 5m_vs_6m, the most important trick to win is to avoid being killed and focusing fire to kill enemy. From Figure 5(a), we see allies have similar attention weights, which means that Qatten indicates an almost equally divided Q_{tot} for agents. This is because each ally marine plays similar roles on this homogeneous scenario. This may also explain the fact that VDN could slightly outperform QMIX when the weights of each Q^i is roughly equal. But we still find some tendencies of Qatten. One main tendency is that, the only alive unit (agent 0) receives the highest weight in the later steps of the episode, which encourages agents to survive for winning. With the refined attention mechanism adjusting weights of Q^i with minor difference, Qatten performs the best.

We analyse another map 3s5z_vs_3s6z and the attention heat map is shown in Figure 5(c). After analyzing the correlation of attention weights and agent features, we notice that Head 1 focuses on the output damage. Units with high CoolDown (indicating just fired) receives high attention values. Head 1 and 2 tend to focus on the Stalker agents (Agent 0, 1 and 2) while Stalkers (Agent 3-7) almost receives no attention at the beginning. While Stalker agent 1 with high health receives attention from head 1 and 2, Zealot agent 6 with high health receives attention from head 0 and 3. In summary, attention weights on map 3s5z_vs_3s6z are changing violently. As the battle involves highly fluctuant dynamics, VDN which statically decomposes Q_{tot} equally thus cannot adapt to the complex scenarios. Similarly, QMIX with a rough and implicit approximation way also fails. In contrast, Qatten could approximate the sophisticated relations between Q^i and Q_{tot} with different attention heads to capture different features in sub-spaces.

(a) 5m_vs_6m Attention Heat Map

(b) 5m_vs_6m Agent Property

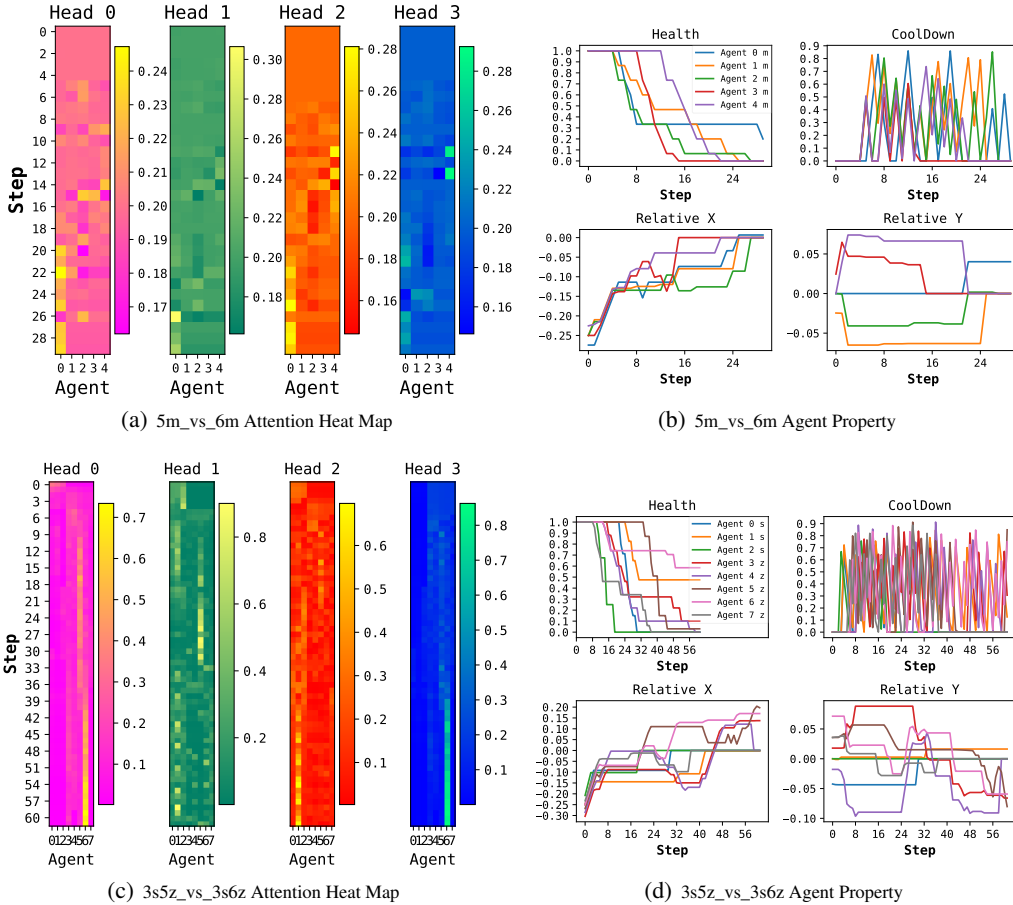(c) 3s5z_vs_3s6z Attention Heat Map

(d) 3s5z_vs_3s6z Agent Property

Figure 5: Attention weights on 5m_vs_6m and 3s5z_vs_3s6z. Steps increase from top to the bottom in the attention heat maps. Horizontal ordination indicates the agent id under each head.