Attention-Aware Actor for Cooperative Multi-agent Reinforcement Learning

Chenran Zhao[1], Dianxi Shi[1,2,3(✉)], Yaowen Zhang[4], Yaqianwen Su[2], Yongjun Zhang[2], and Shaowu Yang[1]

[1] National University of Defense Technology, Changsha 410073, China
dxshi@nudt.edu.cn
[2] Artificial Intelligence Research Center (AIRC), National Innovation Institute of Defense Technology (NIIDT), Beijing 100166, China
[3] Tianjin Artificial Intelligence Innovation Center (TAIIC), Tianjin 300457, China
[4] Environment Information Support 32282 Research Institute, Jinan 250000, China

Abstract. In multi-agent environments, cooperation is crucially important, and the key is to understand the mutual interplay between agents. However, multi-agent environments are highly dynamic, where the complex relationships between agents cause great difficulty for policy learning, and it's costly to take all coagents into consideration. Besides, agents may not be allowed to share their information with other agents due to communication restrictions or privacy issues, making it more difficult to understand each other. To tackle these difficulties, we propose Attention-Aware Actor (Tri-A), where the graph-based attention mechanism adapts to the dynamics of the mutual interplay of the multi-agent environment. The graph kernels capture the relations between agents, including cooperation and confrontation, within local observation without information exchange between agents or centralized processing, promoting better decision-making of each coagent in a decentralized way. The refined observations produced by attention-aware actors are exploited to learn to focus more on surrounding agents, which makes Tri-A act as a plug for existing multi-agent reinforcement learning (MARL) methods to improve the learning performance. Empirically, we show that our method substantially achieves significant improvement in a variety of algorithms.

Keywords: Multi-agent cooperation · Graph · Attention mechanism · Multi-agent mutual interplay

1 Introduction

Cooperation is a widespread social behavior in nature, where humans tend to focus on the most relevant feature area such as objects [3] and faces [9] for understanding the mutual interplay between neighboring humans, allowing them to

This work was supported by the National Key Research and Development Program of China (2017YFB1001901).

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 407, pp. 370–384, 2021.
https://doi.org/10.1007/978-3-030-92638-0_22

quickly process most relevant parts of the local observations during decentralized decision making [21]. This makes humans exceed all other species in terms of range and scale of cooperation.

Attention mechanism [20] has recently emerged as a successful approach in various fields. The main idea behind it is to select salient areas of interest which are more relevant to the current task from the numerous information according to the importance distribution of elements, and it has shown great success in mutual interplay understanding of MARL, examples include a wide range of approaches on designing effective attention-based MARL approaches, which can mainly be divided into two categories, exchanging local perceptual information on the field of view of each agent [6,7,18,23] and modeling how agents cooperate through a centralized critic with the attention mechanism [5,13]. However, most of these methods depend on either the existence of a communication message or the centralized attention mechanism which could access to the global information that contains the information of each agent on the field. Unfortunately, when the agents are independently deployed and communications are disabled or prohibitive, each agent has to predict its own action conditioning on its partial observation trajectory. Besides, typical attention-based approaches to MARL either consider all coagents at all time steps during the decentralized execution or take all these dynamics into account during the centralized training. It is costly and less helpful to take all other coagents into consideration whether in a decentralized or centralized manner, because receiving a large amount of information incurs high computational complexity.

To tackle these difficulties, we resort to the decentralized attention-based model that does not need to assume the existence of perfect communication or consider all agents all the time. Specifically, we are interested in building reinforcement learning (RL) agents, which learn representations guided by an understanding of what is important during multi-agent cooperation and confrontation for decision making. More importantly, since the reconstructed observation has the same form as that of the original observation, it is straightforward to plug in any existing MARL algorithms for decision making.

In this paper, we propose a graph-based attention module that is designed for understanding the mutual interplay between agents (including cooperation and confrontation) during the decentralized execution, called Attention-Aware Actor (Tri-A). Different from previous approaches, Tri-A learns an actor for each coagent that could selectively focus on neighboring agents from local observation, which helps achieve the cooperative goal with decentralized models. The intuition behind our idea comes from the fact that, in many real-world environments, each agent can observe other agents (both enemies and friends) in the field of view of each agent itself, and it is beneficial for each agent to know which one it should pay attention to or cooperate with at different stages of a complete trajectory of the battle. For example, a soccer attacker with the ball needs to pay attention to the opposing team's defenders as well as neighboring teammates, while she/he rarely needs to pay attention to her/his own team's goalkeeper. The specific players that the attacker is paying attention to varies over the complete

trajectory of the game, depending on the situation on the field, especially the formation and strategy of the opponent team and its own team. By applying soft attention to the mutual interplay graph of each agent, our Tri-A is able to make decisions based on the situation on the field dynamically, such as selecting which coagents to attend to or avoiding attacks from enemies, improving performance in multi-agent domains with complex interactions.

We evaluate Tri-A on various MARL algorithms by replacing the original actor with the attention-aware actor (Tri-A). Our experiments on a range of unit micromanagement tasks built in StarCraftII [17] show that our method obtains significant improvement. The attention analysis shows that Tri-A can capture proper target agents that each coagent should pay more attention to as well as proper importance weight of each target agent at each time step t, which reveals the mutual interplay between agents, including cooperation and confrontation.

Our main contributions can be summarized as follows:

- **Attention-based graph structure.** We construct the multi-agent mutual interplay (including cooperation and confrontation) as a graph according to the local observation of each agent. It refines the local observations of each agent by applying the attention mechanism on the different parts of the local observations so as to make agents selectively and dynamically focus on the surrounding agents. Furthermore, it can solve the problem of high computational complexity which exists in the approaches that taking the perceived information from all agents into consideration.
- **Combining with actor under value decomposition architecture.** By combining the graph structure with actor, we propose a novel actor called Attention-Aware Actor (Tri-A) which could be used as a plug-in for any Value Decomposition (VD) architecture algorithms. Since the actor under the architecture of value decomposition is updated with the backpropagation of the upper network structure, we can replace the actor at will without having to design a loss function specifically for it. It improves the performance of making decisions in multi-agent domains with complex interactions.

The remainder of this paper is organized as follows. We first introduce the Markov games, attention mechanism, attnetion-based MARL algorithms and graph network in Sect. 2. Then in Sect. 3, we present the framework of Tri-A in details. Furthermore, we validate Tri-A in the challenging StarCraftII platform and give the specific analysis in Sect. 4. Finally, conclusions are provided in Sect. 5.

2 Background

A fully cooperative multi-agent task can be described as a Dec-POMDP [15] consisting of a tuple $G = \langle \mathcal{N}, S, A, P, Z, O, r, n, \gamma \rangle$. $s \in S$ describes the true state of the environment which contains the global information of all agents on the field. At each time step, each agent $i \in \mathcal{N} \equiv \{1, ..., n\}$ chooses an action $a^i \in A$,

forming a joint action $\mathbf{a} \in \mathbf{A} \equiv A^n$. This causes a transition on the environment according to the state transition function $P(s'|s, \mathbf{a}) : S \times \mathbf{A} \times S \to [0, 1]$. All agents share the same reward function $r(s, \mathbf{a}) : S \times \mathbf{A} \to \mathbb{R}$ and $\gamma \in [0, 1)$ is a discount factor.

We specially consider a partially observable scenario in which each agent i draws local observations $o^i \in O$ instead of global state s according to the observation probability function $Z(o, a) : O \times A \to O$. Each agent i has an observation-action history $\tau^i \in T \equiv (Z \times A)^*$, on which it conditions a stochastic policy $\pi^i(a^i|\tau^i) : T \times A \to [0, 1]$, aiming at maximizing global rewards. The joint policy π has a joint action-value function: $Q^\pi(s_t, \mathbf{a}_t) = \mathbb{E}_{s_{t+1:\infty}, \mathbf{a}_{t+1:\infty}}[R_t|s_t, \mathbf{a}_t]$, where $R_t = \sum_{j=0}^{\infty} \gamma^j r_{t+j}$ is the discounted return.

2.1 Attention Mechanism

In recent years, attention mechanism [20] has made remarkable performance in various domains, including computer vision [1,14], NLP [2,10,20] and RL [4,8,11]. More and more work relies on the idea of the attention to deal with challenges of multi-agent cooperation. 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 the values, where each weight w_j assigned to each value is computed by a compatibility function of the query with the corresponding key.

$$w_j = \frac{\exp(f(V_Q, V_K^j))}{\sum_k \exp(f(V_Q, V_K^k))}. \tag{1}$$

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, the multi-head structure is usually employed to allow the model to focus on information from different representation sub-spaces.

2.2 Attention-Based Algorithms for MARL

ATOC [8] follows the alternative paradigm of centralizing policies while keeping the critics decentralized. Their focus is on learning an attention model for sharing information between the policies.

MAAC [5] applied the soft attention mechanism to the centralized critic so as to dynamically select which agents to attend to at each time step for each agent during the training, which improves the scalability and performance in multi-agent domains with complex interactions.

MADDPG [13] extends DDPG to multi-agent settings. It proposes a multi-agent policy-gradient algorithm using central critics. The centrally computed critic network takes the states and actions of all other agents as its input, and the actor network only relies on the local observation information of a given agent.

Compared with them, which assume the existence of effective communication or take all agents into consideration during the centralized training, our work applies the attention mechanism to the local observation of each coagent so that each coagent can pay more attention to neighboring agents when making decisions, which achieves the cooperative goal with decentralized models in the case of limited communication.

2.3 Graph Network

DGN [7] assumes a graph connection among the agents such that each node is an agent. A multi-head attention model is used as the convolution kernel to extract the connection weights to the neighbor nodes. Agents can share their observations and weights of all agents with their neighbors. In other words, each agent constructs a graph using the perceived information exchanged from others. It means each agent needs to process a large amount of information, which may incur high computational complexity.

G2ANET [12] represents all agents as a complete graph based on a two-stage attention network, where hard attention is used to cut the unrelated edges and soft attention is used to learn the importance weight of the edges. By combining the graph with the policy network, each agent considers the communication vectors of all other agents when making decisions. By combining the graph with the Q-value network, the critic of each agent considers the state and action information of all other agents when calculating the individual Q-value. However, both of them need to process a large amount of information received from other agents.

Different from them, we propose to construct a graph using only the local observations of each agent. In this way, the amount of information each agent needs to process is unchanged. Thus it may not incur high computational complexity. Besides, it explicitly models how agents cooperate and confront within local observations, thus promoting better decision-making.

3 Our Approach

In this section, we describe the proposed method Attention-Aware Actor (Tri-A) for learning actors in a graph-based manner. We construct the multi-agent mutual interplay as a graph according to the local observation of each agent and refine the local observation through a graph-based attention mechanism. It establishes connections between the individual agent and its surrounding agents so that each agent could focus on other agents in its vicinity selectively and dynamically, promoting better decision making of each agent in a decentralized way.

In order to figure out how Tri-A as a plug works in cooperative MARL, we choose a representative algorithm architecture called Value Decomposition (VD) for illustration, and the whole structure is shown in Fig. 1. Figure 1 provides a clear explanation that value decomposition is aimed at decomposing the global

shared multi-agent Q-value Q^{tot} into individual Q-values Q^i to guide individuals' behaviors. Actors from the architecture of value decomposition make decisions based on local observations and generate the individual Q-value $Q^i(o^i, a^i)$ of the agent. More importantly, since the architecture of value decomposition takes the setting of end-to-end, we don't need to design a loss function for the attention-aware actor (Tri-A) because the backpropagation of value decomposition will update the attention-aware actor of each agent naturally.

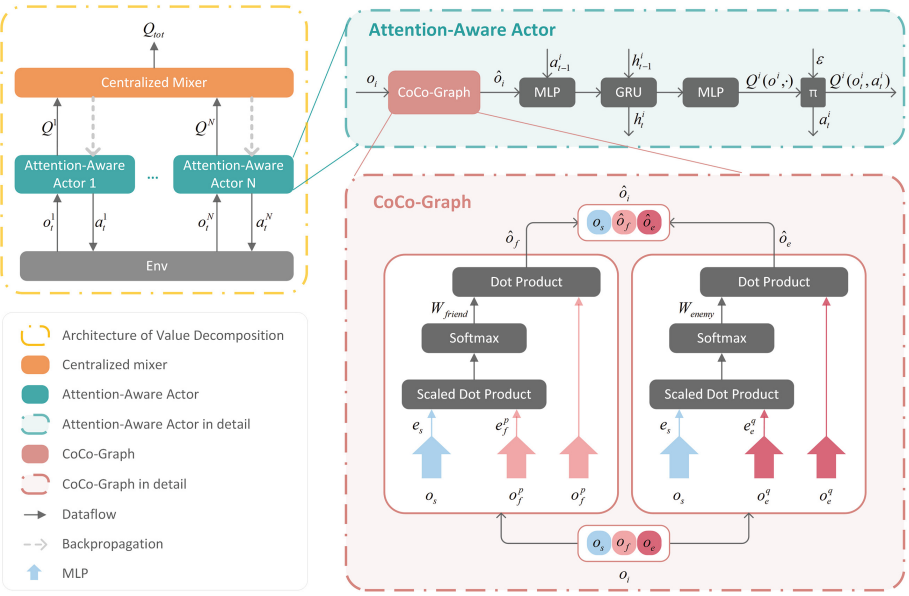


Fig. 1. The overall structure of Attention-Aware Actor. The Attention-Aware Actor is composed of CoCo-Graph architecture and RNN policy network. The left is the overall structure of value decomposition which mixes Q^i into Q^{tot}. The pink part is the CoCo-Graph architecture that reconstruct the local observations o_i into targeted observations $\hat{o}_i$ through soft attention mechanism. The green part is the RNN policy networks which receives last hidden states h^i_{t-1}, last action a^i_{t-1} and current reconstructed local observations $\hat{o}_i$ as inputs. The reconstructed local observation produced by CoCo-Graph architecture will be input into RNN policy networks for decision making or individual Q-value calculating. (Color figure online)

3.1 Multi-agent Mutual Interplay Graph Structure

We construct the multi-agent mutual interplay, including cooperation and confrontation, as a graph according to the local observation of each agent. Agents in the environment are represented by the nodes of the graph, and each node has a set of neighbors (including friends and enemies) which is determined by

distance or other metrics, depending on the environment, and varies over time. The intuition behind it is that neighboring agents are more likely to interact with and affect each other. Thus each agent should adjust its strategy dynamically by observing the movements of the surrounding agents to promote better decision-making. Besides, as the number and position of agents vary over time, the cooperation relationship and confrontation relationship are changing continuously and violently during the battle, and it's vital for us to extract these relationships adaptively. Moreover, since it is costly and less helpful to take all agents on the field into consideration, we assume that there's no effective communication between agents, and each node can only reason about the relationships with each other from its local observations. Inspired by the principle above, we model the relationship between each coagent and its surrounding agents, including cooperation and confrontation, as a CoCo-Graph, and we illustrate it in Fig. 2.

Definition 1. *(CoCo-Graph) The relationship between coagent i and its surrounding agents is defined as a directed graph as $G_i = (N_f, N_e, E_f, E_e)$ which is consisting of the set N_f of friend nodes, the set N_e of enemy nodes, the set E_f of edges which are ordered pairs from i to N_f and the set E_e of edges which are ordered pairs from i to N_e. Each friend node p (for all friends indexed by $p \in \{1, ..., n-1\}$) represents a friend agent entry. Each enemy node q (for all enemies indexed by $q \in \{1, ..., m\}$) represents an enemy agent entry. If i tends to cooperate with friend p, then there is an edge from i to p. If i tends to confront enemy q, then there is an edge from i to q.*

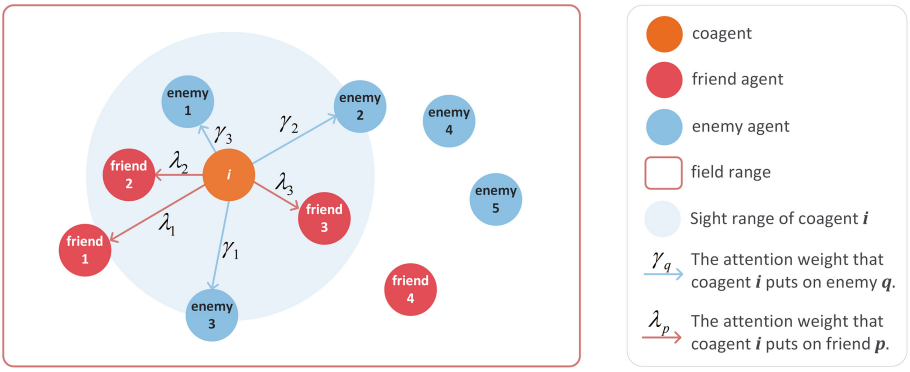


Fig. 2. CoCo-Graph G_i of coagent i

As illustrated in Fig. 2, each coagent i constructs a CoCo-Graph G_i based on its local observation at each time step t during the complete trajectory of game. The CoCo-Graph G_i is closely connected with agents within coagent i's sight range, and the agents out of sight affect coagent i little. By reconstructing

the local observation of each coagent i through CoCo-Graph G_i, the observation is potentially simplified for understanding and reasoning, which is indeed helpful for understanding of the mutual interplay between agents. We then show how the connections between agents can be extracted by the designed attention architecture below.

3.2 Attention-Aware Actor Architecture

In this section, we show how Tri-A can be plugged in existing MARL methods to improve policy learning. Our aim is to learn a graph that indicates the mutual interplay between agents and the potential of each neighboring agent being the target to cooperate with or confront with. Based on the CoCo-Graph introduced above, we design a soft attention architecture to reconstruct local observations of each coagent, which could ideally differentiate between the interested agents and connected but less important agents. More specifically, we train a soft attention model to learn the weights of each coagent i focusing on neighboring agents according to its CoCo-Graph G_i at each time step t during the complete trajectory of the game.

Just like football players can observe the movements of other players on the field, each agent can observe other agents if they are both alive and located within the sight, including friends and enemies. Therefore we can use the information within local observation to help build the embedding vectors of both friends and enemies. First, we separate the local observation o of coagent i into three parts according to the prior knowledge: o_e represents enemy information, o_f represents friend information and o_s represents the information about coagent$i$ itself, as Eq. (2) shows (we drop the parameter dependence wherever its inferable for clarity of presentation).

$$o = (o_e, o_f, o_s). \tag{2}$$

where $\hat{o_e} = \sum_{q=1}^{m} \gamma_q \cdot e_a^q$ and $\hat{o_f} = \sum_{p=1}^{n} \mu_k \cdot e_f^p$, which means o_f is composed of n friends' information o_f^p (for all friends indexed by $p \in \{1, ..., n\}$) and o_e is composed of m enemies' information o_e^q (for all enemies indexed by $q \in \{1, ..., m\}$).

Then we encode the information about coagent i itself, friend agent p and enemy agent q in observation o into embedding vectors e_s, e_f^p and e_e^q respectively by MLP as shown in Fig. 1 at each time step t. Now we can use these embedding vectors to learn the weights of edges in CoCo-Graph G_i ($i \in \{1, ..., n\}$). We calculate and normalize the correlation between e_s and e_f^p, and between e_s and e_e^q within the local observations of each agent through soft attention mechanism according to Eq. (3) and Eq. (4) respectively. On the one hand, we hope that each agent could pay attention to surrounding agents, including both enemies and friends. On the other hand, we hope that agents could understand the mutual interplay between each other by holding targeted observations, and thereby promote cooperation.

$$\lambda_p = \frac{\exp(f(e_s, e_f^p))}{\sum_{k=1}^{n} \exp(f(e_s, e_f^k))}. \tag{3}$$

$$\gamma_q = \frac{\exp(f(e_s, e_e^q))}{\sum_{k=1}^{m} \exp(f(e_s, e_e^k))}. \tag{4}$$

Then we carry out the information integration by concatenating all parts of the local observations according to Eq. (5), where the friend information o_f and the enemy information are refined through CoCo-Graph. The reconstructed local observations are then fed into the RNN policy networks to strategize. The reconstructed local observations have considered the complex interaction between agents, which is indeed helpful for holding targeted observations and understanding the mutual interplay between agents, and thus promote better cooperation naturally.

$$\hat{o} = (\hat{o_e}, \hat{o_a}, o_s). \tag{5}$$

Since the reconstructed observation has the same form as that of the original observation, it is straightforward to plug it in any existing deep RL methods for decision making.

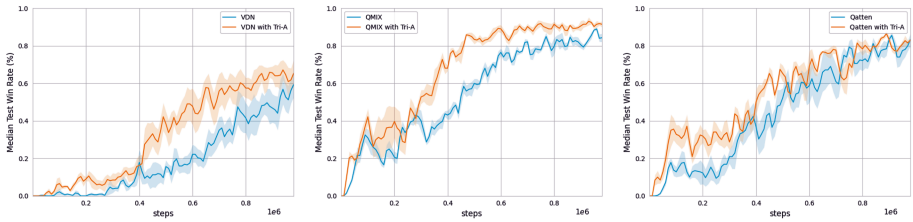
4 Experimental Evaluation

4.1 Settings

The StarCraft Multi-Agent Challenge (SMAC) environment [17] have a rich set of scenarios that allow complex interactions between agents including cooperation and confrontation, so we evaluate Tri-A in the SMAC environment to demonstrate the efficacy of our method. In this environment, each agent is a unit participating in combat against enemy units which is controlled by hand-crafted policies. Besides, agents receive individual local observation from the environment, containing distance, relative location, health, shield, and unit type of other allied and enemy units within their sight range. We consider SMAC maps with three different scenarios, here we briefly introduce the scenarios in Table 1. Training and evaluation schedules such as the testing episode number and training hyper-parameters are kept the same as QMIX [16] in SMAC. The percentage of episodes where the agents defeat all enemy units, i.e., test win rate, is reported as the performance of algorithms. All the results are averaged over 5 independent runs with different random seeds. We show that the RL agents equipped with Tri-A perform better in both convergence speed and the final test win rate than baseline algorithms. What's more, we demonstrate that Tri-A as a plug could be applied across different algorithms, showcasing the generalization ability.

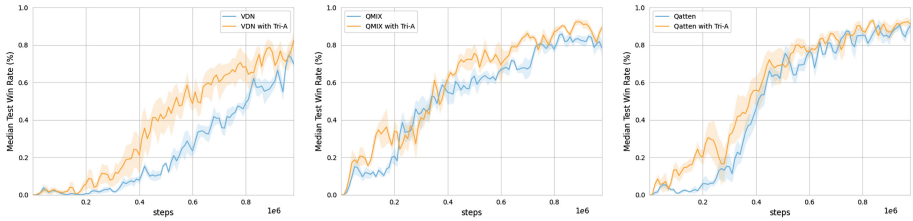
Table 1. SMAC maps in three different scenarios.

Name	Ally Units	Enemy Units	Type
3s5z	3 Zealots 5 Stalkers	3 Zealots 5 Stalkers	Symmetric & Heterogeneous
1s3s5z	1 Colossi 3 Zealots	1 Colossi 3 Zealots	Symmetric & Heterogeneous
	5 Stalkers	5 Stalkers	
5m_vs_6m	5 Marines	6 Marines	Asymmetric & Homogeneous



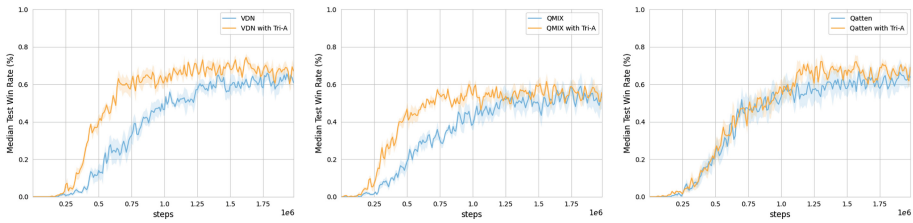
(a) VDN (b) QMIX (c) Qatten

Fig. 3. Median test win rate on map 3s5z



(a) VDN (b) QMIX (c) Qatten

Fig. 4. Median test win rate on map 1c3s5z



(a) VDN (b) QMIX (c) Qatten

Fig. 5. Median test win rate on map 5m_vs_6m

4.2 Validation

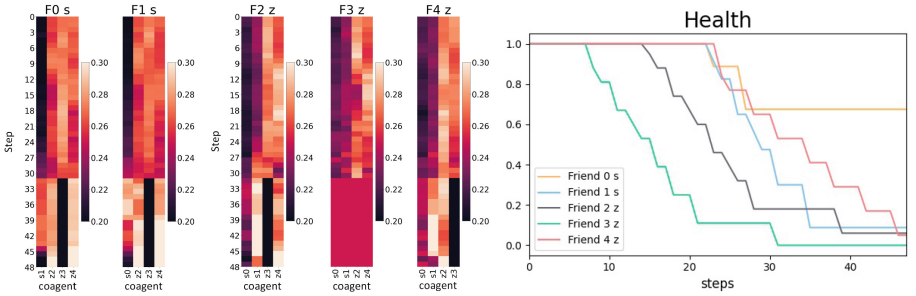
We apply Tri-A on three centralized training with decentralized execution (CTDE) setting algorithms that adopts the architecture of Value Decomposition (VD), including VDN [19], QMIX [16] and Qatten [22], where agents make decisions based on local observations within decentralized actors. Since the actor under the architecture of value decomposition is updated with the backpropagation of the global shared multi-agent Q-value Q^{tot}, we can replace the actor of these algorithms at will without having to design a loss function specifically for the actor. Figure 3, Fig. 4 and Fig. 5 have shown the median test win rate of baseline algorithms (VDN, QMIX and Qatten) equipped with Tri-A and baseline algorithms equipped with original actors in three different scenarios. We observe that there is a significant gap between the baseline algorithms and baseline algorithms equipped with Tri-A in all scenarios. By constructing the CoCo-Graph within local observation for each coagent and focusing on neighboring agents, the observation is potentially simplified for understanding and reasoning, and coagents learn faster and perform consistently better than the baseline algorithms. The qualitative performance indicates that Tri-A is indeed helpful for the understanding of the mutual interplay between agents, and thus promotes better decision making for each coagent.

4.3 Attention Analysis

Next, we visualize the attention weights of each agent focusing on friends and enemies at each time step during a complete trajectory of battle to figure out what has been learned in the attention mechanism and how much effect the attention mechanism actually contributes to the decision making in Tri-A. For better understanding, we choose the 2s3z map to illustrate the attention weight heatmap of coagents focusing on other agents is shown in Fig. 6 and Fig. 8. We also attach some auxiliary snapshots in Fig. 7 to explain some interesting segments in the heatmaps. In all the snapshots, the red colored units indicate the agents controlled by Tri-A.

For 2s3z, the two armies are composed of the same units, 2 Zealots and 3 Stalkers, among which strong Stalkers are much more effective against enemies, so it is of great importance for Stalkers to learn to protect teammates that are vulnerable to certain enemies' attack, and the fragile little Zealots should learn to stick together and focus fire, ordering units to jointly attack and kill enemy units one after another. Besides, a very important technique is making enemy units give chase while maintaining enough distance so that little or no damage is incurred.

Attention Analysis of Attention-Aware Actor Focusing on Friends. For qualitative analysis, we visualize the attention weights (λ_i) of each coagent i focusing on other friend agents at each time step in a complete trajectory of 2s3z battle as shown in Fig. 6.



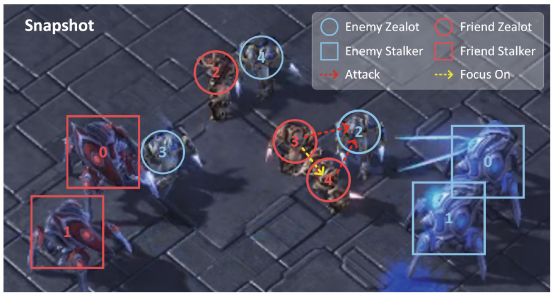
(a) Focus on friends (b) Friends' health

Fig. 6. (a) The attention heatmap of coagents focusing on other friend agents during a complete trajectory of 2s3z battle. The head title of each heatmap indicates the unique mark of a specific agent A from our team, including agent camp (F means Friend), agent id (from 0 to 4) and agent type (s means Stalker, z means Zealot). Bottom horizontal ordination indicates the unique mark of other friend agents that A is focusing on, including agent id and agent type. Vertical ordination represents time steps that increase from top to bottom. (b) The health property of each coagent, which indicates the remaining blood.

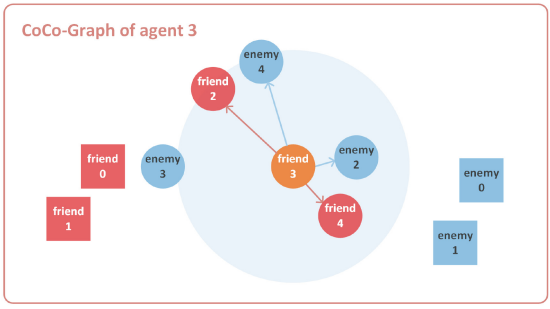
Zealot has short range and short skill cool down, Stalker has a long range and long skill cool down. Therefore, it's beneficial for Zealot units to assemble into formations and fight at the front so that they can focus fire on enemies and destroy the enemy one by one, and it's proper for Stalker units to fight at the back to protect Zealot units from enemies' attack. As revealed by Fig. 6(a), Stalkers always pay more attention to fragile Zealots while Zealots also pay more attention to the other two friend Zealots, which exactly confirms what we have analyzed above.

Besides, as the health property shows in Fig. 6(b), friend Zealot 3 is dead at time step 31. We can tell that the attention weights in friend Zealot 3's heatmap in Fig. 6(a) no longer changes after time step 31, and friend Zealot 3's corresponding attention weights in other friend agents' heatmaps turn to be very dark after time step 31 compared to other living friend agents. What's more, those living agents' corresponding attention weights in others' heatmap turn to be brighter compared to the time step before 31, indicating that the living agents turn their attention to other friend agents after friend Zealot 3 is dead.

In order to illustrate the effectiveness of Attention-Aware Actor more intuitively, we attach an auxiliary snapshot of an appealing segment in Fig. 7, and we also extract the CoCo-Graph of an interesting agent (Friend 3) to explain the complex mutual interplay between agents. As revealed by the snapshot in Fig. 7, friend Zealot 3 helps friend Zealot 4 attack enemy Zealot 2. The CoCo-Graph of coagent 3 provides a clear explanation that these fragile Zealots (Friend 3, 4) have learned to stick together to form a formation so that they can focus fire on the enemy Zealot (Enemy 2) and achieve the goal of weakening the enemy



(a) Snapshot



(b) CoCo-Graph of agent 3

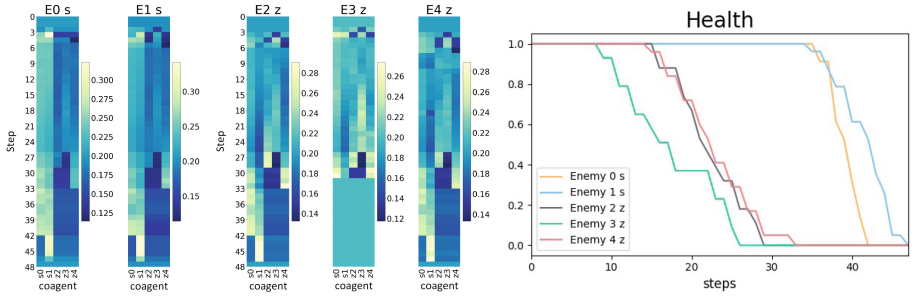
Fig. 7. Snapshot and the surrounding CoCo-Graph of agent 3

combat power as soon as possible. Once the current target enemy is destroyed, friend Zealot 3 will focus on the target enemy (Enemy 4) that another friend Zealot (Friend 2) is attacking, and cooperate with that Zealot to attack the enemy, which is an effective way to win the battle.

Attention Analysis of Attention-Aware Actor Focusing on Enemies. Figure 8 has shown the attention weights (γ_i) of each coagent i focusing on enemy agents at each time step in a complete trajectory of 2s3z battle.

One main tendency is that the corresponding part in heatmaps becomes completely dark after the corresponding enemy is dead, which means our coagents will pay no more attention to this dead enemy, and once an enemy is dead, the corresponding part of our coagents focusing on other living enemies' becomes brighter, indicating that our coagents turn their attention to other living enemies.

As we discussed in Sect. 3, the cooperation relationship and confrontation relationship are changing continuously, which means the attention weights within a complete trajectory of battle are changing violently. Under these circumstances, Tri-A could capture the mutual interplay between agents properly, and thus promote better understanding and decision-making as we analyzed above.



(a) Focus on enemies (b) Enemies' health

Fig. 8. (a) The attention heatmap of coagents focusing on each enemy. The head title of each heatmap indicates the unique mark of a specific enemy agent B that our agents is focusing on, including agent camp (E means Enemy), agent id (from 0 to 4) and agent type (s means Stalker, z means Zealot). Bottom horizontal ordination indicates the unique mark of five agents from our team, including agent id and agent type. Vertical ordination represents time steps that increase from top to bottom. (b) The health property of each enemy, which indicates the remaining blood.

5 Conclusion

In this paper, we present Attention-Aware Actor (Tri-A), a plug designed for deep multi-agent reinforcement learning method, which learns a much more observant actor for each agent in a decentralized way by applying attention to the local observation of each coagent. Without taking all agents into consideration or centralized processing, our method is much more effective in the case of limited communication. Extensive experiments carried on a series of battle games in StarCraftII demonstrate that our method improves the performance on a number of baseline algorithms which adopts the architecture of value decomposition. Additionally, we perform the attention analysis to visualize the attention weights of each coagent focusing on neighboring friends and enemies, and the results provide intuitive explanations that Tri-A is indeed helpful for understanding the mutual interplay between agents and thus promote better cooperation.

References

1. Ba, J., Mnih, V., Kavukcuoglu, K.: Multiple object recognition with visual attention. arXiv preprint arXiv:1412.7755 (2014)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473 (2014)
3. Borji, A., Cheng, M.M., Jiang, H., Li, J.: Salient object detection: a benchmark. IEEE Trans. Image Process. **24**(12), 5706–5722 (2015)
4. Iqbal, S., Sha, F.: Actor-attention-critic for multi-agent reinforcement learning. arXiv preprint arXiv:1810.02912 (2018)

5. Iqbal, S., Sha, F.: Actor-attention-critic for multi-agent reinforcement learning. In: International Conference on Machine Learning, pp. 2961–2970. PMLR (2019)
6. Jaques, N., et al.: Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In: International Conference on Machine Learning, pp. 3040–3049. PMLR (2019)
7. Jiang, J., Dun, C., Huang, T., Lu, Z.: Graph convolutional reinforcement learning. arXiv preprint arXiv:1810.09202 (2018)
8. Jiang, J., Lu, Z.: Learning attentional communication for multi-agent cooperation. arXiv preprint arXiv:1805.07733 (2018)
9. Judd, T., Ehinger, K., Durand, F., Torralba, A.: Learning to predict where humans look. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 2106–2113. IEEE (2009)
10. Lin, Z., et al.: A structured self-attentive sentence embedding. arXiv preprint arXiv:1703.03130 (2017)
11. Liu, Y., Wang, W., Hu, Y., Hao, J., Chen, X., Gao, Y.: Multi-agent game abstraction via graph attention neural network. arXiv preprint arXiv:1911.10715 (2019)
12. Liu, Y., Wang, W., Hu, Y., Hao, J., Chen, X., Gao, Y.: Multi-agent game abstraction via graph attention neural network. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 7211–7218 (2020)
13. Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. arXiv preprint arXiv:1706.02275 (2017)
14. Mnih, V., Heess, N., Graves, A., et al.: Recurrent models of visual attention. In: Advances in Neural Information Processing Systems, pp. 2204–2212 (2014)
15. Oliehoek, F.A., Amato, C.: A Concise Introduction to Decentralized POMDPs. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-28929-8
16. Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., Whiteson, S.: QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. In: International Conference on Machine Learning, pp. 4295–4304. PMLR (2018)
17. Samvelyan, M., et al.: The starcraft multi-agent challenge. arXiv preprint arXiv:1902.04043 (2019)
18. Sukhbaatar, S., Szlam, A., Fergus, R.: Learning multiagent communication with backpropagation. arXiv preprint arXiv:1605.07736 (2016)
19. Sunehag, P., et al.: Value-decomposition networks for cooperative multi-agent learning. arXiv preprint arXiv:1706.05296 (2017)
20. Vaswani, A., et al.: Attention is all you need. arXiv preprint arXiv:1706.03762 (2017)
21. Wyart, V., Tallon-Baudry, C.: How ongoing fluctuations in human visual cortex predict perceptual awareness: baseline shift versus decision bias. J. Neurosci. **29**(27), 8715–8725 (2009)
22. Yang, Y., et al.: Qatten: a general framework for cooperative multiagent reinforcement learning. arXiv preprint arXiv:2002.03939 (2020)
23. Yang, Y., Luo, R., Li, M., Zhou, M., Zhang, W., Wang, J.: Mean field multi-agent reinforcement learning. In: International Conference on Machine Learning, pp. 5571–5580. PMLR (2018)