Visual Explanation for Cooperative Behavior in Multi-Agent Reinforcement Learning

Hidenori Itaya, Tom Sagawa, Tsubasa Hirakawa, Takayoshi Yamashita, Hironobu Fujiyoshi,
Chubu University
1200 Matsumotocho, Kasugai, Aichi, Japan
{itaya, sagawa, hirakawa}@mprg.cs.chubu.ac.jp, {takayoshi, fujiyoshi}@isc.chubu.ac.jp

***Abstract*—Multi-agent reinforcement learning (MARL) can acquire cooperative behavior among agents by training multiple agents in the same environment. Therefore, it is expected to be applied to complex tasks in real environments, such as traffic signal control in a traffic environment and cooperative behavior of robots. In this study, using the multi-actor-attention-critic (MAAC) with the actor-critic method as a basis, we introduce an attention head for the actor that calculates the agent's action. In contrast to the critic in MAAC, which shares the attention head among all the agents, the attention head of the actor in our method is constructed independently for each agent. This allows the attention head of the actor to calculate actor-attention (indicating which other agents are gazed at by each agent) and to acquire cooperative behavior. We visualize actor-attention to analyze the basis of agents' decisions for cooperative behavior. Using single_spread, which is a multi-agent environment for cooperative problems, we show that the basis of decisions for cooperative behavior can be easily analyzed. We also demonstrate that our method efficiently obtains cooperative behavior considering other agents through quantitative evaluation of the cooperative behavior.**

***Index Terms*—multi-agent reinforcement learning, attention, explainability**

I. Introduction

Reinforcement learning (RL) can acquire the optimal behavior of an agent in an unknown environment by learning from the rewards it receives through interactions with the environment. The deep Q-network (DQN), a method that combines Q-learning [1] and a deep neural network (DNN), was proposed in 2015 and achieved a higher score than human players on the Atari2600 [2]. Since the appearance of this method, deep reinforcement learning (DRL), which incorporates deep learning into RL, has become mainstream and can handle problems with a large number of states, such as images. Because of this property, RL has been applied to various tasks such as robot control [3]–[6], automated driving [7], and video games [8]–[10].

Various systems in the real world can be considered as multi-agent systems (MAS). An MAS is essentially a system in which there are multiple agents that recognize situations, make decisions, and act on their own. Examples include crowd behavior in a traffic environment or a team game such as soccer. In a traffic environment such as a road, car drivers and pedestrians are aware of the situation and act according to their own judgment. Also, in soccer, players cooperate with each other for the common goal of scoring points within a team, and act against each other between their own team and an opposing team. In these environments, RL for single-agent learning cannot take into account the relationships between agents, resulting in a significant performance degradation. Multi-agent reinforcement learning (MARL) is an RL method that addresses this problem by training multiple agents simultaneously in the same environment. The problems addressed by MARL can be divided into two categories: "competitive problems", in which multiple agents compete, and "cooperative problems", in which multiple agents cooperate. Competitive problems are those in which each agent seeks to achieve its own goal in an adversarial relationship, such as between its own team and an opposing team in soccer, while a cooperative problem is one in which multiple agents cooperate with each other to achieve a common goal, such as overcoming traffic congestion in a traffic environment. To solve the cooperative problem, it is necessary for multiple agents to have a common goal, and each agent must acquire cooperative behavior by considering other agents in order to achieve the goal. For this reason, many studies have been conducted on MARL for the purpose of acquiring agents' cooperative policies [11]–[16]. However, while MARL achieves cooperative policy acquisition, the reasons for agents' decisions about cooperative actions are unclear. This problem is a major obstacle when it comes to the implementation of MAS using MARL in the real world. The same problem of explaining the reasons for agents' decisions exists in RL as well. Prior research on RL has focused on explainable deep reinforcement learning (XRL), which is devoted to analyzing the reasons for agents' decisions [17]–[27], but these studies have only examined RL for single-agent learning, and MARL has not been addressed.

The purpose of our study is to analyze the reasons for decisions on cooperative behavior considering other agents in the multi-agent environment. For this purpose, we introduce an attention head for the actor that calculates the actions of each agent on the basis of the multi-actor-attention-critic (MAAC) [13] utilizing the actor-critic method. In the attention head of the actor, the state of the target agent is Query and the states of the other agents are Key and Value, and the contribution values of the other agents to the behavior of the target agent are calculated. The critic in MAAC uses a single attention head that is shared among the agents, but our method constructs the attention head of the actor independently for each agent. This makes it possible to calculate which other agents to

gaze at for each target agent. In this study, the attention calculated by the attention head is called actor-attention. By visualizing the actor-attention, we can analyze the reason for the judgments about the cooperative behavior calculated by the actor. We performed an evaluation experiment using simple_spread of the multi particle environment (MPE) [28], which is a multi-agent environment. By comparing the rewards in simple_spread with the introduction of the attention head and with the weight sharing the attention head weight, we show that our method can analyze the reason for judgments on cooperative behavior without inducing a decrease in reward. We also show through quantitative evaluation that this method is useful for acquiring cooperative behavior. We demonstrate that the method can easily analyze the reason of judgments for cooperative behavior by using two cooperative scenes in simple_spread as examples.

Contributions The main contributions of this paper are as follows.

- We propose the introduction of an attention head for actors in the MARL model based on the actor-critic method. Actor-attention, which indicates whom the target agent considered when outputting its actions, can be easily obtained by forward propagation of the model.
- By visualizing actor-attention in the coordination problem, we can analyze the decision-making of agents that have acquired cooperative behavior. We conducted experiments using MPE's simple_spread to analyze agents' decision making for cooperative problems.

Section II of this paper reviews related works and Sec. III introduces the proposed method. In Sec. IV, we present experimental results on MPE and discuss the performance of the proposed method. We conclude in Sec. V with a brief summary and mention of future work.

II. Related works

A. Multi-agent reinforcement learning

The main learning method for MARL is centralized training decentralized execution (CTDE). CTDE is a method in which the learning is performed based on global information (information of all agents) when calculating the gradients of policy models constructed for each agent, but the agents use their own policy models for distributed control during testing and operation. The main CTDE-based MARL methods are described below.

Foerster *et al.* proposed counterfactual multi-agent policy gradients (COMA) [11], in which action values are estimated using a centralized critic shared among agents and action selection is performed by distributed actors, each of which has its own actor. This is one of the earliest methods in which CTDE was introduced, and it achieved a strong performance in a video game called StarCraft. Lowe *et al.* proposed multi-agent deep deterministic policy gradient (MADDPG) [12], which is based on deep deterministic policy gradient (DDPG) [29], a typical DRL method, and extended it to multi-agent by utilizing a centralized critic. The method achieved a strong performance on 2D video games. Sunehag *et al.* designed a centralized action value function that computes the action value of each agent and proposed a value-decomposition network (VDN) [30] that learns distributed actors by summing the action values of each agent. Rashid *et al.* proposed QMix [14], a Q-learning-based method that enables learning to consider multiple agents by utilizing a mixing network that scales each agent's action value function (the centralized action value function) across all agents. Iqbal *et al.* proposed MAAC [13], a method that introduces an attention head to the critic of the actor-critic method. The critic in this method uses the attention head to calculate the other agent's contribution from the target agent's point of view and then calculates the action value based on the contribution. It also shares the attention head introduced in the critique among the agents, which enables learning that efficiently considers the entire agent.

B. Explainable deep reinforcement learning

Several studies have analyzed the reason for agents' decisions in DRL. Sorokin *et al.* proposed the deep attention recurrent Q-network (DARQN) [17], which implements an attention mechanism in DQN, a typical DRL method. It introduces an attention mechanism in the LSTM front layer of the network that outputs the action value function $Q(s, a)$, thus enabling attention to be obtained for the action value function $Q(s, a)$, which is an index for action selection. Manchin *et al.* introduced self-attention to proximal policy optimization (PPO) [31] and aimed to improve the score as well as to analyze the policy. This method analyzes the agent's decision-making for the chosen action by visualizing an attention map for the policy. Weitkamp *et al.* proposed a bottom-up method based on gradient-weighted CAM (Grad-CAM) [32] for the actor to calculate a policy of asynchronous advantage actor-critic (A3C) [33] [22]. Grad-CAM is a method that obtains an attention map by utilizing the response value of the convolutional layer during forward propagation and the gradient during backpropagation. It avoids degradation of the recognition performance by generating the attention map from the gradient information. Since Weitkamp *et al.*' s method is based on Grad-CAM, backpropagation is required to calculate the attention map. Greydanus *et al.* obtained a saliency map in A3C by calculating a perturbed image with a Gaussian filter applied to the gradient during backpropagation [24]. This method requires backpropagation to obtain a saliency map because it is a bottom-up method (similar to Weitkamp *et al.*' s). Shi *et al.* proposed a self-supervised interpretable network (SSINet) [20] that generates a fine-grained attention mask to highlight task-related information relevant to agent decision-making. They generate an attention mask for agents modeled by an actor network by integrating SSINet in front of the actor network. Rupprecht *et al.* propose a tool to visualize DRL agent's behavior [21]. They focus on the importance of understanding the agent's state recognition of very high or low reward states, and analyze this by learning a generative model of the environment and artificially generating new states to induce the target agent's behavior. Mott *et al.* obtained two

attentions ("what" and "where") by using query-based attention in their actor-critic-based DRL method [27]. This method requires significant changes in the network architecture (Key, Value, etc.) because of the need to generate attention queries. The method obtains different attentions associated with "what" / "where" by generating attention queries.

One XRL method for MARL is presented in the study by Boggess *et al.* [34], who proposed a method for generating linguistic explanations to answer queries about an agent's behavior. The method consists of three types of queries and clarifies the agent's behavior under certain conditions. In contrast to Boggess *et al.*'s method, our study clarifies the agent's behavior by using actor attention, which indicates which other agents were considered when the agent chose its action, rather than language.

Yang *et al.* proposed Qattn, which introduces multi-head attention to the mixing network in QMix [15]. The Q-value Q_{tot} shared among agents is calculated by calculating the influence of agent i on the entire environment using multi-head attention and weighting the Q-value Q_i of each agent i. It is possible to intuitively explain the Q-values acquired by learning from the Attention weights in StarCraft. In contrast to the method of Yang *et al.*, our study provides a more intuitive explanation of the behavior by introducing multi-head attention to the Actor, which outputs the actions of each agent. We can also obtain more interpretable Attention weights that indicate which agents considered whom and to what extent in the current state of the environment.

In DRL, policies are important clues for solving a given task and environment. These explanatory methods provide some approach to these policies to analyze decision making in DRL agents. Thus, there are many explanatory methods for DRL for single agents. However, there are relatively few studies on the decision-making for agents in MARL with multiple agents. Since strategies are crucial for understanding agent decision-making in DRL, we believe that strategies are also important in MARL. In this study, we focus on the cooperative behavior acquired by agents through MARL and analyze the decision-making of multiple agents.

III. Proposed method

The main challenge in MARL is the coordination problem, and many methods have thus been proposed to acquire cooperative behavior among agents. However, the reason for the agents' decisions on the acquired cooperative behavior is unclear in these methods, which represents an obstacle when it comes to the social implementation of MARL. To address this, we propose introducing the actor's attention head, which calculates the agent's action, on the basis of the multi-actor attention-critic (MAAC) [13].

A. Structure of actor

In this section, we describe the structure of the actor in the proposed method. An overview of the proposed method is shown in Fig 1. MAAC, on which the proposed method is based, utilizes an actor-critic structure consisting of an actor that calculates the policy of an agent and a critic that calculates the evaluation values for the agent.

As shown in the Fig. 1, the actor in the proposed method consists of an MLP, an attention head, an MLP actor, and a policy net. The MLP is a network that extracts features from the state s_i, s_j of each agent i, j. Here, j is a different agent than i. The attention head is a network that calculates the contribution value x_i of the other agent j to the target agent i using the calculated features e_i, e_j. The MLP actor is a network that extracts features from the state s_i of the target agent i. The policy net is a network that computes the target agent i's action a_i based on the computed features and the target agent i's contribution x_i. These networks do not share the weights of the network among the agents, and the agents have their own parameters. These structures are utilized to obtain actor attention, which indicates which other agents are being gazed at by the agent that has acquired cooperative behavior. The critic in the proposed method has the same structure as MAAC, and unlike the attention head of the actor, it takes as input the state s_i, s_j of each agent i, j and the action a_i, a_j of each agent i, j calculated by the actor. Also, $\mathrm{MLP}_i, \mathrm{MLP}_j$, and Attention head, which extract features from the agent's state s_i, s_j in the critic, share network weights among agents.

B. Attention head in actor

This section describes the processing in the actor's attention head in the proposed method.

Fig. 2 shows the structure of the attention head introduced in the actor. The input to the attention head is the state feature e_i of the target agent i as Query and the state feature e_j of the other agent j as Key / Value. The attention head is composed of Scaled dot product, Softmax, and Dot product. Scaled dot product is a module that calculates the inner product between Query and Key and scales it by the number of dimensions. Softmax is a function that converts the value calculated by the Scaled dot product to a value between 0 and 1, and then converts it to attention. In our study, this attention is called actor-attention att_{actor}. Dot product is a module that weights values using actor-attention att_{actor}.

The actor-attention att_{actor} at the attention head in the actor is calculated by

$$att_{actor} = \mathrm{softmax}\left(e_i \cdot e_j^T / \sqrt{dim}\right). \tag{1}$$

where $\mathrm{softmax}(\cdot)$ is the softmax function, e_i, e_j are the features of agents i, j, and dim is the number of dimensions of feature e.

IV. Experiments

In this section, we present our evaluation of the proposed method and analysis of the reasons for multi-agent decisions using actor-attention for cooperative problems. The following three evaluations were performed.

- A comparison of the performance of the proposed method with the introduction of the attention head and network weight sharing in the actor by reward.

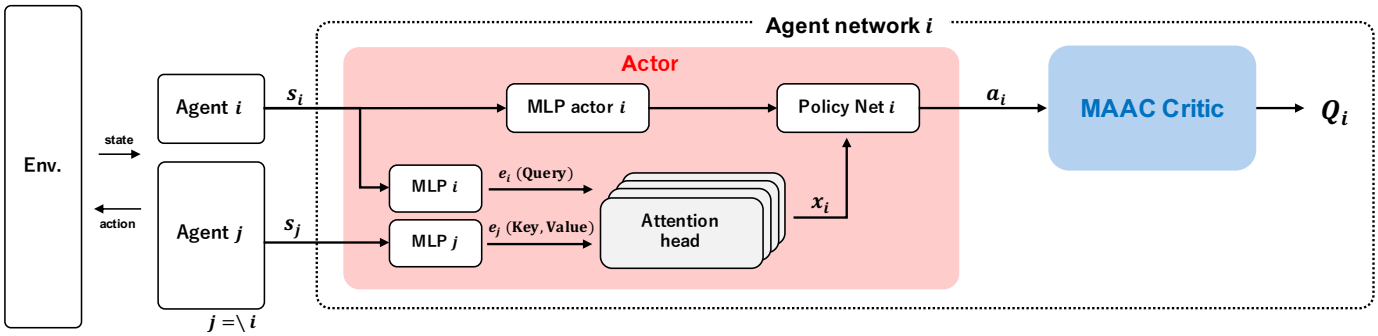


Fig. 1: Overview of the proposed method.

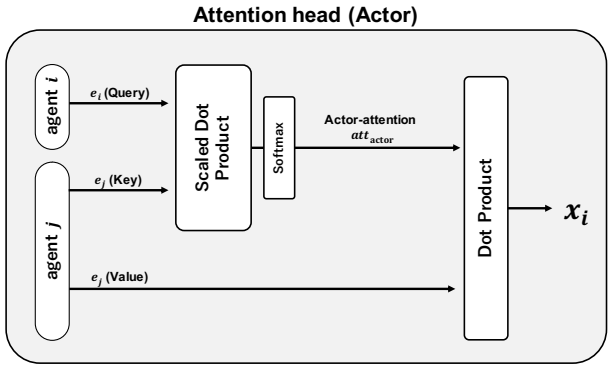


Fig. 2: Structure of attention head for actor.

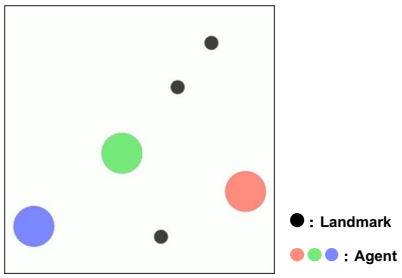


Fig. 3: Example of simple_spread task.

- An evaluation of the quantitative measures of cooperative behavior.
- An analysis of the reasons for decisions about cooperative behavior using actor-attention.

A. *Multi-agent environment for cooperative problems*

For these evaluations, we utilize simple_spread of the multi particle environment (MPE) [35], which is a cooperative problem in a multi-agent environment. As shown in Fig. 3, simple_spread is a task in which multiple agents move toward multiple landmarks that are the goals. In this task, each agent and each landmark is randomly generated, and the goal is to reach the landmarks while avoiding collisions between agents. In this experiment, the number of landmarks was set to 3 (black) and the number of agents to 3 (red / green / blue). Each agent has five possible actions: “Stop”, “Up”, “Down”, “Right”, and “Left”. The reward design is a negative number of distances to the nearest landmark from the agent and a negative reward (-1) in the case of a collision. The episode is assumed to end after 25 steps.

B. *Comparison of performance using reward*

We compare the reward values in simple_spread using a model trained on 100,000 episodes, from two points of view: the introduction of the attention head and the network weight sharing in the actor of the proposed method.

Table I shows a comparison by average reward value between 100 episodes with the introduction of the attention head in the actor and critic. Here, no-head is a model in which neither the actor nor the critic has an attention head, a-head and c-head are models in which one of them has an attention head, and ours is a model in which both of them have an attention head (which is the proposed method).

As we can see, the difference in average reward between a-head and no-head is 0.1, indicating that there is no difference in reward value. The difference of the average reward between c-head and ours is 0.07, which also confirms that there is no difference in the reward value. On the other hand, c-head and ours, which introduces the attention head to the critic, shows higher average rewards than no-head. These results suggest that the introduction of an attention head to the critic contributes significantly to the improvement of accuracy. Although the introduction of the attention head to the actor does not contribute significantly to the improvement of accuracy, it is considered that actor-attention can be acquired without inducing a decrease in accuracy when used in combination with the introduction of the attention head to the critic.

Next, we compared the performance of the proposed method when sharing the network weights in the attention head and MLP. Table II shows a comparison of the average reward values among 100 episodes when sharing the network weights

TABLE I: Comparison of mean reward by introduction of attention head. We evaluated the models with the highest reward values out of the 100,000 episodes trained.

Method	Attention head		Mean reward
	actor	critic	
no-head			–1.96
a-head	✓		–2.06
c-head (MAAC)		✓	–1.59
ac-head (ours)	✓	✓	–1.66

TABLE II: Comparison of the mean reward over 100 episodes with sharing weights of the network in the proposed method's actor. We evaluated the models with the highest reward values out of the 100,000 trained episodes.

Method	Actor weight sharing		Mean reward
	MLP	Att. head	
no-share			–1.66
mlp-share	✓		–2.61
head-share		✓	–2.58
mlp-head-share	✓	✓	–2.61

in the proposed method's actor. Here, no-share is a model without weight sharing in the actor (proposed method), mlp-share and head-share are models with MLP or attention head weight sharing in the actor, and mlp-head-share is a model with both MLP and attention head weight sharing in the actor. As we can see, the mean reward is significantly lower for mlp-share, head-share, and mlp-head-share with shared network weights compared to no-share. Actor is a mechanism to calculate the actions of each agent, and the target to be paid attention to differs from agent to agent. Therefore, our method, which does not share network weights between MLP and attention head, is considered to be the most accurate.

C. Quantitative evaluation of cooperative behavior

In this study, we define cooperative behavior as behavior in which an agent does not move to the nearest landmark and cedes a landmark to another agent. As a quantitative evaluation of the cooperative behavior, the number of cooperative behaviors (the number of times the target agent moved from its initial position to a location other than the nearest landmark) during 100 episodes is evaluated in simple_spread. In this evaluation, landmarks are set at a certain distance apart, and rule-based control is utilized in which agents other than the target agent move to the nearest landmark. Similarly, the number of episodes in which conflicts between agents occur is also compared.

Table III shows the number of cooperative actions among 100 episodes, and Table IV shows the number of episodes in which collisions of agents occurred among 100 episodes. Here, target agent is the agent to be evaluated and our is the proposed method. As we can see in Table III, ours increases the number of cooperative actions for each target agent compared to MAAC, with a total improvement of 56 times for the three agents. Moreover, as shown in Table IV,

TABLE III: Number of cooperative actions in 100 episodes

Method	Target agent			Sum
	red	green	blue	
maac	50	63	69	182
ours	72	85	81	238

TABLE IV: Number of episodes in which collision of agent occurred in 100 episodes

Method	Target agent			Sum
	red	green	blue	
maac	27	24	22	73
ours	19	23	19	61

the proposed method reduces the number of collisions by 12 times compared to MAAC. In other words, compared to MAAC, the proposed method is better able to acquire collision avoidance behavior. These results demonstrate that introducing the attention head to the actor can promote the acquisition of cooperative behavior as well as collision avoidance behavior.

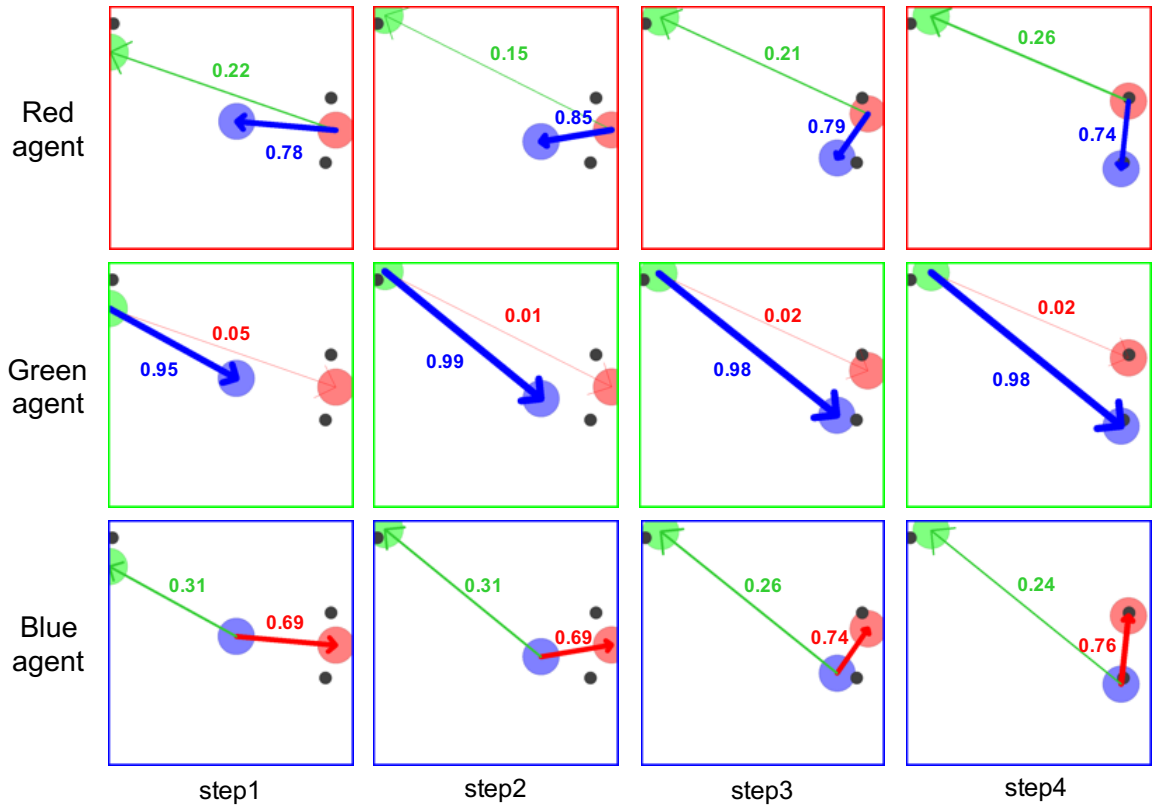
D. Analysis of reasons for judging cooperative behavior using actor attention

Finally, we analyze the reasons for agents' cooperative behavior by visualizing actor-attention in the simple_spread environment. Examples of actor-attention visualization are provided in Fig. 4, where (a) (Scene 1) shows a scene in which the Red agent cooperates with the Blue agent and (b) (Scene 2) is a scene in which the Green agent performs a cooperative action against the Red agent.

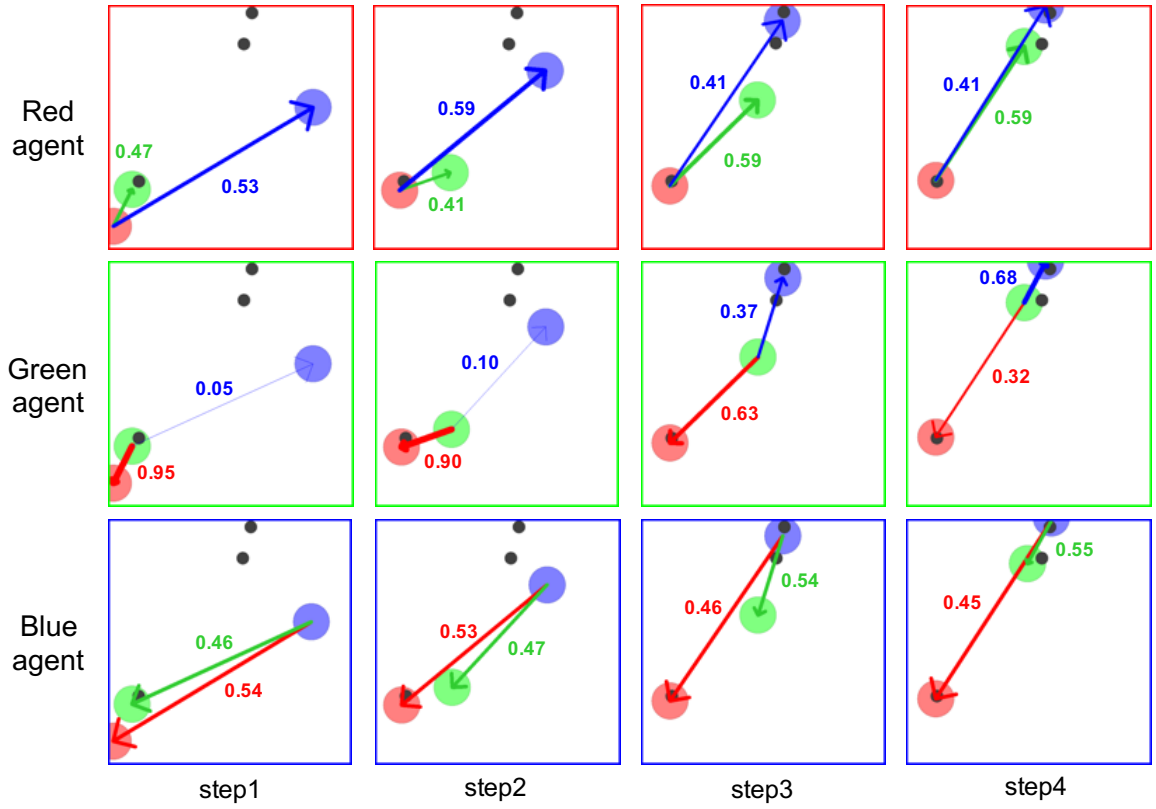
1) Scene 1: Red is located in the middle of the two landmarks in Fig. 4(a). Red gazes more strongly at Blue than at Green throughout steps 1–4. In step 2, Red gazes more strongly at the approaching Blue (0.85) and moves to the upper landmark in steps 3–4. Similarly, Blue gazes more strongly at Red than at Green through steps 1–4, and moves to the lower landmark. On the other hand, Green gazes more strongly at Blue than at Red throughout steps 1–4 and moves to the nearest landmark. These results suggest that Red and Blue are cooperating to reach the landmark without collision by gazing at each other because the landmarks they are aiming for are close. By gazing at the nearest agent (Blue), Green recognizes that Blue is moving away from the landmark that Green is aiming at and moves to the nearest landmark.

2) Scene 2: From step 1 in Fig. 4 (b), Red and Green have the same agent with the nearest landmark. Red gazes at both Green and Blue to the same extent and moves to the nearest landmark. On the other hand, Green gazes strongly at Red in steps 1 and 2 and moves to a distant landmark. Blue gazes at both Red and Green to the same extent and moves to the upper landmark throughout steps 1–4. From these results, we can assume that Green recognizes that Red is moving toward the same landmark and cedes the landmark to Red.

In our study, we analyzed the reason for the cooperative behavior of multi-agents by visualizing actor-attention and showed the analysis in simple_spread.



(a) **Scene 1**



(b) **Scene 2**

Fig. 4: **Visualization of actor-attention in simple_spread**: The border indicates the color of the target agent. The thickness of the arrows between agents indicates the degree to which the target agent is gazing at the other agents. The thicker the line is, the more the target agent is gazing at the agent. The values plotted on the arrows are actor-attention values.

V. Conclusion

In this study, we proposed a method of introducing an attention head to an actor that calculates the agent's behavior based on the multi-actor-attention-critic (MAAC) using the actor-critic method. We obtained actor-attention, which indicates to which of the other agents the agent pays attention, by introducing an attention head to the actor. By visualizing actor-attention, we have realized the analysis of the reason for agents' decisions for cooperative behavior in a multi-agent environment. We performed experiments using MPE's simple_spread to analyze the reasons for MARL agents' decisions on cooperative behavior. In addition, our quantitative evaluation of the cooperative behavior showed that the proposed method is effective in acquiring cooperative behavior. This method requires the state of all agents to be known for the selection of an agent's action due to the calculation of actor-attention. We are therefore planning to develop a MARL method that does not require this information, and to conduct experiments in other environments.

References

[1] C. J. Watkins and P. Dayan, "Q-Learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[2] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[3] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[4] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations," *Proceedings of Robotics: Science and Systems*, 2018.

[5] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *Proceedings of international conference on robotics and automation*, 2017, pp. 3389–3396.

[6] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection," *The International Journal of Robotics Research*, vol. 37, no. 4–5, pp. 421–436, 2018.

[7] M. Toromanoff, E. Wirbel, and F. Moutarde, "End-to-end model-free reinforcement learning for urban driving using implicit affordances," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 7153–7162.

[8] C. Tessler, S. Givony, T. Zahavy, D. J. Mankowitz, and S. Mannor, "A deep hierarchical approach to lifelong learning in Minecraft," *AAAI Conference on Artificial Intelligence*, 2017.

[9] N. Justesen, P. Bontrager, J. Togelius, and S. Risi, "Deep Learning for Video Game Playing," *arXiv preprint, arXiv:1708.07902*, 2017.

[10] K. Shao, Z. Tang, Y. Zhu, N. Li, and D. Zhao, "A survey of deep reinforcement learning in video games," *arXiv preprint arXiv:1912.10944*, 2019.

[11] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[12] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Neural Information Processing Systems*, 2017.

[13] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *International conference on machine learning*. PMLR, 2019, pp. 2961–2970.

[14] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[15] Y. Yang, J. Hao, B. Liao, K. Shao, G. Chen, W. Liu, and H. Tang, "Qatten: A general framework for cooperative multiagent reinforcement learning," *arXiv preprint arXiv:2002.03939*, 2020.

[16] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *International conference on machine learning*. PMLR, 2019, pp. 5887–5896.

[17] I. Sorokin, A. Seleznev, M. Pavlov, A. Fedorov, and A. Ignateva, "Deep attention recurrent q-network," *Proceedings of neural information processing systems workshops*, 2015.

[18] P. Madumal, T. Miller, L. Sonenberg, and F. Vetere, "Explainable reinforcement learning through a causal lens," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 03, 2020, pp. 2493–2500.

[19] Y. Tang, D. Nguyen, and D. Ha, "Neuroevolution of self-interpretable agents," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, 2020, pp. 414–424.

[20] W. Shi, G. Huang, S. Song, Z. Wang, T. Lin, and C. Wu, "Self-supervised discovering of interpretable features for reinforcement learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[21] C. Rupprecht, C. Ibrahim, and C. J.Pal, "Finding and visualizing weaknesses of deep reinforcement learning agents," in *International conference on learning representations*, 2020.

[22] L. Weitkamp, E. van der Pol, and Z. Akata, "Visual rationalizations in deep reinforcement learning for atari games," in *Benelux conference on artificial intelligence*, 2018, pp. 151–165.

[23] T. Zahavy, N. Ben-Zrihem, and S. Mannor, "Graying the black box: Understanding dqns," in *International conference on machine learning*, 2016, pp. 1899–1908.

[24] S. Greydanus, A. Koul, J. Dodge, and A. Fern, "Visualizing and understanding Atari agents," *Proceedings of international conference on machine learning*, vol. 80, pp. 1792–1801, 2018.

[25] R. Zhang, Z. Liu, L. Zhang, J. A. Whritner, K. S. Muller, M. M. Hayhoe, and D. H. Ballard, "AGIL: Learning Attention from Human for Visuomotor Tasks," in *Proceedings of the european conference on computer vision*, 2018, pp. 663–679.

[26] A. Manchin, E. Abbasnejad, and A. van den Hengel, "Reinforcement learning with attention that works: A self-supervised approach," in *In International Conference on Neural Information Processing. Springer, Cham.*, 2019, pp. 223–230.

[27] A. Mott, D. Zoran, M. Chrzanowski, D. Wierstra, and D. Jimenez Rezende, "Towards interpretable reinforcement learning using attention augmented agents," *Advances in Neural Information Processing Systems*, vol. 32, pp. 12 350–12 359, 2019.

[28] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," *arXiv preprint arXiv:1703.04908*, 2017.

[29] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *International conference on learning representations*, 2016.

[30] P. Sunehag, G. Lever, A. Gruslys, W. M. Czarnecki, V. F. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, and T. Graepel, "Value-decomposition networks for cooperative multi-agent learning," in *Adaptive Agents and Multi-Agent Systems*, 2017.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," in *arXiv preprint arXiv:1707.06347*, 2017.

[32] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization," *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

[33] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous Methods for Deep Reinforcement Learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[34] K. Boggess, S. Kraus, and L. Feng, "Toward policy explanations for multi-agent reinforcement learning," in *International Joint Conference on Artificial Intelligence*, 2022.

[35] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," *arXiv preprint arXiv:1703.04908*, 2017.