An Improved Deep Reinforcement Learning-Based Multi-Agent Cooperative Game Approach

Zhongqi Zhao, Chuang Zhang, Haoran Xu, Jiawei Kou and Hui Cheng

October 31, 2023

An Improved Deep Reinforcement Learning-Based Multi-Agent Cooperative Game Approach

1st Zhongqi Zhao
School of Computer and Engineering
Xi'an University of Technology
Xi 'an City, Shaanxi Province,China
3190111026@stu.xaut.edu.cn

2nd Chuang Zhang
School of Computer and Engineering
Xi'an University of Technology
Xi 'an City, Shaanxi Province,China
3200432049@stu.xaut.edu.cn

3rd Haoran Xu
School of Computer and Engineering
Xi'an University of Technology
Xi 'an City, Shaanxi Province,China
3200921050@stu.xaut.edu.cn

4th Jiawei Kou
School of Computer and Engineering
Xi'an University of Technology
Xi 'an City, Shaanxi Province,China
3200121024@stu.xaut.edu.cn

5th Hui Cheng
School of Computer and Engineering
Xi'an University of Technology
Xi 'an City, Shaanxi Province,China
3200131045@stu.xaut.edu.cn

***Abstract*—Multi-agent collaborative games based on deep reinforcement learning have been one of the hot topics in the field of artificial intelligence in recent years. Building on existing research, this paper selects the on-policy Multi-Agent Proximal Policy Optimization (MAPPO) algorithm to explore its performance in multi-agent collaborative games, providing new insights for further research.**

Using the Hanabi game environment, this paper implements the MAPPO algorithm with an appropriate action space to maximize collaborative efficiency and competitiveness. Experimental results demonstrate that the MAPPO algorithm performs well in collaborative gaming scenarios. Compared to the off-policy Value-Decomposition Networks (VDN) algorithm [1], it improves the decision efficiency and outcomes of intelligent agents. This study highlights the feasibility and advantages of the MAPPO algorithm in multi-agent collaborative games.

Furthermore, this experiment delves into the application of the MAPPO algorithm in multi-agent collaborative games, offering valuable insights for enhancing reinforcement learning algorithms and their practical applications. This study also poses new questions and provides guidance and inspiration for future researchers.

Keywords—reinforcement learning;multi-agent collaborative games; MAPPO

I. INTRODUCTION

With the advancement of perceptual intelligence technology, artificial intelligence is gradually shifting from perceptual intelligence to decision-making intelligence, as evidenced by the remarkable performance of AlphaGo and DQN in Atari games[2]. However, the real world is complex and ever-changing, and the decisions of individual agents are no longer sufficient to meet the demands. Group intelligent decision-making is becoming mainstream[3]. For example, in fields such as logistics warehousing and smart cities, multi-agent systems can collaborate to achieve task distribution, planning, and execution. In summary, multi-agent systems based on deep reinforcement learning have vast development prospects and significant application value.

Currently, several teams have made notable progress. The DeepMind team led by Silver et al. proposed the AlphaGo algorithm, which combines deep learning algorithms with Monte Carlo tree search[4]. The team at Tsinghua University implemented collaborative decision-making among multiple intelligent agents in an aerial combat simulation platform, achieving significant performance improvements[5]. Researchers from Beijing Institute of Technology proposed an approach to achieving multi-agent collaborative control through interactive learning, obtaining favorable experimental results[6]. A team at Shanghai Jiao Tong University introduced a multi-agent collaborative path planning method based on game theory and machine learning, which achieved excellent results in logistics delivery scenarios[7]. The research team at Tsinghua University presented a multi-agent collaborative learning method based on dynamic cooperation mechanisms, achieving outstanding performance in multiple tasks[8].

The main research contents of this paper include: comparing the efficiency of the MAPPO algorithm with the VDN algorithm in the field of multi-agent applications; designing the action space for collaborative tasks, designing and implementing a multi-agent collaborative game model based on the Hanabi environment; and verifying the advantages of MAPPO in terms of training time efficiency and scores in multi-agent collaborative tasks.

II. MODEL DESIGN

In this chapter, a multi-agent collaborative game model based on the MAPPO algorithm is designed, taking into account the gameplay process of the Hanabi game.

A. Multi-Agent Design

The goal of MAPPO is to enable multiple intelligent agents to learn robust strategies in both cooperative and competitive environments to maximize cumulative rewards. It employs the proximal policy optimization concept of PPO[9], which aims to ensure that the new policy is not too far from the old policy in each policy update, thereby improving algorithm stability. In the field of multi-agent collaborative games, the Hanabi game presents a highly challenging problem[10]. Hanabi agents utilize a centralized training, distributed execution approach, consisting primarily of three components: the policy network, the action space construction module based on the current environment, and the value-based action decision module. The architecture of the intelligent agents is illustrated in Fig.1.

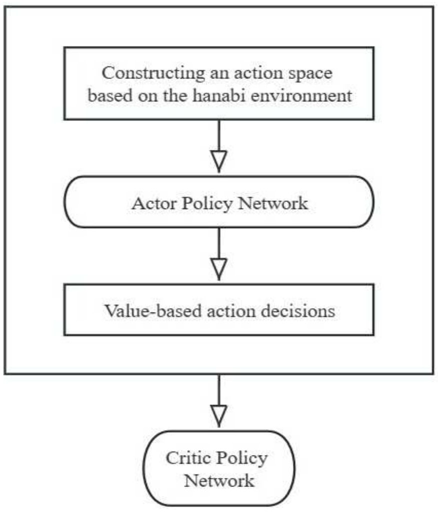
Constructing an action space based on the hanabi environment
Actor Policy Network
Value-based action decisions
Critic Policy Network

Figure 1. Overall Architecture of Multi-Agents

B. Construction of Multi-Agent Action Space

In the Hanabi game, the legal action space includes: first, ensuring that the combinations required to form fireworks in Hanabi exist either in all players' hands or in the un-drawn deck; second, ensuring that the type (color) of the played card matches the responding card and that the card value is greater than the responding card (number). This paper constructs the Hanabi action space state by identifying responding cards, calculating the available actions for the current player's card, and selecting the optimal action card value through three main sections. The processing flow of each subpart is as follows.

1) Identification of Responding Cards: The method for identifying responding cards involves matching and recognizing the card type and value based on the combinations of cards already played and inferred information about one's own hand.

2) Available Actions for the Current Player: · Based on the identified responding card information and game rule restrictions, the paper calculates the action combinations of cards in the current player's hand that match the environment and computes the log probabilities and card values of these combinations.

3) Selection of Optimal Action Card Value: From the available actions for the current player, suitable actions that would result in a total card value greater than the responding card value are selected as valid action states.

C. Actor Policy Network

The policy function of the actor network is represented as a probability density function $\pi(\alpha|s)$, which outputs the action with the highest log probability. This function is responsible for controlling the agent's actions, i.e., the choice of card-playing strategies. The $\pi(\alpha|s)$ function takes an observation state *s* as input and outputs the log probability distribution for the corresponding action *α*. The policy network approximates the policy function by training a neural network. For the parameters of θ, policy gradient algorithms can be employed to facilitate updates, thus completing the training of the policy network $\pi(\alpha|s;\theta)$.

In the training process, the state-value function $V^{\pi}(S_t)$ is the weighted average of the action-value function $Q^{\pi}(S_t,\alpha_t)$ and represents the expected value with respect to the action *α*. The value of this function is independent of the magnitude of action *α* and solely depends on the policy function and the current state S_t. In other words, when the policy function is determined, a larger state-value function indicates the potential for greater rewards in the current state. Therefore, the state-value function can be used to evaluate the quality of the implemented policy. By using the policy network to approximate the state-value function, the policy function can be formulated as shown in Formula 1.

$$V(S_t;\alpha_t)=\int\pi(\alpha|S_t;\theta_t)Q^{\pi}(S_t,\alpha)\mathrm{d}\alpha \tag{1}$$

During training, the actor policy network continuously optimizes the relevant parameters θ, leading to an increase in the state-value function. Consequently, the corresponding objective function for the policy network is defined as shown in Formula 2, allowing it to achieve better performance as training progresses.

$$J_{(\theta)}=\mathrm{E}_S[V(S;\theta)] \tag{2}$$

D. Critic Policy Network

Rewards play a crucial role in reinforcement learning, where the ultimate objective is to find the optimal policy that maximizes the cumulative reward. In the context of the Hanabi game, intelligent agents engage in cyclic interactions, culminating in a final state. Assuming a value function provides the optimal actions, each agent can derive the optimal policy based on this function. However, since the optimal action-value function is typically unknown, it is often estimated using temporal-difference algorithms. The goal of this algorithm is to make the value function predictions for observed actions closer to the target values. To achieve this, a least squares loss function is constructed, as shown in Formula 3. Weight updates, denoted by , are carried out using gradient descent to minimize the loss function.

$$L_t=\frac{1}{2}[Q(S_t,\alpha_t;w_t)-y_t]^2 \tag{3}$$

Hanabi's cooperative gameplay scenario involves receiving dense rewards. Throughout the cooperative game, rewards may be generated after each move or a series of moves. The final reward is provided at the conclusion of the entire game, but it can be considered that there are no rewards during the process. The focus is solely on the final score, effectively leading to a sparse reward scenario, where $r_t=0(t\neq T)$. Training samples in such scenarios have minimal impact on the overall network and cannot lead to significant policy improvements. Training becomes exceptionally slow, and there is a risk that the network may not converge. However, since the Hanabi game's cooperative gameplay is more concerned with long-term benefits, specifically the final score achieved by all players, the discount factor (denoted as γ) can be set to 1, making the cumulative return at time t become $G_t=r_T$.

In the algorithm presented in this paper, the design of the reward function takes several factors into consideration, including the number of cards of each color, their numerical values in each player's hand, and the count of tokens passed. During the gameplay, if a player's hand contains a higher number of cards of a specific color or a wider range of numerical values, it becomes easier to play cards from their hand and complete fireworks. Therefore, this paper chooses to use the average number of cards from the player's hand with more colors and a broader numerical range as one of the reward components.

III. EXPERIMENTS AND ANALYSIS

In this chapter, we measure and compare the performance and efficiency of the MAPPO algorithm and the VDN method in the Hanabi game environment using two metrics: training time and training score.

A. *Evaluation Metrics*

1) Training Time: The level of cooperative collaboration among agents generally improves gradually with training time (or iteration count) and tends to stabilize over time. Therefore, this experiment compares the two algorithms from two perspectives. First, it evaluates time under equal effects. That is, when the scores have stabilized, it compares the time consumed by each algorithm, with shorter times indicating higher efficiency. Second, it evaluates scores under equal time. In this case, the experiment ensures that different reinforcement learning algorithms achieve cooperative scores while training for the same amount of time, with higher scores indicating faster convergence.

2) Training Score: Training Score: The final score in cooperative games represents the overall performance of the algorithm. Each algorithm's intelligent agents compute their respective final scores. Higher scores indicate a higher level of cooperative collaboration.

B. *Experimental Setup*

The multi-agent environment in the experiment is based on the non-reinforcement learning Hanabi game environment provided by the DeepMind open-source toolkit. To avoid errors introduced by differences in initial card hands and turn order, all comparative experiments use the same initial card hands and clockwise turn order for repeated training.

Training Time Comparison Setup:Training time comparison measures the scores achieved by intelligent agents collaborating in a fixed number of cooperative games. During the training process of MAPPO and VDN, two sets of intelligent agents are periodically extracted at the same time intervals and engage in cooperative games with intelligent agents from the DeepMind open-source environment. Each stage involves 2000 rounds of games, and the average scores for MAPPO and VDN are recorded. The performance of MAPPO and VDN in terms of training time efficiency is compared based on the score curves during training.

Training Score Comparison Setup:To assess the performance of the MAPPO algorithm in multi-agent cooperative Hanabi games, multiple intelligent agents trained for the same time intervals engage in cooperative games. Specifically, MAPPO and VDN intelligent agents are evaluated after the same training duration, with each pair playing 2000 cooperative games, and the final scores are recorded.

C. *Experimental Results and Analysis*

1) Comparison of Training Time between MAPPO and VDN: During the training process, as shown in Fig.2, we present the score evolution curves for two types of agents based on MAPPO and VDN, with training progress measured in days.

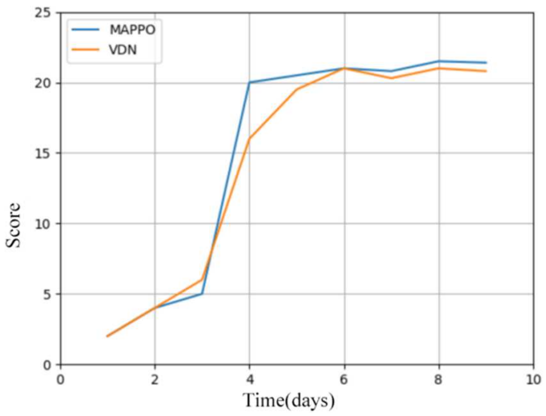

Figure 2. Game Score Changes for MAPPO and VDN Algorithm

As shown in Fig.2, the score curves for both types of intelligent agents exhibit two phases. In the early stage of training (0-4 days), the agents continuously explore various strategies, representing a trial-and-error process. With an increase in interaction frequency and the number of analyzed samples, the agents gradually learn card-playing strategies, leading to a rapid improvement in game scores. In the later stage of training (after 5 days), agents primarily rely on existing knowledge for decision-making and continue to explore better strategies, resulting in slower score improvements.

Simultaneously, based on the results in Fig.2, it is concluded that the training efficiency of the MAPPO algorithm is superior to VDN. Taking a score of 20 as the baseline for early training, the MAPPO algorithm reaches this baseline one day earlier than VDN. MAPPO also achieves convergence earlier, with the curve flattening at 4 days. Furthermore, in the later stages of training, MAPPO consistently outperforms VDN in terms of scores. These results collectively demonstrate the effectiveness of the deep reinforcement learning algorithm MAPPO in training Hanabi intelligent agent models.

2) Comparison of Training Score between MAPPO and VDN: Under the condition of basic parameters being the same, the average scores and highest scores for both algorithms are presented in Table 1.

TABLE I. SCORES OF MAPPO AND VDN IN HANABI GAMES

Number of Players	Score	MAPPO	VDN
2	Average	23.56	23.50
	Highest	23.94	23.78
3	Average	23.48	23.45
	Highest	23.87	23.80
4	Average	23.24	22.86
	Highest	23.55	23.53
5	Average	22.77	21.04
	Highest	23.01	21.62

Table 1 shows the scores of MAPPO and VDN algorithms in the Hanabi game environment with varying numbers of agents, ranging from 2 to 5. All the methods listed in the table have iterated over at least 10 billion environment steps. As shown in the table, MAPPO achieves the best results in terms of both highest and average scores in most cases. This indicates that MAPPO exhibits strong performance in the Hanabi environment.

IPPO's performance is comparable to MAPPO in the setting with 2 agents. However, as the number of agents increases, MAPPO shows significant performance improvement over IPPO and the other two off-policy methods, indicating that the architecture of centralized training and decentralized execution may be crucial.

In summary, in the experimental comparison between MAPPO and VDN, the MAPPO algorithm outperforms VDN in terms of both training time and post-training score performance in the Hanabi game. Therefore, the multi-agent collaborative game model based on the MAPPO algorithm improves game efficiency and effectiveness, carrying significant research value and significance.

IV. Conclusion

This paper presents a study on multi-agent collaborative gaming using the deep reinforcement learning algorithm MAPPO, validated through experiments conducted in the Hanabi game environment. The research involves designing an action space suitable for reinforcement learning models and, through comparison with the off-policy VDN algorithm, confirms the outstanding performance and stability of the MAPPO algorithm in different gaming scenarios. The experimental results demonstrate that the MAPPO algorithm adapts better to scenarios involving multi-agent collaborative gaming, significantly enhancing the agents' collaborative efficiency and competitive capabilities, with broad potential applications. Additionally, the study finds that action space design plays a crucial role in the performance of reinforcement learning models, particularly in addressing issues related to action continuity and high dimensionality.

Research in the domain of multi-agent collaborative gaming holds vast potential for various applications, including intelligent transportation and optimization of power systems. However, this field still faces numerous unresolved challenges, such as improving the robustness of multi-agent collaborative learning and addressing issues related to incomplete information and uncertainty. We will continue to delve deeper into the field of multi-agent collaborative gaming, optimize and enhance algorithms, expand the scope of experimental research, and integrate with other technologies, such as pre-training techniques. In our future research endeavors, we aspire to achieve richer and more in-depth research outcomes, providing more effective solutions and optimization methods for the domains of intelligent decision-making and collaborative cooperation.

References

[1] Sunehag P., Lever G., Gruslys A., ... & Graepel T, “Value-Decomposition Networks For Cooperative Multi-Agent Learning,” arXiv preprint arXiv: 1706.05296, 2017.

[2] C. Du, “Research on Multi-Agent Cooperative Adversarial Methods Based on Deep Reinforcement Learning,” Doctoral dissertation, Xidian University, 2020.

[3] Gronauer S., Diepold K, “Multi-agent deep reinforcement learning: a survey,” Artif Intell Rev 55, 2022, pp. 895-943.

[4] Silver D., Huang A., Maddison C, ... & Hassabis D, “Mastering the game of Go with deep neural networks and tree search,” Nature 529, 2016, pp. 484-489.

[5] Y. Chao, V. Akash, V. Eugene, ... and W. YI, “The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games,” Advances in Neural Information Processing Systems, Inc. Curran Associates, 2022, vol. 35, pp. 24611-24624.

[6] Yu C, Velu A, Vinitsky E, ... “Dynamic Traffic Signal Optimization with Multi-Agent Deep Reinforcement Learning,” arXiv preprint arXiv:1803.08094, 2018.

[7] Marcin Andrychowicz, Anton Raichuk, Piotr Stanczyk, ..., “What matters for on-policy deep actor-critic methods? a large-scale study,” In International Conference on Learning Representations, 2021.

[8] Luo C, Liu X, Chen X, ..., “Multi-agent Fault-tolerant Reinforcement Learning with Noisy Environments,” 2020 IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS), IEEE, 2020, pp. 164-171.

[9] Schulman J, Wolski F, Dhariwal P, ..., “Proximal Policy Optimization Algorithms,” arXiv preprint arXiv:1707.06347, 2017.

[10] J. Wang, L. Cao, X. Chen, ... “A Comprehensive Review of Multi-Agent Game Reinforcement Learning,” Computer Engineering and Applications, 2019, 55(8), pp. 197-207.