An Improved PPO for Multiple Unmanned Aerial Vehicles

Xue Bai, Chengxuan Lu, Qihao Bao*, Shansheng Zhu, Shaojie Xia
The 52nd Research Institute of China Electronics Technology Group Corporation. No.9, Wenfu Road, Yuhang, Hangzhou 311100, China

*baoqihao@cetc.com.cn

Abstract. In recent years, multi-agent reinforcement learning (MARL) has been applied widely, especially in large multi-role games such as StarCraft and unmanned aerial vehicles (UAVs) combat simulations. However, MARL is faced with challenges regarding fast convergence and efficient cooperation. In a multi-agent scenario, on the one hand, when a fully centralized network model is adopted, it is difficult for the model to converge due to the huge action space; on the other hand, it is difficult for a decentralized model to cooperate and achieve global optimization. To jointly control multiple agents, we propose an improved PPO algorithm by combining a centralized network and decentralized networks. Our method not only reduces the action space and accelerates the convergence, but also introduces more diversity for agents’ decision-making.

Keywords: Multi-agents, Reinforcement Learning, Centralized, Decentralized, Global Optimization.

1. Introduction

In recent years, with the development of artificial intelligence, the combat pattern of UAVs has gradually evolved from "flying alone" to "swarm intelligence". With cloud computing, big data, artificial intelligence, UAVs groups can perform coordination, strategic deterrence, and confrontation in military operations. For intelligent UAV group combats, the multi-agent cooperative combat is a trend. The development of military strategies utilizing the interaction information between individuals and the environment is a challenging problem. Reinforcement learning (RL) is suitable for solving such decision-making problems. RL has played a great role in solving game problems in recent years, such as Alpha Go, robot control, automatic pilot, and so on. For a UAV group combat problem, it comes to MARL.

There are multiple agents in MARL, and each agent is an autonomous subsystem that interacts with the same environment. The change of the environment depends on the joint action of all agents. In multi-agent scenarios, multiple agents need to achieve their goals, which may involve the communication between agents and the coordination setting of the goals [1]. At present, there are many kinds of researches of multi-agent combine centralized methods and decentralized methods [2][3], also they have gained some attention of the researchers in RL [4][5]. In MARL, one of the key issues is how to balance between centralization and decentralization, which is the core issue of extending single-agent RL to MARL. When using a fully centralized method, although it is possible to make accurate decisions from the perspective of global optimization, the joint action space will exponentially grow with
the number of agents increasing. The complexity of the deep network will also increase, leading to convergence difficulty. While using a decentralized method, each entity’s action is decided by a single agent. Despite this method eases the curse of the dimension of action space, another problem appears. Since each agent chooses action according to its independent observation, this approach obviously cannot consider the interaction between the agents and may not converge because the learning of each agent is interfered with by the learning and exploration of other agents.

Focusing on the UAV group combat scenario above, this paper combines a centralized network and decentralized networks and improves the widely used PPO [6] to implement multi-agent centralized training and decentralized execution. During the training, it makes use of the global state to train the global optimal strategy. A decentralized execution strategy can remove the communication restriction between multiple agents and avoid the bottleneck in group combats. The following contents as follow. Section 2 describes the current research progress of MARL; Section 3 formulates the UAV group combat problem and proposes our method, including improved design of network and loss function; Section 4 is the simulation experiments; Section 5 is a summary.

2. Related work
There are some value-based MARL methods. VDN [7] simply sums up each agent's local action-value function as the joint action-value function, meeting the requirement that the joint value function has the same monotonicity as the local value function. But it applies the linear method when integrating the value function, which ignores much available information. QMIX [8] uses a network to integrate agent functions, by keeping the network weights positive, the joint value function has the same monotonicity as the local value function. However, value-based RL methods have no advantage in exploration and continuous action space.

In addition to the above value-based RL methods, PPO, DDPG [10], and other widely used policy-based RL methods are also important. How to extend these RL methods to the field of multiple agents is challenging. Since each agent is constantly learning to improve its strategy, from the perspective of each agent, the environment is dynamically unstable and does not conform to the traditional RL convergence conditions. In MADDPG [9] the critic and the actor-networks are trained in a centralized way. Each critic uses other agents’ strategies for learning, so each agent has its own reward function. During the execution, each actor can make decisions only by using its local information. MADDPG can solve the nonstationary environment problem, however, every agent corresponds to an actor and a critic, so independent training often causes a lack of information sharing. COMA avoids the problem by training a critic as the global critic. However, the reward obtained is global, so it is difficult to evaluate how much the action taken by each agent affects the global reward.

To make all the agents share a part of parameters and ensure that each agent executes the personalized behaviors at the same time, this paper proposes the combination of the shared and independent networks.

3. Model

3.1 Scene
With the continuous development of 3D printing technology, the price of a lightweight and miniaturized UAV is far lower than that of a large wheeled UAV. Combined with technologies such as data mining and deep learning, combat in the form of a UAV group in the military has become a reality. In addition, due to the low cost of a UAV, the enemy needs to spend tens or even hundreds of costs to defend against a UAV group, which will bring significant advantage in a war.

In the following part, we formulate the joint defense operations of a UAV group. Due to the large number and low cost of UAVs, during the confrontation, a UAV group can cause huge damage by pursuing and hitting the other party's equipment with less self-damage.

Suppose that the red side (our side) has M small UAVs, and the enemy (i.e. the blue side) has N large UAVs (M>N). During the confrontation, we need to hit the blue side’s UAVs as much as possi-
ble to make them lose combat effectiveness. In addition, it is assumed that both sides can only fight in a certain area. Each episode can be divided into $T$ periods, and after each period, the enemy UAV’s alive state is $S_n^t$ ($n = 1, 2, ..., N$) (1 means alive, 0 means damaged). The purpose is to maximize the loss of the blue side by controlling the UAVs of the red side:

$$\max \sum_{n=1}^{N} (1 - S_n^t)$$

(1)

We assume that when the distance between two UAVs approaches less than a small value ($\varepsilon$), both UAVs will lose:

$$S_m^t = 0, S_n^t = 0, \text{if} \sqrt{(x_m^t - x_n^t)^2 + (y_m^t - y_n^t)^2} < \varepsilon$$

(2)

(3) is the setting when the UAV is damaged, it will no longer exist next time.

$$S_m^t = 0 \text{ if } S_m^{t-1} = 0, S_n^t = 0 \text{ if } S_n^{t-1} = 0, \forall m \in [1, M], \forall n \in [1, N]$$

(3)

In addition, it should meet the limitation that UAVs do not exceed the combat area:

$$boder_x_{min} < x_m^t < boder_x_{max}, \forall m \in [1, M]$$

(4)

$$boder_y_{min} < y_m^t < boder_y_{max}, \forall m \in [1, M]$$

(5)

As a UAV group confrontation is a continuous decision-making process, it satisfies the condition of partially observable Markov and can be modeled as a deep RL problem. For this study, we developed a multi-agent model and applied RL techniques to control UAVs.

3.2 Design
Fig.1 A modified framework based on PPO

In our design, each UAV of the red side is set as an agent, however, the action space may be different for each agent. A UAV’s action space may be discrete, continuous, or a combination of both. To make use of the global state when making decisions for each agent, and take the difference of decision-making of each agent into account, the design is shown below.

Our design is based on the PPO algorithm, which contains two actor networks (the Actor and the Old Actor) and the Critic network. As shown in figure1, each actor is composed of the sharing network and independent networks. The shared network receives the global state, which extracts the characteristics of the state. After the intermediate layer results are obtained, those intermediate results are sent to each independent network. Each agent corresponds to an independent network, which is used to meet the needs of the personalized action output of each UAV.

The network design of the Old Actor is the same as that of the Actor, and both receive the global state and output policy of each agent. In PPO, the Old Actor updates its parameters by copying the Actor’s parameters. When calculating the Actor loss, by limiting the ratio of the new policy and the old policy, the updating amplitude of the Actor can be limited. As a result, learning stability is enhanced. In this paper, the ratio of the old and new policy of each agent is calculated first:

\[
\text{ratio}_i = \frac{\text{policy}_i(s_{t+1})}{\text{old}_i\text{policy}_i(s_{t+1})}, i \in [1, N]
\] (6)

Then the clipped ratio is multiplied by the total advantage function to obtain the surrogate loss of each agent:

\[
\text{adv} = \delta_t + \gamma \delta_{t+1} + \cdots + \gamma^{T-t+1} \delta_{T-1},
\] (7)

where \( \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t) \) (8)

\[
\text{surr}_i = \min(\text{ratio}_i \ast \text{adv}, \text{clip}(\text{ratio}_i, 1 - \varepsilon, 1 + \varepsilon) \ast \text{adv})
\] (9)

Finally, the sum of the surrogate losses of all agents is the loss function of the Actor:

\[
\text{loss}_{\text{actor}} = \sum_{i=0}^{N} \text{surr}_i
\] (10)

In the training process, to enable agents to achieve cooperation and make decisions from the perspective of global optimization, the input state includes speeds, locations, and other information of all agents. At the same time, the reward is set based on the global situation, in other words, the contribution of each agent to the total reward is not distinguished, and it is learned by the policy gradient itself. During execution, the shared network and the independent network related to each agent are loaded, without relying on the communication between agents, there is no bottleneck.
4. Experiments
To perform experiments, we assume that, for each UAV, the optional decisions include five actions: up, down, left, right, and no move. The input state includes the agents’ speeds, positions, and distances from the combat boundaries and from other agents. In addition, according to the optimization objective proposed in Section 3.1, we hope that the UAVs of the red side can approach the enemy UAVs as close as possible. The reward is set in a global way, so the shorter distance between our UAVs and the enemy UAVs, the bigger the reward will be. Besides, the penalty will be added when our UAVs approach the boundaries.

The following is the simulation under different scene settings. In Fig.2, the number of our UAVs is 3 and the number of enemy UAVs is 1, we compare our method against the fully centralized PPO. In the case of a small number of agents, both methods can converge. But, apparently, the proposed method can converge faster and reach a higher average reward. Fig.3 shows the simulation of a scene where the number of our UAVs is 4 and the number of enemy UAVs is 2. Obviously, our method still can converge but the contrast method cannot.

5. Conclusion
In this paper, we have proposed an improvement of the PPO algorithm for multi-agent cases. The neural network includes the shared part and the independent parts. The shared network is responsible for the comprehensive analysis of global information, while each independent part is responsible for each customized analysis and action output of each agent. During execution, each agent loads the same trained shared network and the independent network of the agent, which does not depend on the communication between agents. The results show that the improved method significantly improves the convergence speed and performance.
References

[1] Kaiqing Z, Zhuoran Y and Tamer B 2019 Multi-agent reinforcement learning: a selective overview of theories and algorithms arXiv preprint arXiv: 1911.10635

[2] Frans A, Matthijs T, Shimon W and Nikos V 2008 Exploiting locality of interaction in factored Dec-POMDPs Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 1 International Foundation for Autonomous Agents and Multiagent Systems Richland SC pp517–524

[3] Landon K and Bikramjit B 2016 Multi-agent reinforcement learning as a rehearsal for decentralized planning J. Neurocomputing 190 pp82-94.

[4] Emilio J, Mikael K, Emil G and Fredrik J 2016 Learning to play guess who and inventing a grounded language as a consequence arXiv preprint arXiv:1611.03218.

[5] Jakob F, Gregory F, Triantafyllos A and Nantas N 2018 Counterfactual multi-agent policy gradients AAAI Conference on Artificial Intelligence

[6] John S, Filip W, Prafulla D, Alec R and Oleg K 2017 Proximal policy optimization algorithms arXiv preprint arXiv:1707.06347.

[7] Peter S, Guy L, Audrunas G, Wojciech M, Vinicius Z, Max J, Marc L, Nicolas S, Joel Z, Karl T and Thore G 2018 Value-decomposition networks for cooperative multi-agent learning based on team reward International Conference on Autonomous Agents and Multi-Agent Systems

[8] Tabish R, Mikayel S, Christian S, Gregory F, Jakob F and Shimon W 2018 QMIX Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning International Conference on Machine Learning

[9] Ryan L, Yi W, Aviv T, Jean H, Pieter A and Igor M 2017 Multi-agent actor-critic for mixed cooperative-competitive environments J. Advances in Neural Information Processing Systems

[10] Timothy P, Jonathan J, Alexander P, Nicolas H, Tom E, Yuval T, David S and Daan W 2016 Continuous control with deep reinforcement learning Ninth International Conference on Learning Representations