Training framework based on multi model competition for deep reinforcement learning

Yixian Lin¹
¹ Xiamen University, Xiamen, Fujian, China
e-mail: 24320191152527@stu.xmu.edu.cn

Abstract. In this paper, we propose a simple, effective and universal deep reinforcement learning training framework, inspired by A3C algorithm and genetic algorithm. The framework uses multi-process technology to realize multiple model competition and optimal evolution to optimize the deep neural network during the training process. The experimental results show that the proposed training framework can improve the training effect of reinforcement learning algorithms to a certain extent.

1. Introduction
At present, deep reinforcement learning is a new research hotspot in the field of artificial intelligence. It combines the perception ability of deep learning with the decision-making ability of reinforcement learning in a general form, and can realize the direct control from the original input to the output through the end-to-end learning. Since it was proposed, deep reinforcement learning has made a substantial breakthrough in many tasks that need to perceive high-dimensional raw input data and decision control. At present, the common deep learning algorithms are DQN[1], SAC & SAC-Discrete [2], PG[3] and so on. However, reinforcement learning based on deep neural network has the problem of unstable effect. It takes a lot of time, but the training effect is not good.

One of the reasons is that the data collected by RL agent in each training process is different and has a certain randomness, and the samples collected in a training process have a strong correlation in time. A technology called memory replay is proposed. By storing the data of the agent in an experience replay memory, samples of different time steps are randomly extracted from the memory every time the model is updated. This method reduces the time connection of samples and improves the training effect of deep reinforcement learning to a certain extent. Deep reinforcement learning algorithm based on memory replay has achieved remarkable success in many fields. A3C algorithm[4] uses multiple process and other means to execute their respective agents in multiple independent environment instances to collect samples, calculate gradients, and upload gradients to the global model, so as to break the connection between samples.

But memory replay has the advantage that one sample is used many times, and the utilization rate of data sample is high. In order to combine the advantages of the A3C and memory replay, inspired by the genetic algorithm[5], we propose a training framework in this paper. Combined with the idea of A3C and genetic algorithm, we train multiple agents in parallel. Each agent has an independent experience replay memory. After reaching the specified conditions, we select a current optimal model and assign its parameters to the other models, realizing the evolution of model population. Our method retains the advantages of memory replay, but also absorbs some of the advantages of A3C, and because of the continuous optimal selection in the training process, it can effectively improve the final
model training effect. At the same time, the training framework is separated from the specific deep reinforcement learning algorithm, so it is more universal in use.

2. Related work
In this part, we briefly introduce reinforcement learning, A3C algorithm in reinforcement learning, and genetic algorithm. We mainly introduce the concepts closely related to our training framework, but not the algorithm and its theoretical content.

2.1. Reinforcement learning
It can be seen from the Figure 1 that the components of reinforcement learning[6] include agent, environment and reward.

RL agents interact with the environment over time. At each time step \( t \), the agent receives a state \( s_t \) in the state space \( S \). Then according to the policy \( \pi(a_t|s_t) \), select an action \( a_t \) from the action space \( A \), where \( \pi \) is a mapping from states \( s_t \) to actions \( a_t \). According to the dynamic environment or model, the agent receives a scalar reward \( r_t \), and move to the next state \( s_{t+1} \), for reward function \( R(s, a) \) and state transition probability \( P(s_{t+1}|s_t, a_t) \), respectively. In an episodic problem, the process continues until the agent reaches the terminal state and starts again. Return \( R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \) is the cumulative discount reward from time step \( t \) with discount factor \( \gamma \in (0,1] \). The goal of the agent is to maximize the expected return from each state \( s_t \).

![Figure 1. Reinforcement learning structure.](image)

2.2. A3C algorithm
As shown in Figure 2, the main idea of A3C algorithm is to execute multiple agent replicas in multiple independent environmental instances in parallel. During the training process, each agent replica interacts with their respective environment, collects sample data, calculates the gradient of local neural network, and then uploads gradient to update the global neural network, then download the latest global neural network to local network. The global network cannot be updated by multiple agent replicas’ uploaded gradients in the same time.
2.3. Genetic algorithm

Genetic algorithm is a computational model simulating the natural selection and genetic mechanism of Darwinian biological evolution theory. It is a method to search the optimal solution by simulating the natural evolution process.

As the main carrier of genetic material, chromosome is the collection of multiple genes. Its internal expression (genotype) is a certain gene combination, which determines the external expression of individual shape. For example, the characteristic of black hair is determined by a certain gene combination controlling this characteristic in chromosome. Therefore, it is necessary to realize the mapping from phenotype to genotype at the beginning. Due to the complexity of gene coding, we often simplify it, such as binary coding.

After the emergence of the primary population, according to the principle of survival of the fittest, generation by generation evolution produces better and better approximate solutions. In each generation, individuals are selected according to the fitness of individuals in the problem domain. With the help of the genetic operators of natural genetics, the population representing the new solution set is generated by combining crossover and mutation.

This process will lead to the natural evolution of the population. The later generation population is more adaptable to the environment than the previous generation. After decoding, the optimal individual in the last generation population can be used as the approximate optimal solution of the problem.

3. Method

In this part, we present the training framework based on multi model competition. Inspired by genetic algorithm, combined with the asynchronous training form of A3C algorithm, we propose a reinforcement learning training framework based on multi model competition, shown as Figure 3. According to the idea of genetic algorithm, we divide the training process of the framework into the following steps: initialization of the original population, individual learning, evaluation of individual environmental fitness, individual selection, and population evolution. Next, we go through each step in turn.
• **Initialization of the original population:** Based on the A3C form, we use the multi thread method on the multi-core CPU to initialize the neural network model randomly on each thread. Each neural network is an independent individual, and multiple neural networks on multiple threads form a neural network population.

• **Individual learning:** neural network on each thread is trained by reinforcement learning algorithms such as DQN and A2C. In our experiment, each model uses a unified algorithm for training. In theory, each model can use different reinforcement learning algorithms for training.

• **Evaluation of individual environmental fitness:** after the specified iteration training, we measure the environmental fitness of each model individual according to the evaluation criteria. The higher the environmental fitness, the better the training effect.

• **Individual selection:** select the best model individual at the current stage based on environmental fitness to reproduce the next generation of model population.

• **Population evolution:** each model individual in the population updates its own parameters to the parameters of the model selected, so that each individual in the current population is the current best, so as to realize the evolution of the population.

![Figure 3. Training framework](image)

### 4. Experiment

In this part, we conduct experiment and evaluate the effect of the training framework, mainly the experimental settings and the analysis of the experimental results.

We verify the effect based on the cartpole-v0 of gym. In cartpole-v0, there is a pole on the cart, and the cart can move along the track. The pole can be kept upright by exerting right or left force on the cart. The agent determines the action that should be taken in a time step according to the returned state of the environment. The pole will get a +1 reward for every time it keeps upright. When the angle between the pole and the vertical direction exceeds 15 degrees, or the distance between the cart and the central position exceeds 2.4 units, the episode ends.

We take the reward obtained by each model in the test episode as the standard of environmental fitness of the model. The higher the reward model gets, the higher the fitness, and the model parameters with the highest environmental fitness in the test will be updated to all other models to realize population evolution. In this experiment, A2C algorithm is used as the learning algorithm, the learning rate is set to 0.001, and the model selection is made after every five episodes. We use four threads to create four individuals forming the model population.
We named the individuals in the population as c-A2C according to the A2C learning algorithm they used. Compared with the A2C algorithm alone, we can see the models trained with multi model competition training framework from Figure 4 obtain higher reward and the training effect is better than that of using A2C algorithm alone. It shows that the training framework based on multi model competition can improve the training effect of deep reinforcement learning, because based on this framework, we can select the current optimal model in the training process, and continue to optimize the model on the basis of the current optimal.

5. Conclusion
In this paper, we propose a training framework based on multi model competition, and show that the framework can effectively improve the training effect of deep reinforcement learning. Our experimental result show that using our proposed training framework can make the model get higher reward and better training effect. At the same time, the training framework does not depend on the specific algorithm, so it has a wide range of versatility and flexibility.

References
[1] van Hasselt, H., Guez, A., & Silver, D. (2016). Deep Reinforcement Learning with Double Q-Learning. Proceedings of the AAAI Conference on Artificial Intelligence, 30(1).
[2] Christodoulou, P. (2019). Soft Actor-Critic for Discrete Action Settings. ArXiv, abs/1910.07207.
[3] Sutton, R. S., Mcallester, D., Singh, S., & Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. Submitted to Advances in Neural Information Processing Systems, 12.
[4] Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D. & Kavukcuoglu, K. (2016). Asynchronous Methods for Deep Reinforcement Learning. Proceedings of The 33rd International Conference on Machine Learning, in Proceedings of Machine Learning Research 48:1928-1937.
[5] Davis, L. D. (1991). Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York.
[6] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.