DEEP REINFORCEMENT LEARNING BASED GAME DECISION ALGORITHM FOR DIGITAL MEDIA EDUCATION

ZU-NING LI[1], PING-KUANG[1], TING ZHANG[1], HUA-RUI YAN[1], XIAO-FENG GU[1]

[1] School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610054, China
E-MAIL: l.d.vc@outlook.com

Abstract:

On the traditional digital media education platform, problems raised by users are usually solved by searching the optimal solution in the existing database, which lacks the ability of autonomous learning and long-term planning. This paper introduces a Deep Reinforcement Learning algorithm based on Deep Deterministic Policy Gradient to solve these issues. We applied a trained neural network with this algorithm to play “Klotski” - a sliding block puzzle where the aim is to move a specific block to some predefined location. In this article, we explain why the game is selected and how our method works. We also discuss possible ways forward within the field in the future.

Keywords:

Deep reinforcement learning; machine decision; digital media education

1. Introduction

It is known that "the world's three major educational games" including Klotski, Rubik's Cube and Peg solitaire. We can see them in many digital education games. The reason why we chose the Klotski to conquer is that, different from other board games, Klotski pins much higher expectations on players’ long-term planning ability. From the very beginning, players are set in a dilemma where they are forced to embrace a long-term movement plan. Otherwise they are easy to get stuck. So far, the best solution of Klotski is no less than 81 steps, which is computed by computer through the Exhaustive Attack Method.

DeepMind and OpenAI have harvested remarkable achievements in the field of game playing with the adoption of reinforcement learning. The point is that their algorithm which fundamentally changes the nature of learning problems, decides human teaching to be an integral part of it. However, to solve simulation games by deep reinforcement learning is not the ultimate objective. We want to empower machines with the capability of provide solutions to users with limited priori data.

We propose an algorithm based on Deep Deterministic Policy Gradient to enable machines perform simulation games, like “Klotski”, independently without any priori data. In this way, digital media education platform can solve the long-term planning problem through self-learning under given conditions.

The rest of the paper is organized as follows. Section 2 presents related work. Section 3 introduces the methods proposed in this paper. Section 4 gives the experimental results. Finally, Section 5 gives conclusions and future work.

2. Related Work

With the development of artificial intelligence technology, in some complex decision-making tasks, it is usually necessary to use the Deep Learning (DL) methods to obtain the abstract features of large-scale input data, and adopt the Reinforcement Learning (RL) method to automatically optimize the strategy of solving problems according to the extracted features. Therefore, some researchers have combined the two methods and formed a new research hotspot in the field of artificial intelligence, called the Deep Reinforcement Learning (DRL). Here are some related researches in the field of deep reinforcement learning.

Deep Q-Learning Network (DQN) is said to be the beginning of deep reinforcement learning. DeepMind firstly put forward DQN algorithm to achieve Video-only image input, completely learning the outcome of Atari games through Agent learning. Then DeepMind published an improved DQN article [1] on Nature, which combines deep learning with reinforcement learning to implement an entirely new learning algorithm from Perception to Action end-to-end.

In order to eliminate the correlation between samples, DQN uses the experience replay mechanism to store and use the historical samples obtained through agent's interaction with the environment online. However, uniform probability sampling cannot distinguish the importance of different samples. At the same time, some samples are been discarded before they are fully utilized. To solve this problem, Schaul et al. put forward a double deep Q network with proportional prioritization [2]. This method replaces uniform sampling with priority sampling to improve the sampling probability of some valuable samples and thus accelerate the learning process of the optimal strategy.

By stacking four historical images closest to the current moment to form the input state, DQN effectively alleviates the partial observability of state information, but increases the computational and storage burden of the network. To solve this problem, Hausknecht et al. use the structure of Recurrent Neural Network (RNN) to memorize the continuous historical state information along the time axis, and proposed the Deep Recurrent Q Network (DRQN) model [3]. Experimental results show that DRQN performs better than DQN under partially observable conditions.

Lillicrap et al. modified the deterministic policy gradient method using the DQN extended Q learning algorithm and proposed an algorithm based on the Actor-Critic framework called Deep Deterministic Policy Gradient (DDPG) [4]. This algorithm solves the DRL problem in continuous motion space perfectly.

In addition to DDPG algorithm, another well-known strategy-based reinforcement learning algorithm is Distributed Proximal Policy Optimization (DPPO) proposed by Schulman et al. in 2017 [5]. DPPO is an improved version of Trust Region Policy Optimization (TRPO) [6], which is applicable to many fields and a general optimization idea. Zhang [7], Balduzzi [8] et al. also studied the application of strategy gradient method in deep reinforcement learning.

3. Methods

We chose the Deep Deterministic Policy Gradient algorithm as the backbone of our proposed method, because DDPG is based on Actor-Critic framework, which is applicable to solving the planning problem of continuous motion space. Then, we made some improvements on the basis of DDPG algorithm.

3.1. Deep Deterministic Policy Gradient

The Deep Deterministic Policy Gradient algorithm is model free, off-policy, and uses a deep neural network for function approximation. The following is a brief explanation of relevant terms.

- μ is a strategy function, and the action of each step can be calculated by $a_t = \mu(s_t)$.
- The Q function is simulated by a convolution neural network, with parameter θ^Q.
- The performance of strategy μ is measured by function J.
- Training goal: Maximize the $J_\beta(\mu)$ while minimizing the loss of Q network.
- Optimal Q Network: It is where a Minimized Q Network Loss is achieved.

According to the above definitions, the pseudo-code of DDPG algorithm is shown in table 1.

Table 1 The DDPG algorithm

Pseudocode

```
Randomly initialize critic network $Q(s,a|\theta^{\mu})$ with weights $\theta^Q$ and $\theta^\mu$;
Initialize target network $Q'$ with weights $\theta^{Q'} \leftarrow \theta^Q, \theta^{\mu'} \leftarrow \theta^\mu$;
For ( $episode = 1$ ; $episode < M$ ; $episode{+}{+}$ ):
    Initialize a random process $N$ for action exploration; receive initial observation state $s_1$;
    For ( $t$=1; $i < T$; $i{+}{+}$ ):
        Select action $a_t = \mu(s_t|\theta^\mu) + N_t$ according to the current policy and exploration noise;
        Execute action $a_t$ and observe reward $r_t$ and observe new state $s_{t+1}$;
        Store transition $(s_t, a_t, r_t, s_{t+1})$ in replay buffer $R$;
        Sample a random minibatch of $N$ transitions $(s_i, a_i, r_i, s_{i+1})$ from $R$;
        Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$;
        Update critic by minimizing the loss:
            $L = 1/N \sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$;
        Update the actor policy using the sampled policy gradient:
            $\nabla \theta^\mu 1/N \sum_i (\nabla_a Q(s, a\theta^Q)|_{(s=s_i, a=\mu(s_i))} \nabla_{\theta^\mu} \mu(s|\theta^\mu)_{s_i}$
        Update the target networks:
            $\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$    $\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$
```

3.2. Improvements

Our method is based on DDPG, but there are some improvements, including:

1) Random perturbation is added on the basis of the probability array of actions;

2) At the end of each episode, the method of smoothing reward value is adopted. The reward value is smoothed forward according to a certain attenuation ratio.

The flow chart of our method is shown in fig.1. These improvements enable the algorithm to solve long-term problems, and make the capacity of final network more stable, the effect better. See chapter 4 for detailed experimental results.

In the execution of our method, the process of selecting an action from the possibilities array is shown in table 2, and the process of reward value smoothing is shown in table 3.

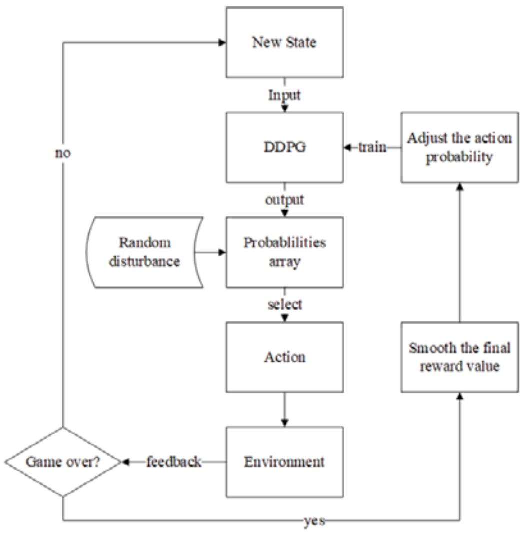

Fig.1 The flow chart of our method

Table 2 Selecting action from possibilities array

```
Pseudocode
$p$ represent the array of possibilities predicted by network;
For ( $i$=0 ; $i < Outputdim$ ; $i$++ ):
    Set $alpha = 0.5$ ;
    Update $probability$ :
        $probability \leftarrow alpha * p[i] + (1.0 - alpha) * full\_like(p[i], 1.0 / p[i].shape[0])$;
        $action \leftarrow random.choice(p[i].shape[0], 1, probability)$;
```

Table 3 Reward value smoothing

```
Pseudocode
Set $runningAdd = rewards[-1]$ ;
Set $discountedRewards[-1] = rewards[-1]$ ;
For ( $i$=rewards.size[-1] ; $i < 0$ ; $i--$ ):
    If $rewards[t] = -1$ Then:
        Return $discountedRewards[t] = -0.5$ ;
    Update $runningAdd$ : $runningAdd \leftarrow runningAdd * gamma + rewards[t]$ ;
    Set $discountedRewards[t] = runningAdd$ ;
```

4. Experiments

Our approach is evaluated in a relatively simple scenario of the Klotski puzzle game. The initial chessmen distribution is shown in fig.2. The green rectangle in the upper left corner represents Cao Cao, the other squares represent obstacles. Our ultimate goal is to move Cao Cao to the target position (red dotted line in Figure 2) with as few steps as possible. After about 700 episodes of continuous learning and trials, our neural network learned a solution successfully, as shown in fig.3. The numbers in the picture represent the sequence of steps.

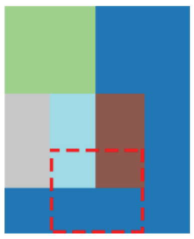
Fig.2 Initial chessmen distribution

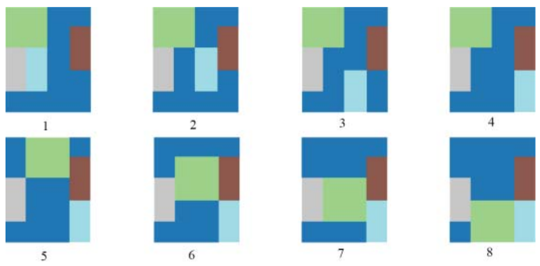

Fig.3 An example of solution

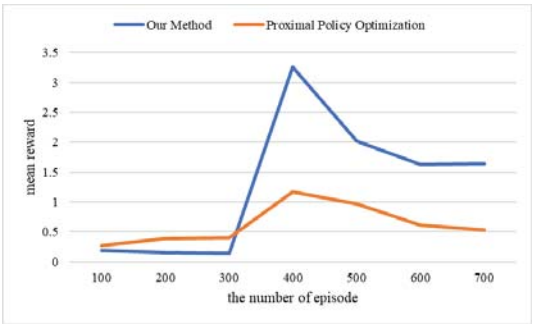

Fig.4 The change of mean reward

We also used the proximal policy optimization algorithm [5] to compare with our method, and the change of reward value with episode is shown in fig.4. Compare with the former, we can conclude that the DDPG-based reinforcement learning framework proposed in this paper has a certain degree of long-range planning capabilities. However, when we increase the difficulty of the game scenario, for example, to add a horizontal obstacle to the board (which is called “Guan Yu” in the traditional Klotski puzzle game), it seems that our agent can hardly find a solution and easily get stuck. This is because it is challenging for the agent to find the target position in the random process of training under relatively complex scenarios. In order to solve this problem, a feasible way is to create a moderately difficult game scenario: Guan Yu is set beside Cao Cao instead of in front of him. In this case, Cao Cao can still be able to find a way out. After trained under moderate difficulty conditions, the network can learn in relatively complex scenarios. This is because the network has learned some experience in the moderately difficulty scenario.

5. Conclusions

Simulation game is the touchstone of digital media education, to test whether the deep reinforcement learning preludes capabilities of reasoning and decision. Such capability covers learning concepts at multiple levels of abstraction, including the adaptable promotion of conceptual knowledge learned from a problem space to many problem spaces.

In this work, we proposed an algorithm with long-range planning ability that can solve Klotski-like problems. We analyzed the experiment results and found that the key lies in whether the network could find the export target in a random process. The proposed method is empowered to play and win Klotski puzzle game completely without human intervention.

At present, our attempts only support to make the machine pass the simple scene of Klotski. However, we are determined to go further in regards of the depth of reinforcement learning and explore a set of optimization algorithms. We look forward to the day when AI agents can learn to play even harder simulation games.

Acknowledgements

This work supported by Sichuan Science and Technology Program (2018CC0136, 2019ZDZX0009, 2019ZDZX0005).

References

[1] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature,* vol. 518, no. 7540, p. 529, 2015.

[2] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952,* 2015.

[3] M. *Hausknecht* and P. Stone, "Deep recurrent q-learning for partially observable mdps," in *2015 AAAI Fall Symposium Series*, 2015.

[4] T. P. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971,* 2015.

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

[6] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. *Abbeel*, "Trust Region Policy Optimization," *Computer Science,* pp. 1889-1897, 2015.

[7] T. *Zhang*, G. Kahn, S. Levine, and P. Abbeel, "Learning Deep Control Policies for Autonomous Aerial Vehicles with MPC-Guided Policy Search," in *IEEE International Conference on Robotics & Automation*, 2016.

[8] D. *Balduzzi* and M. Ghifary, "Compatible Value Gradients for Reinforcement Learning of Continuous Deep Policies," *Computer Science,* vol. 8, no. 6, p. A187, 2015.