Application of Q-Learning and RBF Network in Chinese Chess Game System

Wenyang He\(^{a}\), Wenlong Zhao\(^{a}\) and Yuan Jiang\(^{b}\)

School of Information Engineering, Nanchang Hangkong University, Nanchang 330063, China

*Corresponding author e-mail: 3347841990@qq.com, 747048364@qq.com, jiangyuan@nchu.edu.cn

Abstract. Machine game is one of the important directions of artificial intelligence research, and Chinese chess is a typical game process. This paper first introduces the game principle of Chinese chess. Then Q-learning method and evaluation function are added to train data through a large amount of self-learning. A chess game system based on Q-learning is designed and developed. The experimental results show that Chinese chess with Q-learning algorithm has the ability of evolutionary learning, which effectively improves the game level of Chinese chess.

1. Introduction

Machine game plays an important role in the field of artificial intelligence [1]. Many methods and techniques of artificial intelligence can be applied in machine games. Many methods and techniques of artificial intelligence can be applied in machine games. With the success of computer games in Othello, Checker and Chess, scholars all over the world have focused on more complex Chinese Chess and Shogi Go [2]. Comparisons of the complexity of several chess traversal searches are shown in Table 1.

| Board game     | Space complexity | Tree complexity |
|----------------|------------------|-----------------|
| Chess          | \(10^{27}\)      | \(10^{122}\)    |
| Chinese chess  | \(10^{52}\)      | \(10^{150}\)    |
| Shogi          | \(10^{11}\)      | \(10^{239}\)    |
| Go             | \(10^{160}\)     | \(10^{499}\)    |

In recent years, many scholars have introduced learning algorithms in the field of artificial intelligence to optimize and learn, such as backgammon, jump and chess [3]. Domestic research seldom involves self-adapting and self-learning and most of them use the traditional method to set the valuation function manually by relying on experience. The research focuses on how to improve the data structure, deepen the level of game tree search, and rely on the improvement of computer operation speed and multi-CPU parallel operation to strengthen the chess power of computer chess players.

This paper combines Q-learning algorithm with Chinese chess machine game [4]. Through Q-learning method, the internal relationship and game process between chess pieces are represented [5].
The Chinese chess problem can be explained by a game tree. The difference between game tree and general decision tree is that its decision-makers are not one party, but opposed to each other. We use red and black to represent them. In Figure 1, the square node represents the state of the red side. The circular node represents the state of the black side. The connection between square node and circular node corresponds to a collocation method. If a good chess player can look forward to four steps, it means that in his mind the game tree will be expanded into four layers (Depth), as shown in Figure 1.

![Figure 1. Four-tier Game Tree.](image)

Each node in the game tree represents a composition. For each node in the tree, the red and the black will choose the branch which is most beneficial to them from the sub-nodes. Because the value of the game tree transmits from bottom to top, it is required that the situation represented by leaf nodes must be scored very accurately. The most accurate assessment of the situation is the situation in which the winner and loser has been identified. That is to say, a complete game tree is established to distinguish the winner from loser at the leaf nodes.

2. RBF neural network model

2.1. Network structure

RBF (Radial Basis Function) neural network is a three-layer forward network, which consists of input layer, hidden layer and output layer. The input layer to the hidden layer is a non-linear relationship mapping, and the hidden layer to the output layer is a linear relationship mapping. It not only has highly efficient learning efficiency, but also avoids the local minimum problem. As shown in Figure 2, \(x_1, x_2, \ldots, x_n\) are the input of the input layer. \(h_i\) is a hidden layer Gauss basis function. \(w_1, w_2, \ldots, w_n\) are the weights from the hidden layer to the output layer. \(y_1\) is the actual output.

![Figure 2. RBF neural network structure.](image)
2.2. Jacobian identification algorithm

According to Figure 2, input variable \( \mathbf{x} = [x_1, x_2, \cdots, x_n]^T \) with input layer, Radial Basis Vector \( \mathbf{H} = [h_1, h_2, \cdots, h_p]^T \) of RBF Network, where \( h_j \) is Gauss Basis Function,

\[
h_j = \exp \left( -\frac{\| \mathbf{x} - \mathbf{C}_j \|^2}{2\mathbf{b}_j^2} \right), \quad j = 1, 2, \cdots, P
\]  

(1)

The center vector of the \( j \)th node of the network is \( \mathbf{C}_j = [c_{j1}, c_{j2}, \cdots, c_{ji}, \cdots, c_{jM}]^T \), \( i = 1, 2, \cdots, M \).

The base width vector for setting the network is \( \mathbf{B} = [b_1, b_2, \cdots, b_j, \cdots, b_p]^T \), \( b_j \) is the base width parameter of node \( j \) and is greater than zero. The output weight vector of the network is \( \mathbf{W} = [w_1, w_2, \cdots, w_p]^T \). The output of the identification network is

\[
y_i(k) = w_1h_1 + w_2h_2 + \cdots + w_ph_p
\]  

(2)

Taking the Performance Index Function of Identifier is

\[
J_0(k) = \frac{1}{2} \left[ y(k) - y_i(k) \right]^2
\]  

(3)

In order to get better identification effect of RBF network, an inertia term is added when the weight coefficient is revised. According to the gradient descent method, the output weight coefficient of RBF neural network, the base width parameters of node center and hidden node are modified as follows:

\[
\Delta \mathbf{w}_j = \left[ y(k) - y_i(k) \right] h_j
\]

\[
\mathbf{w}_j(k) = \mathbf{w}_j(k-1) + \eta_0 \Delta \mathbf{w}_j + \alpha_0 \left[ \mathbf{w}_j(k-1) - \mathbf{w}_j(k-2) \right] + \beta_0 \left[ \mathbf{w}_j(k-2) - \mathbf{w}_j(k-3) \right]
\]  

(4)

\[
\Delta b_j = \left[ y(k) - y_i(k) \right] w_j h_j \frac{\| \mathbf{x} - \mathbf{C}_j \|^2}{\mathbf{b}_j^2}
\]

\[
b_j(k) = b_j(k-1) + \eta_0 \Delta b_j + a_0 \left[ b_j(k-1) - b_j(k-2) \right] + \beta_0 \left[ b_j(k-2) - b_j(k-3) \right]
\]  

(5)
\[
\Delta c_{ji} = \left[ y(k) - y_i(k) \right] w_j \frac{x_j - c_{ji}}{b_j^2} \\
c_{ji}(k) = c_{ji}(k-1) + \eta_0 \Delta c_{ji} \\
+ \alpha_0 \left[ c_{ji}(k-1) - c_{ji}(k-2) \right] \\
+ \beta_0 \left[ c_{ji}(k-2) - c_{ji}(k-3) \right]
\]

In the formula, \(\eta_0\) is the learning rate, \(\beta_0\) and \(\alpha_0\) are the inertia coefficient, and \(\eta_0, \alpha_0\) and \(\beta_0\) are all selected on the interval \((0, 1)\).

The situation value after playing chess evaluates the position of playing chess as Jacobian information.

\[
\frac{\partial y(k)}{\partial \Delta u(k)} \approx \frac{\partial y_i(k)}{\partial \Delta u(k)} = \sum_{j=1}^{p} w_j h_j \frac{c_{ji} - x_i}{b_j^2}
\]

3. A brief introduction to q-learning algorithm

In this paper, RBP network is used to represent the evaluation function of Chinese chess situation. Q-learning method is used to adjust the weights of the network by learning [6]. This paper takes Chinese chess as an example to introduce this method of enhancing learning.

Assuming A1, A2, A3 \(\cdots\) Am are intermediate situation sequence, these situation are evaluated using the artificial neural network evaluation function mentioned in the previous chapter. At this point, the weights of the network are random values. Assume that the evaluation values corresponding to the situation sequence are \(Q1, Q2, Q3 \cdots Qm\), the final winning or losing result is expressed by the numerical value Z(e.g. -1 for the black side to win, 0 for the sum, 1 for the red side to win).

Q-learning is that in the early stages of learning is almost equal to random selection. The result of playing chess is far from that of winning chess. In a series of walks, occasionally there will be a similar way to win chess results. If so, this series of moves will be rewarded. Q learning will change Q value according to the reward received at this time. If the moves can get reward the corresponding Q value will increase and the action will be easier to choose. The updated formula of Q value is as follows:

\[
Q(s, a) = Q(s, a) + \\
\alpha (\gamma + \zeta \max Q(s_{t+1}, a_{t+1}) - Q(s, a))
\]

Where \(s_t\) denotes the state of time \(t\) and \(a_t\) represents the action chosen at time \(s_t\). \(\max Q(s_{t+1}, a_{t+1})\) denotes the maximum value of Q corresponding to the moving step that can be selected at the next moment (\(t+1\)). \(\gamma\) is a reward (only when it can be obtained, if \(\gamma\) can’t be obtained it is 0). \(\alpha\) is the learning coefficient (about 0.1) and \(\zeta\) is the discount rate (about 0.9). In Q learning, the formula (8) is used to update the Q value in order to reinforce learning.

4. Obtaining training data

4.1. Several methods of obtaining training data

After using RBF network to represent evaluation function and Q-learning as learning algorithm, a large number of training samples are still needed to train the network [8]. In the field of machine game, the
training data used for learning are a large number of chess spectrum sequences, which can be obtained by the following methods.

(1) Fixed opponent game

Fixed-opponent game refers to the computer and a fixed opponent that can be human experts or game software to match, in the continuous victory and failure to improve themselves. But the disadvantage of this method is that it is easy to produce evaluation function limited to the fixed opponent. When playing with other opponents, the computer will not adapt to the situation [12].

(2) Expert chess score database

There is a lot of chess knowledge in expert chess score. The advantage of learning directly from expert chess score is the fast learning speed. This is reflected in two aspects, on the one hand, because the chess score already exists, learning directly from the chess score can save the time of playing. On the other hand, because both sides of the game are human experts, they will not easily appear stupid tricks. The way they go through also represents the essence of Chinese chess playing, and can quickly improve the level of machine game.

(3) Online learning

Baxter used this method to train their chess program KnightCap. They run KnightCap on a network server and many players play chess through many clients and servers. This online learning method works well. Through 308 rounds in three days, KnightCap's grade increased from 1650 to 2150, which is equivalent to human's B level to human's master level.

(4) Random play

Random game includes two methods. One is to let the trained game software play with a game software that can generate random walk. Another method is to let the game software which can generate random walk and play with itself. The game sequence generated is used as training data.

4.2. Choosing the Appropriate Method

This paper focuses on the application effect of neuron network combined with Q-learning algorithm in Chinese chess evaluation [9]. The self-learning method is used when selecting the method of obtaining training data. After initializing the random Q value, the mobile chess pieces are selected based on the Q value, and the learning process is continuous. Figure 2 shows network combined with Q-learning algorithm.

![Network combined with Q-learning algorithm](image)

**Figure 3.** Neuron network combined with Q-learning algorithm.

5. Experimental

5.1. Experimental process

An 81x30x3 RBF network is established in the experiment. In the course of action, we should give priority to actions with high Q value, but simply choosing the actions corresponding to the maximum Q value is not a good way to conduct Q learning. If the action corresponding to the accidentally large Q value in the random number is initialized in this way, it will be selected all the time. No matter how many times the action is repeated, only this action will be selected, and no other action can be selected.
The $\epsilon$-greedy method is used here to deal with. The process is to assign a suitable constant between 0 and 1 in advance. When choosing moving chess pieces, a random number is generated between 0 and 1. If the value ratio is small, a random step is chosen. If this number is greater than $\epsilon$, choose the step to move with a large Q value. In this way, we can learn the appropriate Q value for all kinds of chess moves without depending on the initial value of Q value.

5.2. Experimental results

In order to test the effect of the experiment, the same game software with the same fixed chess value, position value, flexibility value and threat and protection value was used to play the game. In order to ensure the fairness of the game, the same search engine is used and the same search depth is used. Firstly, through self-study training, a software battle is conducted once after each study. In the first 50 matches, the process of starting each game is almost random movement of Chinese chess pieces. After 100 times training, playing chess with software began to only focus on killing the opponent's chess pieces. When the number of training reached 200 times, it began to have the ability to fight against software game. When the training reached 500 times, playing chess began to have a global concept, not focusing on killing a single piece of chess. When the number of training reaches 1000 times, the ability of playing chess with defensive consciousness will be enhanced obviously. From the number of training times and the number of battling times we can see that the ability of playing chess is constantly improving with the increase of the number of training times. Fig.4 shows the number of repetitions in Q-learning.

![Figure 4. Repetition times of training.](image)

From the above experiments, it is obvious that Q-learning can effectively improve the game level of Chinese chess. The experimental process of self-learning and software game proves that this method is feasible.

6. Conclusion

Chess game is a good experimental site for AI. The research on self-learning method of evaluation function is a rich research field, which can produce many useful technologies to solve the learning and planning problems in uncertain environment.

This paper explores the combination of Q-learning and neural network for Chinese chess machine game. Through the experiment of software design on Chinese chess, it is proved that Q-learning can improve the combat ability of Chinese chess very well.

There are many tasks to be done in the next step. It is necessary to further study various factors affecting the effect of Q-learning, to further optimize the neural network and to improve the Q-learning algorithm.
Acknowledgments
This work was financially supported by National Natural Science fund (No.61663030), Department of Education Science and Technology (No.GJJ150701).

References
[1] Sutton R S. Learning to Predict by the Methods of Temporal Differences [J]. Machine Learning, 1988, 3(1):9-44.
[2] J.E. Laird, "Using a Computer Game to Develop Advanced AI," Computer, vol. 34, no. 7, pp. 70-75, July 2001.
[3] Machine Learning - Artificial Intelligence; Study Results from University of Saarland in the Area of Artificial Intelligence Reported (Dynamics in matching and coalition formation games with structural constraints) [J]. Journal of Robotics & Machine Learning, 2018.
[4] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al. (2015) Human-Level Control through Deep Reinforcement Learning. Nature, 518, 529-533. http://dx.doi.org/10.1038/nature14236.
[5] Dayan W P. Q-learning [J]. Machine Learning, 1992(3); 279-292. Science, 1989.
[6] Tsitsiklis, John N. Asynchronous Stochastic Approximation and Q-learning [J]. Machine Learning, 1994(3):185-202
[7] Peter Dayan, Nicol N. Schraudolph, and Terrence J. Sejnowski (2001) Learning to evaluate Go positions via temporal difference methods. Computational Intelligence in Games, Springer Verlag, 74-96.
[8] Machine Learning; Study Results from Beijing Institute of Technology Update Understanding of Machine Learning (Resilient control for wireless networked control systems under DoS attack via a hierarchical game) [J]. Computers, Networks & Communications, 2018.
[9] Machine Learning - Artificial Intelligence; Study Results from University of Saarland in the Area of Artificial Intelligence Reported (Dynamics in matching and coalition formation games with structural constraints) [J]. Journal of Robotics & Machine Learning, 2018.
[10] Stamenković Lidija J., Antanasijević Davor Z., Ristić Mirjana Đ. et al.. Modeling of methane emissions using artificial neural network approach [J]. Journal of the Serbian Chemical Society, 2015, 80(3).
[11] Luke E. K. Achenie, Angela Scarpa, Reina S. Factor, Tao Wang,Diana L. Robins, D. Scott McCrickard. A Machine Learning Strategy for Autism Screening in Toddlers [J]. Journal of Developmental & Behavioral Pediatrics, 2019, 40(5).
[12] Asakimori, K., Okuyama, A., Matsui, Y. et al.. Computer literacy education using self-learning environment [C]. Information Technology Based Higher Education and Training, 2004. ITHET 2004. Proceedings of the Fifth International Conference on, 2004