Using Neural Network and Monte-Carlo Tree Search to Play the Game TEN

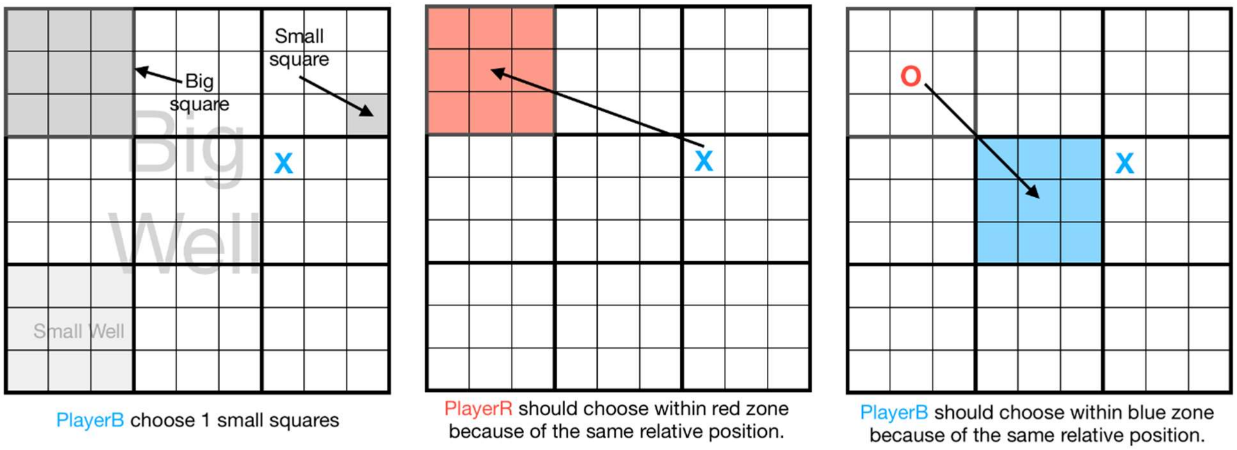
Weijie Chen

Fall 2017

1. INTRODUCTION

• Game TEN

The traditional game Tic-Tac-Toe enjoys people's favor. Moreover, the game evolves. There is an advanced version of Tic-Tac-Toe, called TEN in iOS. The TEN has two parts, Big Well and Small Wells. The Big part is traditional Tic-Tac-Toe 9 squares. However, every square in Big Well is divided into 9 small squares, a Small Well. At beginning, the first player chooses 1 big square and put a X or O in one of 9 small squares of this big square. Then next player should choose the big square, whose relative position of Big Well is the same as the relative positive of small square of Small Well. And the game goes as this rule.



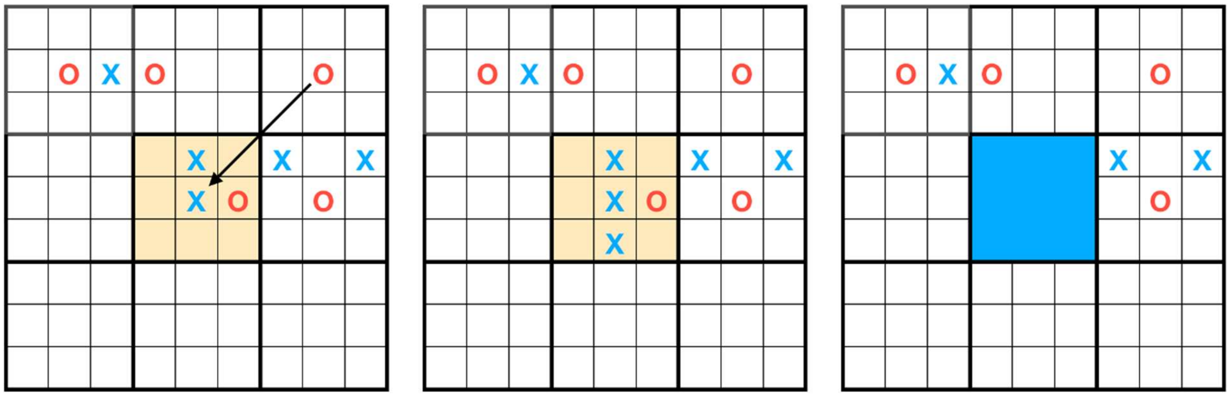
PlayerB choose 1 small squares

PlayerR should choose within red zone because of the same relative position.

PlayerB should choose within blue zone because of the same relative position.

Figure 1. Basic rules for Game TEN

Briefly, the current small square in Small Wells decide the next big square in Big Wells. When there are same 3 X or O in a row in one Small Well, this player wins. And the player occupies the big square in Big Wells. If one big square is occupied, the next player who should choose this big square, is free to choose any other unoccupied big squares.



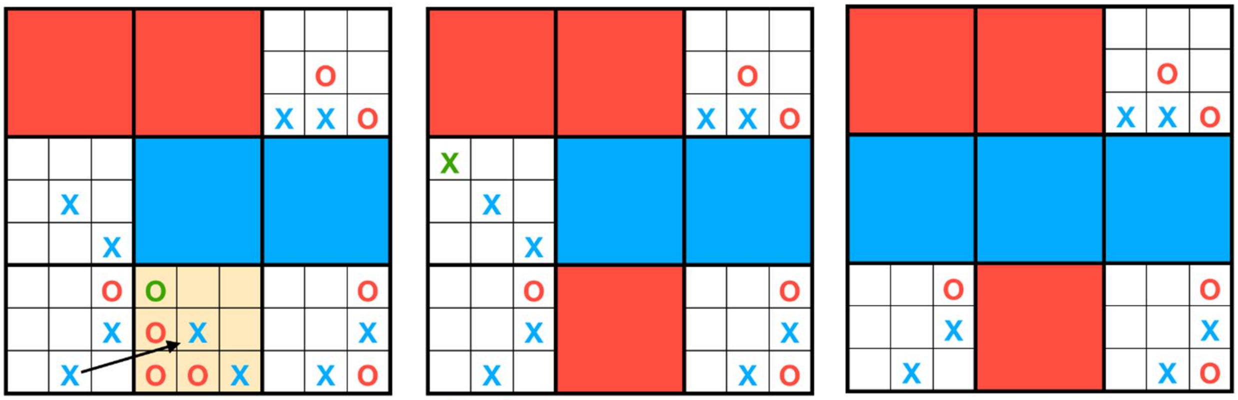
After some steps, PlayerB has a chance to line 3 Xs in the center big square

PlayerB has 3 X in a row now

PlayerB occupies the big square, and PlayerR can choose any other big square in its turns

Figure 2. Win a Tic-Tac-Toe in a Small Well

Who wins Tic-Tac-Toe in the Big Well wins the whole game.



PlayerB has a trick. PlayerB allows PlayerR to have chance of green O.

PlayerR put an O in the green position, occupying the big square. PlayerB has a free chance to put a green X.

PlayerB has 3 big square in a row in Big Well. So PlayerB wins the whole game.

Figure 3. Blue player wins a Tic-Tac-Toe in Big Well, and the whole game.

• Board game algorithm

The board games are always a research hot spot in Artificial Intelligence (AI). Since Deep Blue [1] beat Master Kasparov in chess, scientists are exploring new optimization method to conquer other board games. Recently, AlphaGo Zero [2] proves the algorithm can find a powerful strategy to beat the human champion, even in the board game Go, whose state space is more than 10^170 [3]. The most surprising thing about this algorithm is the training materials is irrelevant to the human knowledge. Traditional board game algorithm will learn to mimic the best moves of human players. However, AlphaGo Zero abandon the human knowledge domain, choose to learn entirely from itself. The Monte Carlo Tree Search is used to generate two players playing against themselves, or the training dataset [4]. It is a useful way to train the board game AI. The algorithm selects the move with a higher possibility of winning to play, records those winning moves to train the deep neural network. The researchers repeat those steps for countless times and finally have the most advanced version of board game AI.

• ELO rating system

ELO rating system [5] is designed by Professor Arpad Elo. It is one of the most general rating systems for player versus player games, including board games, sports, and online games. The main idea is to calculate an expectation of winning based on ranks of both players. By comparing the real results (win, lose or draw) and expectation, the new scores for both players are computed. For instance, a higher-ranking player will earn fewer points while winning a lower-ranking player, but lose more points while losing to a lower player. This system can estimate a nearly actual level of a player.

2. METHODS

• Monte Carlo Tree Search

The algorithm for TEN in this project will train itself by selecting the next move with a higher possibility. At each training iteration, the algorithm will simulate two players playing against each other. The algorithm will input the current chessboard state into the neural network, get a map of winning possibility. The move with top x% winning rate will be chosen to be played as the next move. X will be 50 at the beginning to improve the strategy variety. After a period, x will be 30 to accelerate the training progress. After one game is over, the move sequence of the winner will be recorded as training samples to train the neural network.

• Training the neural network

The neural network will accept the specific chessboard states as features, and the winning moves as labels. The neural network is multi-layer perceptron. There are nine hidden layers, one of which has ninety nodes. The activation functions of all layers are ReLU. The input features is a 10*9 array. The first 9*9 array is the chessboard state. The last 1*9 array is the big picture of the chessboard. The output of the neural network is a 9*9 array. Each element in the output array is the winning possibility of the corresponding position.

- **Criterion**

Because it is not the traditional optimization problem, it is impossible to use general loss function such as square error or hinge error. The criterion here is a simulation of a tournament. Ten players will start to compete with each other. One of them will decide the next move using the neural network. Others will play entirely random. They will begin with 1500 rating points. The points will be updated according to the result of a game. After 1000 games with random two players, the rating points of the neural network will be the criterion of the efficiency of the learning.

3. RESULTS

First, the game of all random player starts. The points distributes as the following:

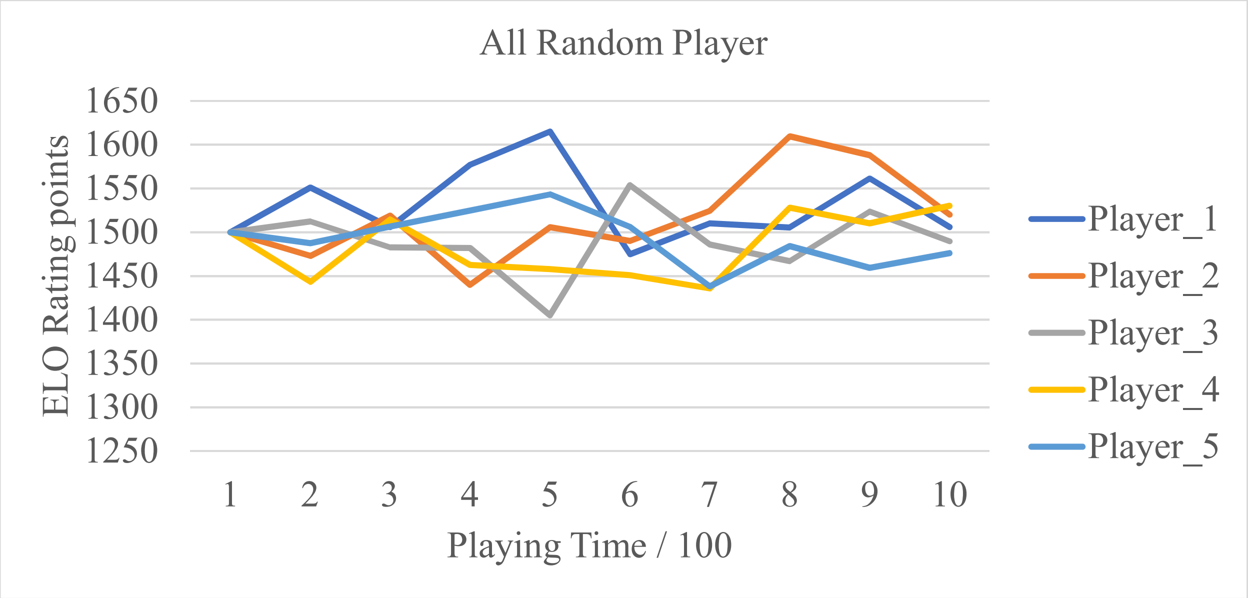


Figure 4. The rating distribution of all random player.

Then, the NN player takes part in the game.

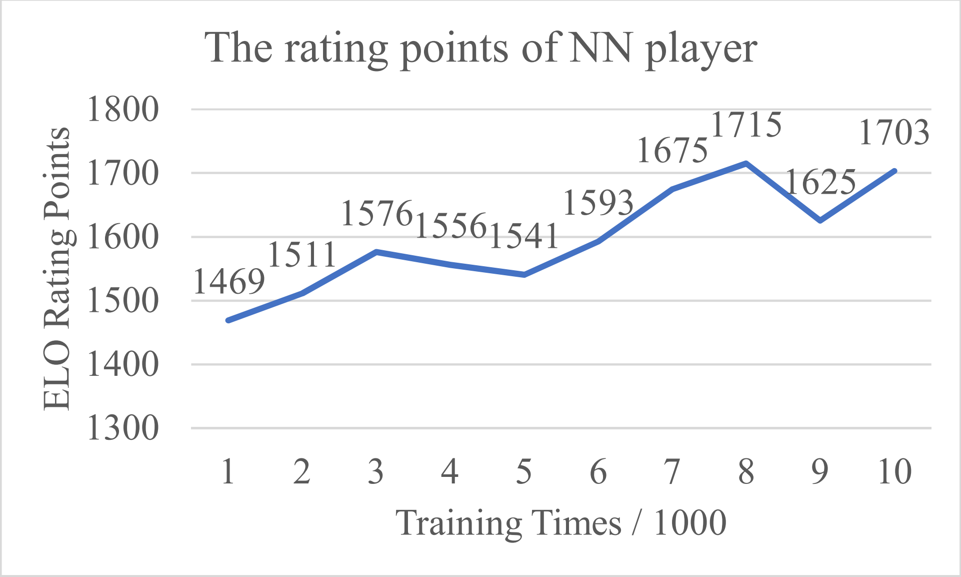


Figure 5. The rating distribution of NN player.

From that, the training does make a difference.

4. CONCLUSION

By inputting the dataset generated by Monte Carlo Tree Search, the algorithm can automatically improve its board game skill. Because the architecture of the neural network is simple, the training efficiency is not so good. If the further version is developed, I believe the better architecture can lead to the better performance.

5. REFERENCE

[1]. Campbell, M., Hoane, A. J., & Hsu, F. H. (2002). Deep blue. Artificial intelligence, 134(1-2), 57-83.

[2]. Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., ... & Chen, Y. (2017). Mastering the game of go without human knowledge. Nature, 550(7676), 354-359.

[3]. Tromp, J., & Farnebäck, G. (2006, May). Combinatorics of go. In International Conference on Computers and Games (pp. 84-99). Springer, Berlin, Heidelberg.

[4]. Coulom, R. (2006, May). Efficient selectivity and backup operators in Monte-Carlo tree search. In *International conference on computers and games* (pp. 72-83). Springer, Berlin, Heidelberg.

[5]. Elo, A. E. (1978). *The rating of chessplayers, past and present*. Arco Pub..