On Policy Gradient Applied to Chinese Standard Mahjong

Lan-Zhou Zheng, Jing Sun, Jing-Hui Bi, He-Xiang Zhang

WeiZhiYu Technology co., ltd.
zhenglz@smzy.cc

Abstract

In recent years, neural networks have gained a rapid growth, with spectacular breakthroughs in fields like computer vision, speech recognition and natural language processing. By means of neural networks, algorithms based on reinforcement learning methodology also got remarkable achievements in various domains, such as perfect-information game and imperfect-information game. Chinese standard mahjong is a multi-player imperfect-information game which originated from China and popularized worldwide. The large amount of hidden information of mahjong makes it a great challenge for designing an intelligent agent for mahjong, attracting many researchers working on it. Based on deep reinforcement learning method, we developed a mahjong intelligent agent, which got a good performance in IJCAI's Mahjong competition.

1 Introduction

There are already many researches on game Artificial Intelligence (AI). A milestone of game AI is the Alpha Go series from DeepMind team. In the year of 2017, Alpha Go Master defeated the human world champion by 3-0 on aggregate. The principle behind it is mainly the neural networks together with Monte Carlo Tree Search [Silver et al., 2016], while in this stage the AI still needed a lot of human experience. Later, Alpha Go Zero [Silver et al., 2017] was released, starting with no knowledge, it studied and trained all by itself and finally beat its predecessor Alpha Go. The performance boost mainly comes from the reinforcement learning (RL). Go is a perfect-information game, means that each player involved in the game get the very same information. On the contrary, imperfect-information game is the game that players have their own private information. A good example of the later one is mahjong. Mahjong is a 4-player participated game, each player has up to 14 private cards and the opponents have no idea about which cards are them. When a player makes a decision, like discarding a card, he should consider all the possible outcomes afterwards based on his assumption over the opponents' strategies. In early stage, a Nash equilibrium [Nash et al., 1950] is tried to be solved in order to find an optimal strategy over imperfect information game. Nash equilibrium is applied well to no-limit Texas Hold'em [Brown et al., 2017], which is a two-payer unlimited form of poker game. While in Mahjong, players and cards are getting more, making hidden information and explore space too large and too difficult to solve. Recent years, benefited from the growth of deep leaning, mahjong models generated by neural networks are getting better and better performance. In 2020, Microsoft Asia team published Suphx, a Japanese mahjong AI, could beat most top human players and is rated above 99.99% of all the officially ranked human players in the Tenhou platform [Li, 2020]. The great breakthrough made by Suphx is also attribute to reinforcement learning method.

Reinforcement learning is different from supervised learning nor unsupervised learning. The core idea of reinforcement learning methodology is to attain a goal, which usually is the maximum of the accelerated rewards feedbacked by the environment over a period of interaction [Sutton, 1998]. Incorporated with deep neural networks, which also called deep reinforcement learning got widely used in designing game AI such as Go, Atari video games [Mnih et al., 2013], Starcraft II [Vinyals, 2019] and Dota-2 [Berner, 2019].

We propose a Chinese Standard Mahjong AI based on Policy Gradient algorithm, which is a specific method of deep reinforcement learning. The process mainly includes two stages. First, we trained a supervised learning model based on convolutional neural networks using human players' labeled data. Second, we optimized the trained model by policy gradient method. The later shared the same constructure with the former, therefore the RL learning is actually a fine-tuned process. A significant advantage of using a trained model for RL instead from a blank state is that the time consumed to reach convergence is way shorter. The supervised model finally got an accuracy of over 75% in our test set and the RL trained model get even better overall performance in game test.

This paper is organized as follows. Section 2 explains a brief rules and terms of Chinese Standard Mahjong. Section 3 introduces our work in detail, which includes the data processing module, supervised training module and RL training module. In Section 4, the results of our experiments and the evaluation of our model are given and finally section 5 makes a conclusion.

2 Brief rules of Chinese Standard Mahjong

There are many variations of Mahjong in China, usually distinguished in types of cards and the actions player can do with a card. The rules of Chinese Standard Mahjong were formulated by the General Administration of Sport of China in 1998, and have become one of competitive sports through the whole country. Chinese Standard Mahjong has 144 cards with 36 types and usually each type has 4 same cards. 36 types could further divide into 3classes, which are number cards include 27 types, character cards include 7 type and flower cards include 2. The details are showed in table 1.

Class	Type
Number	1W…9W, 1T…9T, 1B…9B
Character	East, West, South, North, Middle, Rich, White
Flower	Season, Flower

Table 1: Card types and its classes

Different classes of cards have different operations which are demonstrated as following.

Chow operation: On number cards. When the player of your last seat discards a number card and you have got two cards which together with the discarded card they could combine into successive three cards. You can pick up this card and the three cards become a meld which should be put aside to show the others.

Peng operation: On number and character cards. When the other player discards a card and you have got two of the same type. Peng meld is also should be shown to the others.

Public Kong operation: On number and character cards. When the other player discards a card and you have got three of the same type. Meld should be put aside to show others.

Private Kong operation: On number and character cards. When you draw a card and you have got three of the same type. Melds should be put aside but could keep secret.

Supplementary Kong operation: On number and character cards. When you draw a card and you have got a Peng meld of the same type. The drawn card should be put with the Peng meld to become a Kong meld.

Open flower: On flower card. When you draw a flower card, you put it aside to show and draw another card immediately until a non-flower card.

Win operation: When you draw a card or pick up a card, the cards in your hand satisfy some certain rules. A group of 3 or 4 same cards is called a Flush, a group of three successive cards is called a Straight and a pair of same cards is called a Pair. The win hand usually requires 4 either Flush or Straight and a Pair.

Initially, each of 4 players is dealt with 13 cards. Randomly decided, there is a banker who starts to draw a card and then makes a decision. On drawing a card, there are 4 kinds of operations—declaring a win, making a private Kong meld, making a supplementary Kong and otherwise discarding a card. After making a meld, the player should draw a card immediately until he declares win or discards a card. On a player discarding a card, other players can pick up this card to make a Peng, chow or public Kong meld or declare win. If no player picks up the discarded card, the next player should draw a card. Game ends when anyone declares a win or no cards left in the wall.

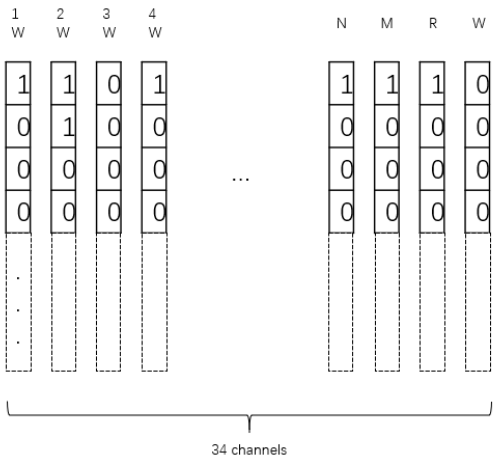
We designed 3 neural network models together with some logic judgment to integrate a Chinese Standard Mahjong AI. A Discard model in charge of discarding a card, a Peng model and a Chow model decides whether to make a Peng or Chow meld respectively.

3 Model architecture

3.1 Data processing

We did much work on exploring an optimal input structure to neural networks. A series of features are abstracted from human game records and then mapped to matrix according to one-hot encoding. Since flower cards don't contribute for forming a win hands and it is not allowed to be discarded, so we omit flower cards' positions in input structures. The basic unit of the input is a vector with 34 dimensions, each dimension stands for a type of card except the flower cards. For cards information such as a player's private cards, we use 4 such unit which stands for the 4 cards of each type. According to one hot encoding, the 4 by 34 matrix has the ability to represents the whole cards in Mahjong. In addition to the card information, we also calculate some extra information to enhance the input. For example, a best division of a player's current cards is that a combination of groups which is the most similar with a win hand. We add the best division to the

Figure 1: Input structure



input to emphasis the combination feature. A state of Ting is that when a player is about to win just need to get another one card. We also add this Ting state information into the input, using one unit with whole 1 or zero to respect if the player in Ting state. For Chow and Peng model, we add another vector to indicate the card discarded. The experiment results showed that the stacked information helped to improve the performance of the neural network model.

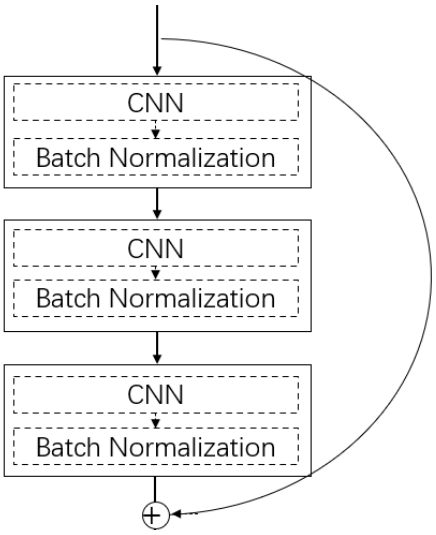
When input to the neural networks, we make the 34 as the channel dimension, which got better performance compared to single channel. The final input shape is like Figure 1.

3.2 Supervised learning model

In general, the Discard operation is the most important operation in wining a Mahjong game. For designing an AI based on neural networks, we abstract the discard operation to a classification problem. Since there are totally 34 types of cards could be discarded, so the discarding choice must be among the 34 types, which makes it a 34-classes problem with each class represents a certain type of card. Similarly, the Peng and Chow operation are also taken as classification problem and share the same neural network architecture. The only difference of Peng/Chow from Discard is the number of output dimensions, which is 2, meaning do the Peng/Chow operation or not. As to various Kong and Win operation, we use logic to judge whether to make a Kong/Win or not.

The model we used is a CNN based architecture. Each CNN followed by batch normalization [Ioffe, 2015] is taken as a layer. Every 3 or 5 such layers together with a residual connection [He et al., 2016] make up a block, blocks then are stacked to form the final model. The details of a block is shown in figure 2.

Figure 2: A CNN block



3.3 Reinforcement learning model

After much works on supervised learning method, we found it hard to further improve the performance of the Discard model, while Peng/Chow models have achieved satisfied performance. Therefore, we turned to reinforcement learning method in order to get a better Discard model.

In reinforcement learning terms, the AI is called an agent, which takes an action through its policy on the current state of environment. Based on the action the agent made, the environment gives a reward and transfers to next state according to some certain rules. The constant interactions between the environment and the agent until the game ends generates a trajectory, which we denote as $\tau = \{s_1, a_1, r_1, s_2, a_2, r_2 \ \dots s_T, a_T, r_T, END\}$. Here the subscript means the time step from 1 to T, s stands for the state, a stands for the action and r for reward. The probability of a trajectory occurs is:

$$p_\theta(\tau) = p(s_1) \prod_{t=1}^{T} p_\theta\,(a_t|s_t) p(s_{t+1}|s_t, a_t) \quad (1)$$

The accumulated reward the trajectory τ get is the summation of the rewards the agent gets. Therefore, the whole rewards the agent get through several trajectories is:

$$\overline{R_\theta} = \sum_\tau R(\tau) p_\theta(\tau) = E_{\tau \sim p_\theta}[R(\tau)] \quad (2)$$

The agent is supposed to adjust its policy to maximize the expectation of R_θ, which is the core idea behind Policy Gradient. The policy is implemented by neural networks, which is exactly the same as the supervised architecture. We take the R_θ as the target function, and the maximum is found by gradient decent (take the opposite of R_θ) given by following formula:

$$\nabla \overline{R_\theta} = \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} R(\tau^n) \nabla \log\,(p_\theta)(a_t^n|s_t^n) \quad (3)$$

The training process of reinforcement learning is based on the supervised trained parameters. This process saved calculating time to a large extent but fine-tuned the parameters to get more rewards, which theoretically improve the discarding performance.

We carefully designed several patterns to calculate the reward the environment gives back to the agent, which is vital to the convergence of the training process. We introduce Win Distance into the reward calculation. Win distance is defined as the number of cards in lack to reach a win hands. For example, a player got [1T, 1T, 2T, 2T, 3T, 3T, 7T, 7T, 7T, E, W, N, N, S] in hands, and the most similar win hands template is [1T, 1T, 2T, 2T, 3T, 3T, 7T, 7T, 7T, W, W, N, N, N]. In this case the win distance is 2. When a player discards a card, he may discard a card that make the win distance smaller or otherwise the bigger. If the discarding operation shrinks the distance, the environment would give a positive value back to the agent; while on the contrary, if the discarding makes the distance even far away, a negative value would be output. In this way, agent interacts with the environment and generates trajectories, which then fed into neural networks to adjust its parameters.

In practice, a penalty term which is the entropy of the model, is added to the loss function to encourage the networks to explore more actions.

$$\nabla J(\theta) = E_{\tau \sim p_\theta}[R(\tau)\nabla \log(p_\theta)] + \alpha \mathrm{H}(\theta) \quad (4)$$

Here $\mathrm{H}(\theta)$ is the entropy of the model and $\alpha > 0$ is a trade-off coefficient.

4 Experiments results

4.1 Supervised learning

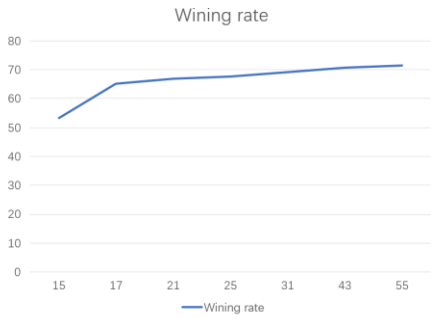
The data we use for training the supervised models are generated from online human player games. The size of the dataset is shown in table 2. For each dataset, we took 90% as train dataset and the other as test dataset. The final accuracy on each dataset is also given in table 2.

Dataset	Size	Accuracy
Discard	9M	71.5%
Peng	2M	92.4%
Chow	2M	90.7%

Table 2: The size of datasets

We also build a game platform for our AIs to game with each other in order to look into the actual performance of the integrated AI and here the wining rate is an important index of its performance. We found that stacking more information on the input is an efficient way to improve the wining rate. From 15 stacked units (mentioned in section 3.1) to finally 55 stacked units in Discard model, the improvement of the accuracy is as figure 3 show. Note that the 17th unit got a significant improvement which is the unit indicated the Ting state.

Figure 2: The wining rate as input information stacks on



4.2 Reinforcement learning

In reinforcement learning stage, we got trajectories through self-play, which means the very same AI is joined as 4 players in a game. The trajectories are fed into the neural networks then to apply policy gradient to get the optimal parameters.

The reward setting of the environment feeds back is of great concern. In the beginning, we use the final score of a game according to Mahjong rules. However, the model doesn't converge in such way. It is assumed that the final score of Chinese Mahjong is usually of large variance and the intermediate score during the game is not monotonous but a fluctuating change. Therefore, it is not proper for reinforcement learning to modulate its parameters according to the final game score. The win distance introduced in section 3.3 is a good tool for defining a reward from environment, in which includes a global information of whether to win a game and is much stable compared to the final score.

Figure 3 shows the wining probability of RL models over its

Figure 3: The change of wining probability of RL model during RL training.

baseline in game test as RL training steps goes on. In the final actual game test, the RL-trained model wins 75% over 100000 games. Note that the RL AI and baseline AI share the same Peng/Chow models while the Discard model is different.

5 Conclusion and Discussion

In this paper, we proposed a reinforcement learning method for designing a Chinese Standard Mahjong AI. The training process mainly includes two stages, the supervised learning stage and reinforcement learning stage. Learning blankly from reinforcement learning is feasible but could cost a lot of resources and time. However, using human data to train a supervised model and then tuning the parameters by reinforcement learning method could save time on order of magnitudes. The results showed that this two-stage method could get good performance.

There are still many aspects to improve in this project. We used Policy Gradient, which is a basic reinforcement learning method. Further improved RL methods such like Actor-Critic, Advantage Actor-Critic may get better performance. The structure of the neural networks could be designed more fitted to the sparse input. These aspects will be explored more in our future work.

References

[Silver et al., 2016] Silver, D. , Huang, A. , Maddison, C. J. , Guez, A. , Sifre, L. , & Driessche, G. V. D. , et al. (0). Mastering the game of go with deep neural networks and tree search. *Nature*.

[Silver et al., 2017] Silver D , Schrittwieser J , Simonyan K , et al. Mastering the game of Go without human knowledge[J]. Nature, 2017, 550(7676):354-359.

[Nash et al., 1950] Nash J F . Equilibrium Points in N-Person Games[J]. Proceedings of the National Academy of ences, 1950, 36(1):48-49.

[Brown et al., 2017] Brown N, Sandholm T . Safe and Nested Subgame Solving for Imperfect-Information Games[J]. 2017.

[Sandholm, 2018] Sandholm T . Depth-Limited Solving for Imperfect-Information Games[J]. ence, 2018, 347(6218):122-3.

[Li, 2020] J Li, S Koyamada, Q Ye, G Liu, HW Hon. Suphx: Mastering Mahjong with Deep Reinforcement Learning.2020

[Sutton, 1998] Sutton R S , Barto A G . Reinforcement Learning[J]. A Bradford Book, 1998, volume 15(7):665-685.

[Mnih et al., 2013] Volodymyr M, K, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller. Playing Atari with Deep Reinforcement Learning.

[Vinyals, 2019] Oriol Vinyals, I. Babuschkin. Grandmaster level in StarCraft II using multi-agent reinforcement learning[J]. Nature.

[Berner, 2019] C Berner, G Brockman, B Chan, V Cheung, S Zhang. Dota 2 with Large Scale Deep Reinforcement Learning

[Ioffe, 2015] S Ioffe, C Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.

[He et al., 2016] He K , Zhang X , Ren S , et al. Deep Residual Learning for Image Recognition[J]. 2016.