The Report of Team ALONG for IJCAI 2020 Mahjong AI Competition

Zhongyue Huang, **Minggao Wei**, **Guan Yang**
NetEase Game AI
{huangzhongyue, weiminggao, yangguan}@corp.netease.com

Abstract

Recently years, Artificial Intelligence (AI) based on deep-learning methods has achieved human-level performance in many challenging tasks such as Go, chess, and more complex environment StatCraft. Chinese Standard Mahjong has always been a popular four-player imperfect-information game but very challenging for AI researchers due to its complex playing/scoring rules and rich hidden information. This study utilizes two newly learning-techniques including imitation learning (IL) and reinforcement learning (RL) to solve the complex problem. At imitation learning stage, expert dataset is applied to train a ResNet structure model. Secondly, we take RL to strengthen the IL model according to large-scale sampling and training. Experiment shows that 1) imitation model exhibits the nature of gathering the winning's condition and then 2) RL method do significantly outperform the IL model, which can win more scores than IL model.

1 Introduction

Chinese Standard Mahjong is a four-player game with imperfect information. There are 144 tiles, including 36 characters, 36 dots, 36 bamboos, 16 winds, 12 dragons, and 8 flowers which lead to a large number of 14-tile combinations. A winning hand of 14 tiles usually contains four melds and one pair. There are 45 kinds of sequences, 34 kinds of triplets, and 34 kinds of pairs, in addition to some special winning tiles pattern, such as thirteen orphans, which result in various winning hands and different winning scores. In Chinese Standard Mahjong, a winning hand must have at least 8 points. It would be judged as falsely winning declaring if a player whose score of hand less than eight fans claiming a tile to win. It would be judged as a draw when all tiles are drawn and each play's score of the hand is less than eight fans. There are 81 types of fan with different points, including specific combinations of tiles and special winning approaches such as self-draw. Huge amounts of tiles combination plus the limitation of winning score make the decision process especially complicated.

IJCAI 2020 Mahjong AI Competition apply Mahjong Competition Rules formulated by All-China Sports Federation in 1998. To reduce randomness, the competition adapts Swiss tournament and duplicate format. The four players that are matched together will be randomly assigned to four decks of tile walls, each of which is played 24 times (in full permutation). One deck corresponds to a ranking score, ranged from 1 to 4. The player gets four ranking scores and one subpoint per match. To make a rank, all ranking scores are added to accumulated subpoints which is divided by a large number. In addition to the way of scoring, the wall has some changes. Eight flowers are removed, and the remaining 136 tiles are randomly divided into 4 sub-walls. The player can only draw tiles from his own sub-wall.

2 Methods

It's extremely difficult to train a model using reinforcement learning from the beginning owling to critical conditions of declaring winning and high risk of draw. Human's experience was used first to train a supervised model. After that, the supervised model was used to further improve reinforcement learning model.

2.1 Features

Chinese Standard Mahjong has 42 unique tiles including 8 flowers. As duplicate format is used in the competition, 34 tiles remain and 8 flowers are removed. We use 593 $\times$ 34 feature map to represent a state. There are three types of features: base features, look-ahead features, and avaliable-action features.

base features. The directly observable information includes private hand, Concealed Kong, open melds, discarded tiles, taken tiles, prevailing wind, dealer's wind, and the number of tiles left in each wall. In addition to that, we simply calculate some features, such as not-seen tiles and all tiles on hand.

look-ahead features. Chinese Standard Mahjong has 81 types of Fan with different points. Fan relating to ways of winning declaration cannot be processed into features. To compensate, distances between tiles in current hand and tiles distribution of all fans is computed. The distance from a fan is defined as the least number of tiles not satisfying certain this fan pattern. For example, a special fan named 'SanSeSanBuGao' consists of 30 different tile patterns. The num-

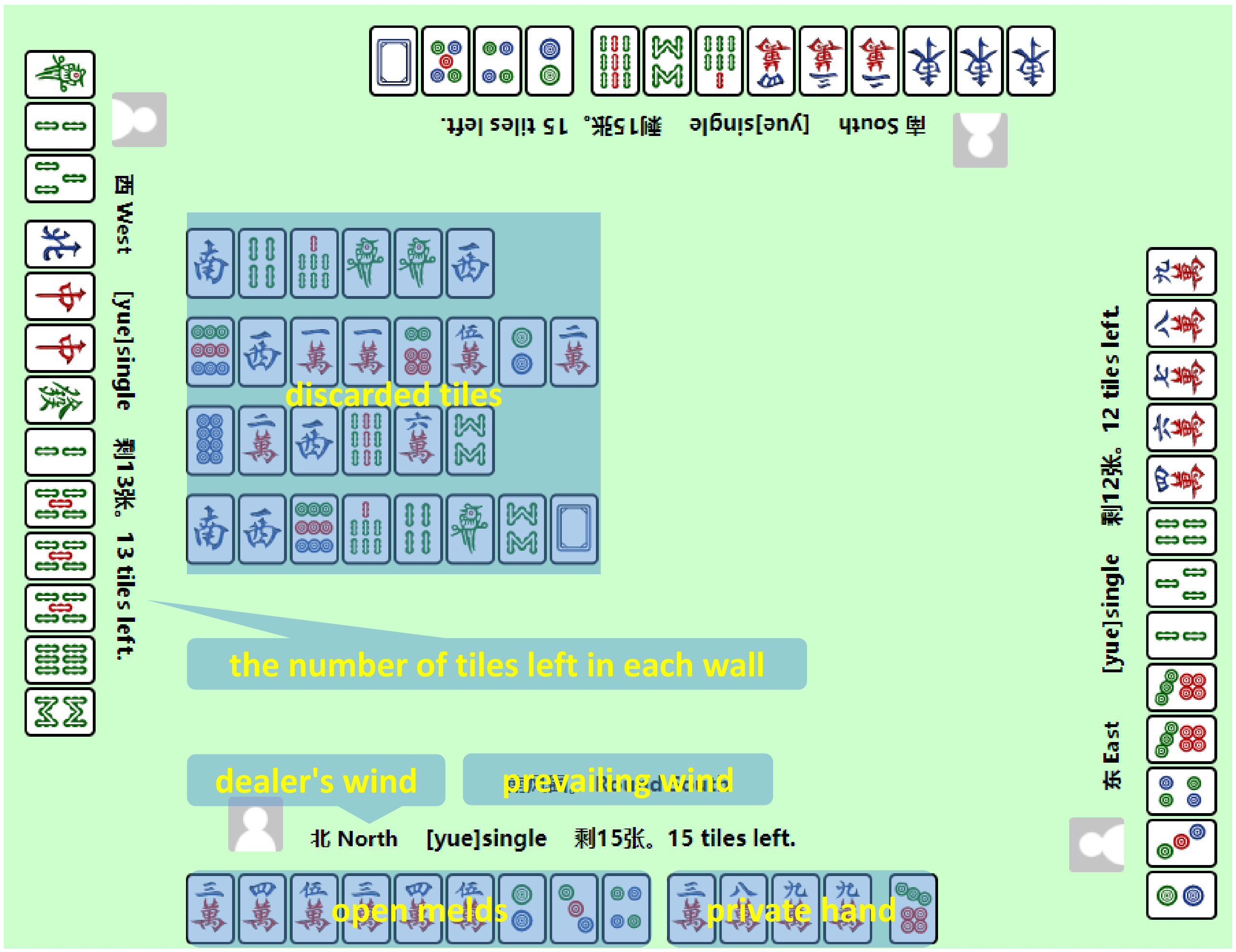
南 South [yue]single 剩15张。15 tiles left.
西 West [yue]single 剩13张。13 tiles left.
东 East [yue]single 剩12张。12 tiles left
北 North [yue]single 剩15张。15 tiles left.
discarded tiles
the number of tiles left in each wall
dealer's wind
prevailing wind
open melds
private hand

Figure 1: Base features.

ber of tiles in current hand differs from each tile pattern is computed, and then the least num is used.

available-action features. We use four channels to indicate current available actions including discarding, Chow, Pong and Kong. For the three actions of Chow, Pong, and Exposed Kong, a channel is used to represent the current tile discarded by others. In addition, there is an extra channel used to indicate tiles in private hand that are related to Chow.

2.2 Model

We use one model that consists 50 ResNet blocks to predict all actions. The model has 131 outputs, of which 34 represents discarding 34 unique tiles, 45 indicates Chow 45 sequences of tiles, 1 means whether or not to Pong current tile, 34 represents Kong 34 different tiles, and 1 is passing.

2.3 Imitation Learning

We use behaviour clone to imitate human policy with mahjong data log provided by competition organizer. Behaviour clone is a method that construct a dataset (s_t, a_t) where s_t is features and a_t is labels in the semantics of supervised learning. We do supervised learning on the dataset minimizing the cross entropy loss resulting in average human policy model π_h with parameter θ_h.

Dim	34	45	1	34	1
Action	Discard	Chow	Pong	Kong	Pass

Table 1: The 131 outputs of the model.

2.4 Reinforcement Learning

Optimization Method. Reinforcement learning has been successfully applied in many different scenarios such as Atari games [Mnih *et al.*2015], Go [Silver *et al.*2018] and continuous control [Lillicrap *et al.*2016]. The training is based on Proximal Policy Optimization [Schulman *et al.*2017]. We use

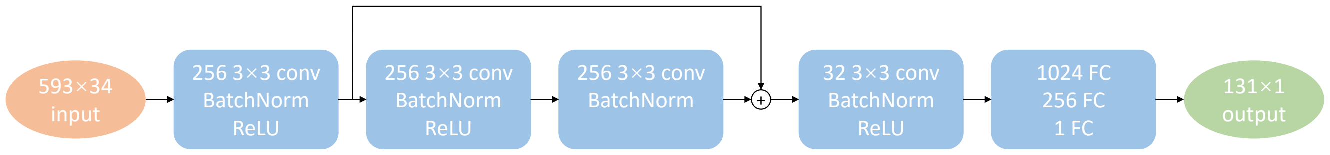

Figure 2: The structure of the model.

PPO-Clip which the target of optimization is

$$\arg\max_{\theta} min(r_t(\theta)\hat{A}_t, clip(r_t(\theta)\hat{A}_t, 1-\epsilon, 1+\epsilon)) \quad (1)$$

where θ is the parameters of model and $r = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ is the probability ratio where $r(\theta_{old}) = 1$. A_t is the advantages defined as $A_t = R_t - V_t$ measures how better the current action than average.

In order to stabilize the training process, we modified the target of optimization as dual PPO [Ye *et al.*2020]. The final optimization target is

$$\arg\max_{\theta} max(min(r_t(\theta)\hat{A}_t, clip(r_t(\theta)\hat{A}_t, 1-\epsilon, 1+\epsilon)), c\hat{A}_t) \quad (2)$$

where $c > 1$ is hyperparameter that controls the optimization bound.

We initialize the model parameters with average human policy model parameters θ_h. During training phase, the training agent plays 128-1024 games against past models parallelLy and generating training samples to update policy with Equation (2).

Reward. Due to the sparse reward and large state space it's hard for rl agent to learn better policy even with imitated human policy model. Besides final score as the final reward, we add extra reward when agent reach shanten state to guide the rl agent learn better policy.

3 Experiments

3.1 Dataset

We use the human match dataset provided by the organizer. There are about half a million games, a total of 32 million steps. We filter some data belonging to people who have made low-level mistakes, such as false winNing. In addition, we also try to extract data from winners to further improve data quality.

3.2 Supervised Learning

Table 2 shows our training results on different datasets. It can be seen that the larger the batch size, the higher the accuracy. The winner dataset has a high accuracy rate on the same dataset, but the accuracy rate on all datasets is low. The last column is the scores of the two models against each other trained on the same batch size and different datasets. When batch size is small, the filtered winner dataset has advantages, but when batch size becomes larger, the all dataset with a large amount of data gets a positive score.

Dataset	Batch Size	Acc	Winner Acc	Score
all	128	71.96	-	-0.31
	512	73.21	-	+0.13
	2048	74.00	-	+0.25
winner	128	70.90	74.58	+0.31
	512	71.33	76.82	-0.13
	2048	72.75	78.68	-0.25

Table 2: The comparison of all dataset and winner dataset.

agent	1000	10000	50000	100000
sl	+0.33	-0.15	-0.27	-0.21
rl	-0.33	+0.15	+0.27	+0.21

Table 3: The comparison of sl agent and rl agent.

3.3 Reinforcement Learning

We apply reinforcement learning to the model of batch size 2048 trained on all datasets. As shown in the table 3, we match two sl models and two rl models to play against each other. In the beginning, the scores fluctuate, due to the randomness of Mahjong. But as the number of games increases, the score gradually stabilizes. Finally, rl agent is about 0.2 points higher than the sl agent.

4 Conclusion

We combine IL and RL to train a powerful mahjong AI. By playing with others in the botzone, we found that the agent can make reasonable actions. There is a lot of space for optimization due to the complexity and randomness of Mahjong.

References

[Lillicrap *et al.*, 2016] Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *ICLR (Poster)*, 2016.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal

Policy Optimization Algorithms. *arXiv:1707.06347 [cs]*, July 2017. arXiv: 1707.06347.

[Silver *et al.*, 2018] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419):1140–1144, December 2018.

[Ye *et al.*, 2020] Deheng Ye, Zhao Liu, Mingfei Sun, Bei Shi, Peilin Zhao, Hao Wu, Hongsheng Yu, Shaojie Yang, Xipeng Wu, Qingwei Guo, Qiaobo Chen, Yinyuting Yin, Hao Zhang, Tengfei Shi, Liang Wang, Qiang Fu, Wei Yang, and Lanxiao Huang. Mastering Complex Control in MOBA Games with Deep Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):6672–6679, April 2020.