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 bamboo, 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 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.

2 Methods
It’s extremely difficult to train a model using reinforcement learning from the beginning owing 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 available-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 ‘SanSanBuGao’ consists of 30 different tile patterns. The num-
The number of tiles left in each wall

**Figure 1:** Base features.

<table>
<thead>
<tr>
<th>Dim</th>
<th>34</th>
<th>45</th>
<th>1</th>
<th>34</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>Discard</td>
<td>Chow</td>
<td>Pong</td>
<td>Kong</td>
<td>Pass</td>
</tr>
</tbody>
</table>

Table 1: The 131 outputs of the model.

...
PPO-Clip which the target of optimization is

$$\arg \max_{\theta} \min(r_{t}(\theta)A_{t}, \text{clip}(r_{t}(\theta)A_{t}, 1 - \epsilon, 1 + \epsilon))$$  \hspace{1cm} (1)$$

where $\theta$ is the parameters of model and $r = \frac{\pi_{\theta}(a_{t}|s_{t})}{\pi_{\text{old}}(a_{t}|s_{t})}$ is the probability ratio where $r(\theta_{\text{old}}) = 1$. $A_{t}$ 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

$$\arg \max_{\theta} \max(min(r_{t}(\theta)A_{t}, \text{clip}(r_{t}(\theta)A_{t}, 1 - \epsilon, 1 + \epsilon)), cA_{t})$$  \hspace{1cm} (2)$$

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

We initialize the model parameters with average human policy model parameters $\theta_{h}$. 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.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Batch Size</th>
<th>Acc</th>
<th>Winner Acc</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>128</td>
<td>71.96</td>
<td>-</td>
<td>-0.31</td>
</tr>
<tr>
<td></td>
<td>512</td>
<td>73.21</td>
<td>-</td>
<td>+0.13</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>74.00</td>
<td>-</td>
<td>+0.25</td>
</tr>
<tr>
<td>winner</td>
<td>128</td>
<td>70.90</td>
<td>74.58</td>
<td>+0.31</td>
</tr>
<tr>
<td></td>
<td>512</td>
<td>71.33</td>
<td>76.82</td>
<td>-0.13</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>72.75</td>
<td>78.68</td>
<td>-0.25</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>agent</th>
<th>1000</th>
<th>10000</th>
<th>50000</th>
<th>100000</th>
</tr>
</thead>
<tbody>
<tr>
<td>sl</td>
<td>+0.33</td>
<td>-0.15</td>
<td>-0.27</td>
<td>-0.21</td>
</tr>
<tr>
<td>rl</td>
<td>-0.33</td>
<td>+0.15</td>
<td>+0.27</td>
<td>+0.21</td>
</tr>
</tbody>
</table>

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


**Figure 2:** The structure of the model.
