Building a Computer Mahjong Player Based on Monte Carlo Simulation and Opponent Models

Naoki Mizukami\textsuperscript{1} and Yoshimasa Tsuruoka\textsuperscript{1}

\textsuperscript{1} The University of Tokyo
Introduction

• Imperfect information games are challenging research
  • Contract bridge [Ginsberg 2001]
  • Skat [Buro et al 2009]
  • Texas Hold’em [Bowling et al 2015]

• We focus on Japanese Mahjong
  • Multiplayer
  • Imperfect information
  • Enormous number of information sets
    • Mahjong: $10^{60}$
    • Texas Hold’em: $10^{18}$
Related work

• Computer poker
  • Nash equilibrium strategy
    • CFR+ method has solved Heads-up limit hold’ em poker [Bowling et al 2015]
  • Opponent modeling
    • Opponent modeling and Monte Carlo tree search for exploitation [Van der Kleij 2010]
    • The program updates a hand rank distribution in the current game state when the showdown occurs [Aaron 2002]
Japanese Mahjong

• Rules
  • It play with four players
  • A player can win round by completing a winning hand consisting of 13 tiles
  • One game of mahjong consists of 4 or 8 rounds

• Terms
  • *Waiting*
    • A player’s hand needs only one tile to win
  • *Folding*
    • A player gives up to win and only tries to avoid discarding a winning tile for opponents
    • Is not action but strategy
One-player mahjong [Mizukami et al 2014]

• Implement folding system
• One-player Mahjong
  • A One-player Mahjong player only tries to win
  • It is trained by supervised learning using game records
  • It plays an important role in our Monte Carlo simulation
• Recognizing Folding situations
  • Folding system is realized by supervised learning
  • Positions in game records are annotated manually
• Result: Beyond average human players
• Problem: It is difficult to annotate required data
Proposed method

• Overview
  - Opponent modeling by supervised learning
  - Original game
  - Monte Carlo Simulation
  - Decides moves
  - Abstracted game

• Advantage
  - It is not necessary to predict opponents’ specific hands
  - Can be trained models only using game records
Training setting

• Game records
  • Internet Mahjong site called "Tenhou"

• Dataset
  • Training data $1.7 \times 10^7$
  • Test data 100

• Models
  • Waiting: logistic regression model
  • Winning tile: logistic regression model
  • Hand score: Linear regression model
Waiting

- The model predicts whether an opponent is waiting or not

Input

Discarded tiles
Opponent’s hand
revealed melds

Label: waiting

Output  $P(\text{opponent} = \text{waiting}) = 0.8$
Evaluation and result

- Evaluation
  - Area Under the Curve

<table>
<thead>
<tr>
<th>Player</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert player</td>
<td>0.778</td>
</tr>
<tr>
<td>Prediction model</td>
<td>0.777</td>
</tr>
<tr>
<td>-Discarded tiles</td>
<td>0.772</td>
</tr>
<tr>
<td>-Number of revealed melds</td>
<td>0.770</td>
</tr>
</tbody>
</table>

- Same prediction ability as the expert player
- Expert player: Top 0.1% of the players
Winning tiles

• Model predicts opponents’ winning tiles
• In general, there are one or more winning tiles
  → Build prediction models for all kinds of tiles

Input

Discarded tiles

Opponent’s hand

revealed melds

Winning tile or

Output : 0.0 0.10 0.15
Evaluation method

1: Input opponents’ information  e.g winning tiles
2: Tiles that a player has are arranged in ascending order of probability of being a winning tile for opponent

Ranking about winning tiles for opponent

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>...</th>
<th>6</th>
<th>...</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>🤚</td>
<td>🤚</td>
<td>🤚</td>
<td>🤚</td>
<td>🤚</td>
<td>🤚</td>
</tr>
</tbody>
</table>

Evaluation value = 6 / (14-2) = 0.5
Result

- *Random*: Tiles are arranged randomly

<table>
<thead>
<tr>
<th>Player</th>
<th>Evaluation value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert player</td>
<td>0.744</td>
</tr>
<tr>
<td>Prediction model</td>
<td>0.676</td>
</tr>
<tr>
<td>Revealed melds</td>
<td>0.675</td>
</tr>
<tr>
<td>Discarded tiles</td>
<td>0.673</td>
</tr>
<tr>
<td><em>Random</em></td>
<td>0.502</td>
</tr>
</tbody>
</table>
Hand Score (HS)

• The model predicts the score that the player has to pay

Input

Discarded tiles

Opponent’s hand

revealed melds

Hand Score 2,600

Output 2,000
Evaluation method and result

• Evaluation method
  • Mean Squared Error (MSE)

<table>
<thead>
<tr>
<th>Player</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prediction model</td>
<td>0.37</td>
</tr>
<tr>
<td>-Revealed Melds</td>
<td>0.38</td>
</tr>
<tr>
<td>-Revealed fan value</td>
<td>0.38</td>
</tr>
<tr>
<td>Expert player</td>
<td>0.40</td>
</tr>
</tbody>
</table>

Performance of prediction model is **higher** than that of an expert player.
Overview of proposed method

\[
P(p = \text{waiting}) = \frac{1}{1 + \exp(-w^T x_p)}
\]

\[
P(\text{Tile} = \text{winning}) = \frac{1}{1 + \exp(-w^T x_p)}
\]

\[
HS = w^T x
\]
Application of opponent models

• Using three prediction models to estimate an expected value

• $LP$ (Losing probability)

  \[ LP(p, \text{Tile}) = P(p = \text{waiting}) \times P(\text{Tile} = \text{winning}) \]

• $EL$ (Expected Loss)

  \[ EL(p, \text{Tile}) = LP(p, \text{Tile}) \times HS(p, \text{Tile}) \]
Monte Carlo simulation

• The program calculates Score($Tile$) for each tile
  • Program selects the tile that has the highest Score($Tile$)

\[
Score(Tile) = sim(Tile) \times \prod_{p \in \text{opponents}} \left(1 - (LP(p, Tile))\right) - \sum_{p \in \text{opponents}} EL(p, Tile)
\]

• Procedure of $sim(Tile)$
  1: Discard a tile
  2: Opponent’s turn
  3: Program’s turn
  4: Repeat 2,3
  5: Get reward
Evaluation setting

• Compared to our previous work
  • Moves are computed in a second
  • Length of a game is four rounds

• VS state-of-the-art program
  • Mattari Mahjong
  • Duplicate mode
    • can generate same tile sequences
    • can compare the result

• VS human players
  • Internet Mahjong site ``Tenhou’’
## Result

- **VS Mattari Mahjong**

<table>
<thead>
<tr>
<th></th>
<th>1st (%)</th>
<th>2nd (%)</th>
<th>3rd (%)</th>
<th>4th (%)</th>
<th>Average rank</th>
<th>Games</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed method</td>
<td>25.2</td>
<td>25.6</td>
<td>24.7</td>
<td>24.5</td>
<td>2.48±0.07</td>
<td>1000</td>
</tr>
<tr>
<td>Mattari Mahjong</td>
<td>24.8</td>
<td>24.7</td>
<td>25.0</td>
<td>25.5</td>
<td>2.51±0.07</td>
<td>1000</td>
</tr>
<tr>
<td>[Mizukami+ 2014]</td>
<td>24.3</td>
<td>22.6</td>
<td>22.2</td>
<td>30.9</td>
<td>2.59±0.07</td>
<td>1000</td>
</tr>
</tbody>
</table>

- **VS Human players**

<table>
<thead>
<tr>
<th></th>
<th>1st (%)</th>
<th>2nd (%)</th>
<th>3rd (%)</th>
<th>4th (%)</th>
<th>Average rank</th>
<th>games</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed method</td>
<td>24.1</td>
<td>28.1</td>
<td>24.8</td>
<td>23.0</td>
<td>2.46±0.04</td>
<td>2634</td>
</tr>
<tr>
<td>[Mizukami + 2014]</td>
<td>25.3</td>
<td>24.8</td>
<td>25.1</td>
<td>24.8</td>
<td>2.49±0.07</td>
<td>1441</td>
</tr>
</tbody>
</table>
Conclusion and Future work

• Conclusion
  • Performance of the three prediction models is high
  • Our program outperforms state-of-the-art program by Monte Carlo simulation

• Future work
  • Consider final rank
  • Improve players’ actions in simulation
Training of 1-player mahjong players

- A weight vector is updated so that the player can make moves as expert players.
- We used the averaged perceptron.

$$W' = W + X - X$$

$x$: feature vector
$W$: weight vector
Recognizing folding situations

• We train a classifier for folding situations using a machine learning approach
• This approach requires training data.
  ➔ Positions in game records are annotated manually
Setting

• Dataset
  • Training data $1.77 \times 10^7$
  • Test data 100

• Features
  • Discarded tiles, number of revealed melds, and so on
  • 6,888 dimension

• logistic regression model

$$P(p = \text{waiting}) = \frac{1}{1 + \exp(-w^T x_p)}$$
Setting

• Dataset
  • Training data 1.77 × 10^7
  • Test data 100

• Features
  • Discarded tiles, number of revealed melds, and so on
  • 31,416 dimension

• logistic regression model

\[
P(Tile = \text{winning}) = \frac{1}{1 + \exp(-w^T x_p)}
\]
Setting

• Dataset
  • Training data $5.92 \times 10^7$
  • Test data 100

• Features
  • Revealed Melds, Revealed fan value and so on
  • 26,889 dimension

• Linear regression model

$$HS = w^T x$$
Evaluation and result

- Evaluation
  - Area Under the Curve

<table>
<thead>
<tr>
<th>Player</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expert player</td>
<td>0.778</td>
</tr>
<tr>
<td>Prediction model</td>
<td>0.777</td>
</tr>
<tr>
<td>-Discarded tiles</td>
<td>0.772</td>
</tr>
<tr>
<td>-Number of revealed melds</td>
<td>0.770</td>
</tr>
</tbody>
</table>

- Same prediction ability as the expert player
- Expert player: Top 0.1% of the players
Flowchart of program’s turn

Pick up a tile

Win check

YES  Win

Decide one-player mahjong moves

Discard a tile and compute \textit{ODEV}

Fold

YES  Win

Win check for opponent

YES  Win

Next player

- \textit{ODEV (One-Depth Expected Value)} is an expected value that is calculated by searching game trees until the program’s next turn.
- \textit{Fold}: a player picks up a tile and discards no tiles
Flowchart of opponent’s turn

- Opponent player has two binary parameters indicating whether he is waiting or folding.