Script for "How to Win a Chinese Chess Game"

Jinzhong Niu

April 25, 2004

1 Presentation

- Set Up

  Chinese chess is played on a 10×9 board, on which either player has 16 pieces. Pieces are located at the intersections of the grid instead of inside small blocks as in traditional chess. Totally 7 kinds of pieces are used: *General/King*, *Guard*, *Minister/Elephant*, *Rook*, *Knight*, *Cannon*, and *Pawn/Soldier*. The 4 blocks on either sides make the palace for the King.

- King

  A king may move one space at a time in either of the four directions: north, east, south, or west, but is not allowed to leave his palace.

  And the two kings cannot be on the same vertical line without any other piece between them.

- Guard

  A guard may move one space at a time diagonally, but same as the kings cannot leave the palace.

- Minister

  A minister can move two spaces at a time diagonally, but may not cross the river that separates the board into two halves.

  A minister is blocked and cannot move in the corresponding direction if another piece is directly adjacent to it on a diagonal path along which the minister can move without the blocking piece.

- Knight

  A knight can make 'L'-shaped moves by going one space vertically/horizontally AND two spaces horizontally/vertically. The possible moves are blocked if another piece is immediately next to the Knight in the direction in which the knight is to move two spaces.

- Cannon

  A cannon can move any number of spaces north, east, south, or west. And it can capture any opponent's piece in its path as long as there is exactly one piece of either player's in between.

- Pawn

  A pawn can move only one space at a time. If it is before the river, it may only move forward, but once it arrives the other side of the river, it may move forward, left, or right. A pawn can never move backwards.

- Training

  Based on what we know, it usually takes at least 30 years for a human to become a Master of Chinese Chess. How about a computer? What we know about a chess program, "KnightCap", is that it used *temporal difference learning method* (TD) to learn its evaluation function while playing on an online server and improved from a low rating to the level of US Master in just 300 or so games and 3 days of play.

  The author of the paper proposed to train a Chinese chess program by using TD and running it to play against itself. The goal of learning is to obtain some simple strategies, like piece values or weights.

- Why Temporal Difference Learning

  In Chinese chess, the average branching factor for a game tree is usually around 30 and a game averagely may last around 100 ply. Thus the size of a game tree is 30^{100}.

- Searching

  The author used alpha-beta search for better moves in a game tree. Alpha-beta search is superior to some typical search methods, like MiniMax, in the sense that the game tree keeps track of only the best branch so far, cutting off all other ones that will never by picked. Thus more game space is searched by alpha-beta than other methods in the same amount of time.

  No matter which search method is used, it is impossible to generate the full-fledged tree for the best move. Always, only several ply are envisioned.

- Horizon Effect and Quiescence Cutoff Search

  The author found horizon effect in 3-ply search. That's it cannot see a negative result in the near future just beyond 3-ply. One solution is to use 4-ply search instead. Unfortunately it was found to be very slow. The author finally chose quiescence cutoff search method in which the search continues, even though the specified depth has been reached, if a capture would possibly occur in the state.

- Evaluation Function

  Each piece type is considered a feature and assigned some weight. To evaluate how good a state is, an evaluation function is computed to obtain the weighted sum of features. A player gets a higher score if he has a feature available or the opponent lost it. It is the weights that need to be learned.

- TD(λ) and Updating the Weights

  With two factors, α and λ, the recursive expressions are given for the weights of pieces.

  α is the learning rate, determining how quickly the weights should be updated over time. λ is the discounted factor, determining how much of the possible future states should be taken into account now.

- Examples of Feature Table and Array of Weights

  Examples are presented to explain how, based on the feature table, the array of weights is updated step by step over time.

- Final Reward

  The final reward for a loser are defined as follows: If the game is a draw, the final reward is 0; if the board evaluation is negative, the final reward is twice of it; and if it is positive, then the final rewards is -2 times the board value. Similarly but with 2 and -2 switched, the final reward for the winner can be easily obtained.

  If a weight turns out to be negative, 0 is used instead. And to avoid a weight being too large, the author normalize the weights by projecting them into the range of 0 to 1.

- Summary of Main Events

  When it is time for either player to move, the program first updates weights for him using the equations of TD(λ), then does alpha-beta search for the best move, and let him act. The step alternates between the two players until the game finishes.

- After the Game Ends

  After the game ends, the program will calculate final rewards for both players and normalize the weights for pieces between 0 and 1.

- Results

  The Chinese chess program developed by the author has been executed in some 10-game series and some 100-game series. The learned weights from the previous series are passed over into the next series. Initially all weights are set to be 1.

- Observed Behavior and Weight Differentiation

  It was observed that in the early stages of the game, decisions made by players were pretty random.

  After 20 games, the weights began to differentiate and the Rooks turned out to be the most valuable pieces. The comparison was also made between the resulted weights and the ones assigned manually by H. T. Lau in his publications based on human playing experience. The author noticed the guards, ministers, and pawns might be overvalued compared to the cannons and the knights.

The parties played better after 250 games in the sense that the moves always aim to protect the valuable pieces of the player's own and capture those of the opponent's. And the weights are much closer to H. T. Lau's weights than the previous check point.

- Testing

  To test the results of the program, the author ran a policy using the weights learned after 250 games against another using the weights assigned by H. T. Lau. It turned out that the first player won 3 times, while the latter won only once. They also tied once.

- Future Works

  There are some work that may be done in the future. 8 different types of features can be considered in the evaluation function:

  - Piece values
  - Comparative piece advantage

    Trading pieces can favor the player who has more pieces left, which however has not been considered in state evaluation.
  - Mobility

    A same piece should be assigned a higher weight if it has more places to move into.
  - Board position

    Some board positions are important and should be valued.
  - Piece proximity

    The distance between some specific pieces should be considered so that an aggressive policy is achieved.
  - Time value of pieces

    Certain types of pieces have different weights all the way through a game.
  - Piece combinations
  - Piece configurations

    Some combinations of pieces can help win the game or give players advantages, say "a cannon behind a knight."

- Conclusion

  With the development of both computer hardware and software, strong Chinese chess programs have been built. It was claimed that a computer program will beat the "world champion of Chinese chess" before 2012.

- When and What

  The 2004 World Computer Chinese Chess Championship is scheduled on June 25-26, 2004. Big prizes are up for grab.

- References

2 Discussions

- Although it was claimed that the developed program may be too simple to be powerful, reinforcement learning should make it good enough and comparable to some rule-based complicated programs.
- *Did the author compare his work with others'?*

  It is just a student project. The author didn't do the comparison.
- So far .....
- *Is 5 games sufficient to show if the author's work does matter?*

  Again, it is just a student project.
- It is noteworthy that the author took a self-play approach to train the program. Purely self-play is not good since nothing can be learned to beat good players. However playing against previous results makes sense.
- It is interesting that with regard to Chinese chess the sizes of the sets of state and action decrease while the game goes on.
- The initial value may have an effect on the final result.