

Predicting Human Decision-making in Games

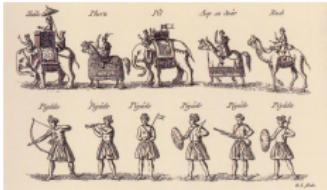
Who? Charles Zheng

From? Stanford University

When? May 2, 2016

History of Chess

Shatranj (\approx 500 AD)



Chess (1450 AD)



Shogi (\approx 1500 AD)



Doubutsu Shogi (Animal Shogi)



2009, Madoka Kitao





ピ ウ ブ ツ シ ょ う き
LET'S CATCH THE LION!









ピ ウ ハ ツ シ ょ う き
LET'S CATCH THE LION!



A

B

C

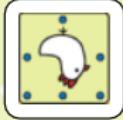


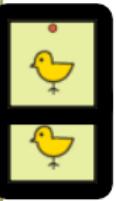
1

2

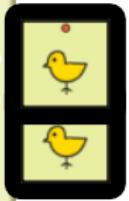
3

4











What is a game?

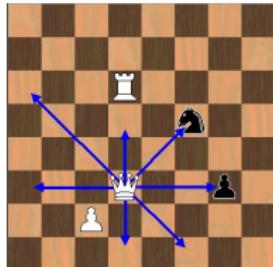
- Set of states $s \in S$.
- Set of winning states for player 1, $W_1 \subset S$.
- Winning states for player 2, $W_2 \subset S$.
- Each state has a set of legal actions $\mathcal{A}(s)$.
- Player 1 and player 2 take turns choosing the action.
- A *transition function* $P_a(s, s')$ determines the next state s' resulting from taking action a in states s .
- In *deterministic games*, the transition function equals one for one s' and zero otherwise.

Recorded Game Data

- List of moves made by both players: "1. Pawn from e2 to e4, 2. Pawn from e7 to e5."
- Convert to list of *state-action* pairs, (s_i, a_i) .
- s_i is state of game at the beginning of turn i , a_i is the move selected at turn i .

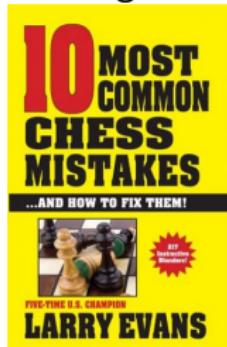
The prediction problem

- Database of games from players p_1, \dots, p_m with:
 - Date played
 - Player 1 and player 2 identities
 - List of moves → player, state, action pairs (p_j, s_i, a_i)
- In a *new game*, can we predict which move a given player p will make, given the game state s ?
- In other words: given (p, s) , guess $a \in \mathcal{A}(s)$.



Motivation

- First step for building superhuman AI player
- Detecting computer-aided cheating (Regan 2013)
- Objectively evaluating professional players (Matej 2011)
- Betting on results of professional games?
- Writing a book about ‘most common chess mistakes’



- Use machine learning to learn about human learning (?)

Section 2

Methods

Features

- In order to apply machine learning, need numeric features for the state s .
- Let $x_1(s), \dots, x_p(s)$ denote the features, $\vec{x}(s)$ =feature vector.
- Borrow features from chess programming: material, board control, king safety, etc.?
- Or try to do generic *feature selection* or *representation learning*?

Features

We use a minimalistic featurization. The 3x4 board is converted into a 136-length binary vector.

x_1	Player 1 has a king at (1,1)?
x_2	Player 1 has a rook at (1,1)?
x_{23}	Player 2 has a bishop at (1, 3)?
x_{132}	Does player 1 have two rooks in hand?

We also consider *second-order interactions*: e.g. $x_4x_{23} =$
Does player 1 have a pawn on (1,1) and player 2 have a
bishop on (1, 3)?

Policy model

- Fit the model

$$\Pr[A = a|s] = \frac{\exp[\beta_a^T \vec{x}(s)]}{\sum_{a \in \mathcal{A}} \exp[\beta_a^T \vec{x}(s)]}.$$

where \mathcal{A} is the set of *all possible moves* in the game (not just legal moves in s).

- When *predicting*, choose the action with the highest predicted probability among *legal* actions

$$\hat{A}(s) = \max_{a \in \mathcal{A}(s)} \beta_a^T \vec{x}(s).$$

Policy and values

- A policy $\pi(s, a)$ specifies a probability distribution of *actions* for each state $s \in S$.
- The value of a state $V^{\pi_1, \pi_2}(s)$ is the probability that player 1 wins if player 1 uses policy π_1 and player 2 uses policy π_2 .

$$V^{\pi_1, \pi_2}(s) = \begin{cases} 1 & \text{if } s \in W_1 \\ 0 & \text{if } s \in W_2 \\ \sum_{a \in \mathcal{A}} \pi_i(s, a) \sum_{s'} P_a(s, s') V^{\pi_1, \pi_2}(s') & \text{otherwise} \end{cases}$$

where $i = 1$ if it's player 1's turn and $i = 2$ if it's player 2's turn.

Evaluation functions

- In a deterministic game, let $s'(s, a)$ denote the state with probability 1 resulting from (s, a)
- Suppose player 1 knows player 2's policy. Then it would be rational for player 1 to choose a which maximizes the

$$V^{\pi_1, \pi_2}(s'(s, a)).$$

- However, humans are not perfectly rational nor omniscient. Perhaps players have an intuitive *evaluation function* E which approximates the true value function,

$$E(s) \approx V^{\pi_1, \pi_2}(s).$$

- This is also how chess engines work—idea first proposed by Shannon (1949)

Evaluation model

- Suppose a player does have a mental evaluation function E . Should they always choose a which maximizes $E(s'(s, a))$?
- Even in perfect information games, there is an advantage to being unpredictable!
- This leads to a multinomial model of player choice:

$$\Pr[A = a|s] = \frac{\exp[E(s'(s, a))]}{\sum_{a' \in \mathcal{A}(s)} E(s'(s, a'))}$$

Note that $E(s)$ need not be a probability: it could be a real number. Real-valued $E(s)$ may be more realistic, anyways.

How to fit the evaluation model?

- For fixed player p , let $\{(s_i, a_i)\}_{i=1}^n$ denote all the state-action pairs for that player in the database.
- Fit a logistic evaluation model, minimizing the loss

$$\sum_{i=1}^n \vec{x}(s'(s_i, a_i))^T \beta - \log\left(\sum_{a \in \mathcal{A}} \exp[\beta^T \vec{x}(s'(s_i, a))]\right).$$

- Loss function is convex!
- For large feature models, recommended to use L1 and L2 regularization.
- To predict, choose $a \in \mathcal{A}$ maximizing $\beta^T s'(s, a)$.

Section 3

Data

Doubutsu Shogi on LittleGolem

- Data obtained from littlegolem.com, a free turn-based game site, using `scrapeR`
- 85 players, 727 games, 17094 turns (state-action pairs)
- Oct 2012 to Apr 2016
- Usernames have been anonymized

