CSC2541-f18 Course Project Proposal
Human-Like Chess Engine

Reid McIlroy-Young
University of Toronto
reid.mcilroy.young@mail.utoronto.ca

Karthik Raja Kalaiselvi Bhaskar
University of Toronto
karthikraja.kalaiselvibhaskar@mail.utoronto.ca

Abstract

We propose to create a chess engine with human-like behaviour. To do this we would take an existing engine and replace the policy selection with an algorithm that attempts to minimize risks in addition to winning. We do not know which algorithm will work so propose three as starting points.

1 Method

1.1 Data

For this project we need games played between humans with average skill levels, instead of the collections of advanced player's games more commonly used [1]. To this end we used a subset of the 432,335,939 games on Lichess [2] (on November 2018). Lichess maintains a player ELO rating system which allowed us to extract games played between evenly matched ELO players. We considered players evenly matched if they had the same ELO when rounded up to the nearest hundred. The counts for each bin are shown in figure 1. During training games were randomly sampled from a selected bin, thus giving an engine that has only been trained on one ELO range of players. We also set aside the 22,971,939 games from Septmber of 2018 as a holdout set.

1.2 ELO

The main system used for evaluating chess players and evaluating chess engines ELO [4] is much less useful than one might expect. Since ELO relies on the win rate of a single individual against other ranked individuals within some time period and does not refer to any external factors [4] it is excellent at rating players within a well connected player community. But when attempting to compare chess players or engines that are not already ranked or capable of directly playing each other the ELO does not work, this is before considering that engine performance varies greatly depending on the computing resources provided. Thus when a group presents an ELO rating for their engine, e.g. [5], their ELO is only well defined within their testing system. Therefore is more useful to view ELO as a simple ranking and to ignore the values when comparing between groups.

We have two separate ELO rankings used throughout our paper. First, there is the ELO used by Lichess, these numbers are derived based on human players playing ranked events and are accurate for most individuals. Second the ELO ranking used by Leela chess to compare different generations of their engines, these are less accurate and prone to inflation, since every new engines must beat the previous ones.

1st Deep Reinforcement Learning, Final Project Report (CSC2541-F18 2018), Toronto, Canada.

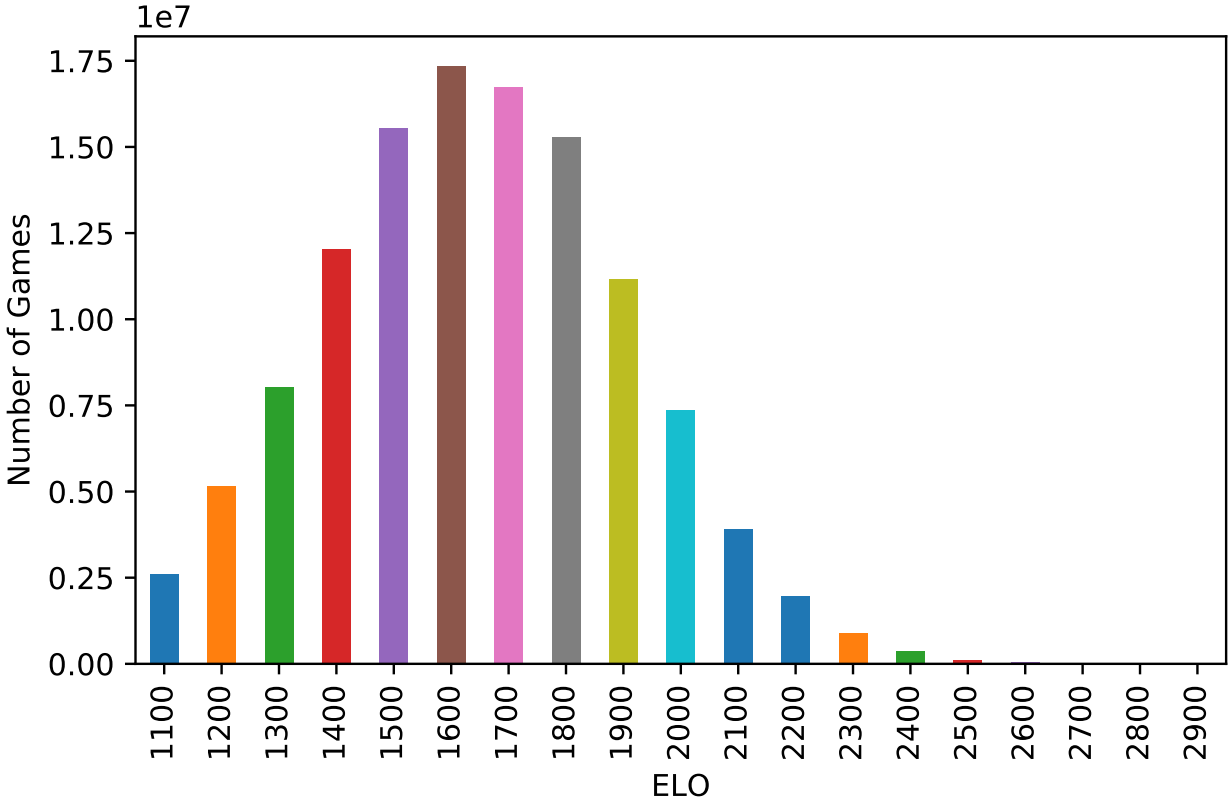

Figure 1: Distribution of games between similar ELO players

We did attempt to create our own ELO ranking system for comparing our own engines but the overhead in getting well defined ELOs for the benchmarking is high and simply ranking a series of engines provides nearly the same information. That said, measuring our own ELO would be considered in future work.

1.3 Leela Chess

The *AlphaZero* chess engine created by *DeepMind* is closed source and does not have an version available to use [5]. There is though an free and open source implementation of the zero training and playing system, *Leela Chess Zero* [3]. As the name suggests *Leela Chess Zero* is trained via self play and besides tweaks to the hyper parameters and residual layers is a *clean room* reimplementation of the *AlphaZero* system. For our work we replaced the self play with human games, thus going back from and unsupervised to a supervised domain.

The reinforcement learning

Note that it is impossible to identify ? from watching human behavior in a single task. This is because any ? is fundamentally indistinguishable from an infinite set of reward functions that yield exactly the policy observed in the task. We introduce the idea of behavioral equivalence below to tease apart two separate issues wrapped up in the challenge of identifying rewards. Definition 1. Two reward functions , 0 R |S| are behaviorally equivalent in all MDP tasks, if for any (E, R), the set of optimal policies for (R +) and (R + 0) are the same

2 Experiments

2.1 Training

To

3 Section

4 Conclusions

- Intro
- lit review
- data
- chess engines
  - leela
  - stockfish
  - random agent
- methods
  - how to measure human like
  - supervised vs self play
  - leela training config
- results
  - win rates
  - kl
  - path following
- discussion
  - haibrid is not very good
  - future work needed
  - new engines are better
  - computational limits of chess
- conclusion

- elo dist
- board trajectories
- KL divergences
- winrate
- tie rates

References

[1] Omid E David, Nathan S Netanyahu, and Lior Wolf. Deepchess: End-to-end deep neural network for automatic learning in chess. In *International Conference on Artificial Neural Networks*, pages 88–96. Springer, 2016.

[2] Thibault Duplessis. Lichess, 2018.

[3] Gary Linscott Gian-Carlo Pascutto. Leela chess zero, 2018.

[4] Mark E Glickman. The glicko system. *Boston University*, 1995.

[5] 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, 2018.

[6] Joona Kiiski et al. Tord Romstad, Marco Costalba. Stockfish: A strong open source chess engine, 2018.