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.

Since the 80s computers have been working towards perfecting the game of chess, with the most notable even being the 1997 match where Kasparov was defeated by an IBM machine, *Deep Blue*[5]. The goal of IBM was to create system that could win chess games and to this end Deep Blue took some very inhuman actions. Most notably, at a critical juncture *Deep Blue* did not take the obvious line and instead took a much more delicate and refined path that requires much more modern chess engines to discover. The move was thus, both inhuman and unexpected, it was later discovered, to have been randomly selected[9]. This kind of decision making is only possible with algorithmic systems.

Modern chess engines have far exceeded human ability [8] in the last two decades, but despite humans learning from them there are significant difference between human and computer chess strategies. This is most evident when looking at end games, if there are seven pieces in a legal configuration the optimal moves for each player are known, these solutions are called endgame tablebases and are available commercially. Six pieces tablebases are available for free and are much smaller 100 TB vs 90 GB compressed, so those are what most researchers use. Again when humans are compared to computers their moves are different. Humans will be less likely to give up pieces, will make generalization mistakes where they assume a pattern holds more than they should and will play to their open having more opportunity to make a mistake than the optimal move[1].

The differences in human and computer learning of chess suggest the mechanism of learning is different. We propose to train a chess engine that takes moves more like a human. This would be done by modifying the policy selection of an existing chess engine, hopefully *Leela Chess Zero* an open source implementation of the *AlphaZero* system [8]. We also have access to millions of human vs human chess games played on *Lichess* so we can do training against real data, not just self play. Our interest in human like chess engines arises from attempts to understand human decision making with the goal of better understanding why and how humans make decisions, but there are also uses for it in machine learning. Much of the contemporary machine learning algorithms have direct biological analogues, neural networks are one such example, by continuing to explore biological learning we may uncover new techniques for machine learning.

Chess engines are a well known and deep field with many solutions existing. The current most well know engine is *Stockfish* which uses a tree search based system to explore the game state much like

Preprint. Work in progress.

a Markov decision process, but with the weights not being learned. The tree based models are not probabilistic which makes variance estimates difficult so we will initially explore neural network based models, of which *Leela Chess* is the best regarded and already uses a reinforcement learning component. There are others could also be used, including *DeepChess* [3] and *NeuroChess*, with *Giraffe* [6] a Temporal-Difference Learning based system being the best secondary candidate for adaptation. There is also a small amount of lirture around cheat detection [2] whcih could help us improve our evaluation criteria.

We do not know yet which modification to the policy selection will yield our desired results so we have started collecting a set of candidates. One method is to add a strong risk minimization component from some of the work in robotics[10]. Another is to change the state exploration criteria to those used by robots moving in a complex environment, again building on robotics[7]. A final approach based on work with cellular automata is to make the moves and/or board states stochastic[4]. The goal of all these techniques is to make the chess engine more sensitive to risk and less likely to embark on difficult trajectories and instead to play within a smaller area governed by simpler heuristics like humans are known to (outside of masters).

References

[1] Ashton Anderson, Jon Kleinberg, and Sendhil Mullainathan. Assessing human error against a benchmark of perfection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 11(4):45, 2017.

[2] David J Barnes and Julio Hernandez-Castro. On the limits of engine analysis for cheating detection in chess. *Computers & Security*, 48:58–73, 2015.

[3] 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.

[4] Vijaykumar Gullapalli. A stochastic reinforcement learning algorithm for learning real-valued functions. *Neural networks*, 3(6):671–692, 1990.

[5] Garry Kasparov. The chess master and the computer. *The New York Review of Books*, 57(2):16–19, 2010.

[6] Matthew Lai. Giraffe: Using deep reinforcement learning to play chess. *arXiv preprint arXiv:1509.01549*, 2015.

[7] Steven M LaValle and James J Kuffner Jr. Randomized kinodynamic planning. *The international journal of robotics research*, 20(5):378–400, 2001.

[8] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

[9] Nate Silver. *The signal and the noise: why so many predictions fail–but some don't*. Penguin, 2012.

[10] Aviv Tamar, Dotan Di Castro, and Shie Mannor. Policy gradients with variance related risk criteria. In *Proceedings of the twenty-ninth international conference on machine learning*, pages 387–396, 2012.