UNIVERSITY OF WATERLOO

Hybrid Chess Engine

Mitchell Catoen, Ross Duquette, Jordan Edwards, David Ferris

Winter 2019 – ECE457B Computational Intelligence
ECE Dept, University of Waterloo
Instructor: Fakhri Karray

Abstract and Overview

This purpose of this project is to design and build a chess engine which maximizes play by applying specific strategies at the stage of the game when they are most effective. Three engines based on (1) a deep CNN, (2) a genetic algorithm, and (3) statistics and search techniques all propose moves, and a fuzzy-logic system selects the strongest move given the stage of the game. This hybrid approach compensates for deficiencies the techniques possess individually, and results in an overall stronger engine. The engine is approximately 1800 ELO.

"Play the opening like a book, the middle game like a magician, and the endgame like a machine."

Rudolf Spielmann

Problem Formulation

Most chess engines utilize a single neural network architecture, training methodology, and/or search algorithm to evaluate the optimal move to play. Hybridizing various approaches can result in overall improved engine performance, particularly when the game phase (opening/middle/end) is considered.

Opening

- Well-known theory
- Large historical corpus for supervised learning
- Large search tree

Middlegame

- Requires "creativity"
- Novel positions
- Massive search tree
- Difficult to calculate - need heuristics

Endgame

- Checkmate calculations
- Active kings
- Medium search tree

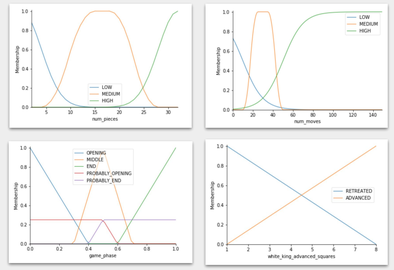
Tools & Algorithms:

Algorithm:

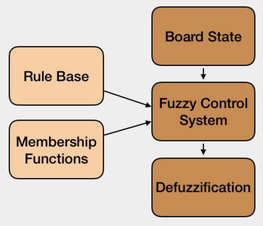
1. Evaluate game phase using a Mamdani fuzzy inference system.
2. Select engine given game phase.
3. Make move, repeat (1).

Fuzzy Game Phase Estimator

Fuzzy Set: boards in the opening/middle/endgame.

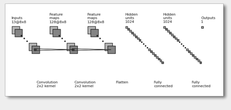


Membership Functions



Opening Engine: Convolutional Board Evaluation

- Trained CNN on over 1 million board states seen in GM games



Middlegame Engine: Genetic Algorithm

- Positional priors were determined through evolutionary learning

$$s(b_i) = \frac{1}{2} + \frac{1}{2} \cdot \frac{i}{N_m} R_i \pm 0.05 \cdot M_p$$

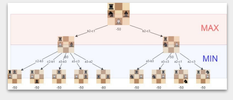
$M_p : Grandmaster\_move$

$R_i : Result\_of\_game_i$

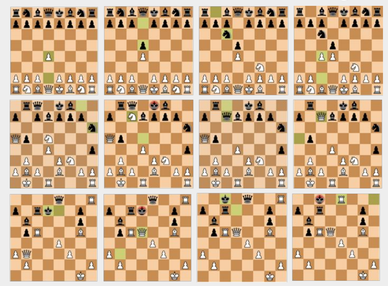
$s(b_i) : State\_of\_ith\_move$

Endgame Engine: Minimax Search

- Rapidly explores the tree of possible moves
- Branches are pruned based on piece value heuristic

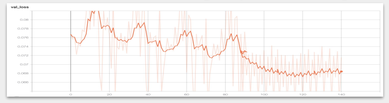


Experiments & Analysis



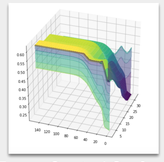
Deep Learning Experiments

- Convolutional versus fully-connected
- 2-headed model (from_square, to_square)
- Multiple input data encodings (one-hot legal moves, board encodings, piece weights, etc.)



Observations

- The fuzzy inference system is extremely robust - monotonically increasing
- Tying record against Stockfish 5 (ELO 1780)



Fuzzy Control Surface

Conclusion

This project demonstrates that a hybrid chess engine combining techniques from statistics, deep learning, and fuzzy logic can be stronger than any system alone. It also demonstrates that supervised deep learning on strong player datasets can produce excellent results in the opening phase where situations are not entirely novel. This bucks the trend in artificial intelligence where reinforcement learning systems are increasingly used for complete information games like chess (ex. Alphazero). It is also shown that fuzzy logic can be used to robustly measure the progress and phase of a chess game, even with a small rule base.