Chess AI Project Proposal

By: Ryan Handlon

Summary

Recently my friends and I have gotten into chess and play together quite regularly. I currently have a rating of around 1100 elo on chess.com compared to grandmasters who are rated 2500 elo and higher. So, while I'm not the greatest at the game, I think it would be cool to build an AI that could at least beat me. For my project I would like to build a homemade Minimax/Alpha-beta pruning based chess AI. I plan on trying and implement my own board evaluation system, book moves, and endgame tables. I'd also like to implement a predesigned board evaluation system so I could compare it with mine to see the differences. The end goal of this project would be to build an AI that could beat both me and my friends consistently.

Project Description

My Motivation for building a chess AI is that it's recently become a hobby of my friends and I. We've been playing each other semi competitively for a couple months now and during the day when I need a quick break, I'll open chess.com and do some puzzles for a bit. Knowing that there are already so many strong chess Ai's out there, I think it would be cool to build my own that could hopefully beat my friends and me. On chess.com I am currently rated 1120 in rapid (10 minute) chess and 1579 in puzzles, my formal goal would be to try and build an AI that could get that rating or higher, and or beat my friends and I in a tournament.

The idea of chess playing robots has been around since 1769, however back then it was a simple automaton with a human hidden inside actually playing the game. Since then a lot of progress has been made, from Alan Turing publishing a program on paper that could play a game of chess in 1951 to Deep Blue beating World Grandmaster Gary Kasparov 3.5-2.5 in 1997, to AlphaZero, a neural net Ai that uses Monte Carlo tree search, beating Stockfish 28-0 with 72 draws in a 100 game match. As chess AI's evolved, they used a variety of techniques including minimax and alpha-beta pruning searches, genetic algorithms, machine learning, and neural networks, all backed by a whole lot of computing power. These methods would be altered slightly over the years using things like being allowed to reference opening book moves, different ways of evaluating the current board state, and endgame tables showing how to move based on what endgame situation the AI is in.

My proposed approach is to build a chess AI that uses minimax and alpha-beta pruning for the bulk of the game playing. My implementation will use multiple attributes to evaluate each board position based on three main factors. Those being the base value of the pieces, location of the pieces on the board, and location of pieces to each other.

In chess each piece has a relative value used for relative comparison of how powerful they are. Situationally piece values differ, but the base values of the pieces in chess are that pawns are 1 point, knights and bishops are 3 points, rooks are 5 points, queens are 9 points, and kings are invaluable.

The location of pieces would be factored in by looking at different attributes based on each piece. For example, a pawn can promote to any piece other than a queen if it gets the other end of the

board, so the closer to the end of the board it is, the more valuable it is. Another example would be that 3 point pieces cover more squares in the center of the board than on the edge. A knight in a corner of the board can see 2 squares and a knight on an outer edge can see 4 squares, but a knight in the middle of the board can see 8 squares. Similarly, a bishop on an outer edge of the board can see 7 squares, but in the center, it can see 13 squares.

For pieces related to each other I'd look at things like rooks being on the same file or rank, because then they support each other making them more powerful, or that a bishop is weaker if there is a pawn chain on the same square color as it because the amount of squares it can see is blocked.

Another thing I plan on implementing is knowledge of book moves, allowing the AI to have a basic knowledge of good practices for starting the game. I also plan on looking into implementing endgame tables, but I'm a little confused on how they work so It may be tough. Should I implement all of this and still have time I want to implement a board position evaluation system I find on the internet and compare its efficiency to my own and see how I can improve mine.

Project Plan/schedule

I plan on splitting my project into three biweekly deliverables, the first one being due March 26th and the last being due on April 23rd. This would leave about a week to clean up the project and write a report before the submission sue date of April 29th.

Deliverable #	**Tasks to do:**	**AI Functionality when tasks done**	**Due Date**
Deliverable 1	Minimax/Alpha-Beta Pruning Implemented on a Chess API.	The AI should be able to play the game at this point.	Due: 3/26
Deliverable 2	Implement my own custom board evaluation system. If time also implement a predesigned one off the internet	The AI should be able to capable of playing the game hopefully intelligently and meat at least a beginner level player.	Due: 3/9
Deliverable 3	Implement book moves and start playing against better people and AI's. Fine tune my board evaluation system. Start written report. If time implement endgame tables.	Hopefully, if all things go well, the AI should be able to beat me and my friends at this point	Due 3/23
Final Submission	Write report and finalize everything	Working chess Ai and a written report turned in.	Due: 4/29

Equipment and Facilities Needed

No equipment or facilities needed for this project other than the laptop I currently own.

Budget

No budget needed.

Biographical Sketches

I am the only person working on this project. One of my recent hobbies involves playing chess and I'm currently rated 1120 in rapid (10 minute) and 1579 in puzzles on chess.com. While I am by no means a master, my rating indicates I'm about average or slightly above average at the game. I believe I have a pretty good understanding of the game and strategies involved.