Michael Kronovet

**Update 2**

I was able to include all of the features I had specified in Update 1: Chess AI with neural network evaluation function and minimax search algorithm and sockets for multiplayer. I also added a mode where players can practice against themselves and a tutorial screen with short animations detailing piece movements and castling in case players want to reference how the pieces work.

**Update 1**

My idea has gradually shifted into what is now a chess game with a neural network evaluation function (tells the AI how good a certain board is) and a minimax search algorithm (searches for moves that the AI can make and then uses neural network to say how good each move is). My game also has sockets for multiplayer. Initially, I planned to use machine learning for the search algorithm in addition to the neural network, but upon further inspection the machine learning algorithms I was looking into are not nearly as relevant for chess as minimax for searching moves. Also, I initially wanted to use HTML and CSS to turn the game into a website, but having all these features is already really overwhelming with the complex AI and sockets so it will just be in tkinter on the computer.

**Previous Proposal from TP 1**

Currently, chess AI is not as efficient as it could be. The way chess AI is usually implemented is with the mini-max algorithm or other similar methods which are based off arbitrary heuristics determined by people. Programmers tell the AI what moves are better and what board positions are better than other. Essentially, this takes away a lot of the intelligence of the AI, since it is just doing what you tell it to do without much of its own input. This is why I want to make a chess AI that is smarter and can generate moves by itself without much input from these arbitrary scores. I plan to do this by implementing a neural network that will be trained by observing the games of chess grandmasters in order to make makes. My plan is for the neural network to have 64 input neurons for each board position. The neurons will then go through several hidden layers: At each layer, the neural network will recall the grandmaster moves it encountered with similar board compositions, and then it will merge the board positions and those recalled moves until they converge to a single output move. Ideally, I would like to implement the neural network myself, however depending on how equipped I am to do this, I would consider using TensorFlow in order to build and train the neural network.

Also, depending on how well the neural network works, I would like to try and implement other machine learning algorithms as well in order to see how they do in determining the optimal chess move. I am considering using simulated annealing where the AI would be able to filter through global minima (moves that are relatively good but not the best) in order to find effectively find the global maxima (best move in the situation). I am also interested in using genetic algorithms for determining the optimal moves. For this method, I would take a fairly random assortment of the moves, combine the moves to make child moves (probably do this by trading the piece and the move between the two given parent moves) and repeat until they converge to an optimal move.

In addition to implementing superior AI, I would also like to be able to turn the chess engine into a web application. I would use Django, HTML, and CSS in order to accomplish this. In order for this game to accessible to others and not just solely on my computer, I want to make a website out of the game. I would implement the backend chess code in python and use CSS and HTML to make the graphical user interface.

As a last step, I would also like to include multiplayer into the project. The central idea of the project is more focused on AI development, so this would most likely be a final step after everything else has been finished.

Steps:

1. Finish basic chess game
2. Attempt programming neural network by myself
3. Either finish making neural network by myself or implement neural network using TensorFlow
4. Program either simulated annealing or genetic algorithm method for chess AI
5. Turn application into a website
6. Implement sockets for multiplayer