23.12.2023

Ich habe alles aufgesetzt und die grobe Struktur geplant. Ich habe mit der Berechnung der legalen Züge begonnen, welche ich alle in einem Array speichere. Ich habe mit den Türmen (rooks) angefangen. Außerdem habe ich die legalen Züge des Läufers (bishops) und der Dame (queen) berechnet.

24.12.2023

Ich habe die legalen Züge der Springer (knights) berechnet.

25.12.2023

Ich habe mit den legalen Zügen der Bauern (pawns) begonnen. Dafür habe ich meine benutzerdefinierten FEN-Strings erweitert.

26.12.2023

Ich habe die legalen Züge der Bauern abgeschlossen.

31.01.2024

Um die legalen Züge des Königs zu berechnen, muss ich zuerst alle Züge des Gegners berechnen. Außerdem muss ich die Züge herausfiltern, die den König in Schach setzen.

05.01.2024

Ich habe die Berechnung der Rochade (castling moves) durchgeführt.

13.01.2024

Ich habe eine FEN-String-Datenbank verwendet und 56.000 FEN-Strings berechnen lassen. Um zu testen, ob alle korrekt sind, habe ich auch eine Python-Schachbibliothek die gleichen FEN-Strings überprüfen lassen. Der erste Fehler, der mir in meinem Code auffiel, war, dass der König Gegner schlagen könnte, die gedeckt sind.

22.01.2024

Ich habe die restlichen Fehler im Code gelöst und einen Test mit 28 Millionen Positionen durchgeführt, ohne Fehler.

18.02.2024

Ich schreibe Code, der verschiedene Perft-Tests durchführt, damit ich später einfach Optimierungen am Code für die legalen Züge vornehmen kann.

22.02.2024

Ich beginne mit dem Minimax-Algorithmus mit Multithreading.

08.03.2024

Um den Minimax-Algorithmus zu verbessern, werde ich die Berechnung der legalen Züge an ein CNN weitergeben innerhalb des Minimax-Baums, da dies schneller geht. Dafür schreibe ich einen Algorithmus, der die Ausgabe-Indexe des CNN in uint16_t-Züge umwandelt und eine Umkehrfunktion dafür erstellt. Ich gehe davon aus, dass es inklusive Beförderungen 4272 Züge gibt (also 64*64 + 176 Beförderungen).

08.04.2024

Ich strukturiere meine Dateien und den Code und packe alte Sachen vorerst in einen separaten Ordner.

09.04.2024

Ich bereite die Umwandlung von Bitboards in Eingabe- und Ziel-Tensoren vor, um das neuronale Netzwerk zu trainieren. Da die Speicherung von PyTorch-Modellen in C++ nicht möglich ist, muss ich das Netzwerk in Python trainieren und erstelle dafür eine .csv-Datei mit den Trainingsdaten.

10.04.2024

Ich habe das erste Testmodell trainiert, um die Funktionen und alles zu testen. Außerdem schreibe ich Genauigkeitstests für Modelle.

15.04.2024

Nach einigen Tests zeigte sich, dass das Modell vergleichsweise, selbst im kleinstmöglichen Zustand, langsamer ist als mein Code, der die Züge manuell berechnet. Deshalb kann ich das Modell nicht verwenden und nutze stattdessen meinen Code.

20.04.2024

Ich habe die Bewertungsfunktion mit Standardbewertungsmethoden begonnen.

22.05.2024

Ich habe kleinere Fehler im Quellcode gelöst und teste nun gegen einen Schachbot mit einer Elo von 550.

24.05.2024

Ich habe in der Bewertungsfunktion das Schachmattsetzen als höchste Priorität festgelegt, wodurch der Gegner mit 550 Elo besiegt wurde. Vor weiteren Tests will ich aber erst den Minimax-Algorithmus optimieren. 10.05.2024 Ich habe die Vorbereitungen für das Move-Ordering gemacht.

11.06.2024

Ich habe Move-Ordering implementiert.

13.06.2024

Ich habe Positionen zur Bewertung hinzugefügt.