Advancing Chess Engine Design with Elo-Integrated Evaluation (EIE)

Paul-Andrei Aldea
University of Michigan

Pranav Bangarbale
University of Michigan

Jerry Li
University of Michigan

1 Background

The source code for the project may be found on GitHub.

1.1 Engine Overview

Existing chess engines have the capability of extensively analyzing chess positions to vast depths. As a reference, Stockfish is capable of multi-threading computations across 1024 CPU threads, and recall up to 32 TB worth of positional data in a transposition cache [1] [4]. Furthermore, Stockfish is capable of prioritizing certain branches of computation based on a priori likelihood that a branch is likely to yield a positive outcome. Generally, this algorithm is implemented as a specialized alpha-beta search over a given positional representation. As of 2024, the playing strength of the Stockfish engine is an estimated 3642 ELO [2]. To place the strength of a chess engine in perspective, the highest rated human chess player is rated at 2831 ELO, while the over-the-board (OTB) classical average rating is around 1400-1599 as defined by FIDE [3].

Given the playing strength of the Stockfish engine, it is also the most common baseline to the evaluation of a chess position. Typically, games will feature the evaluation given by Stockfish during analysis to guide the player and explain any mistakes made during play. Since a chess engine is guaranteed to be stronger than a human player, the given evaluation can be trusted to provide a continuation with a higher playing strength and accuracy than the analysis made by a human player. Providing engine feedback to players is commonplace among all known online chess hosts [4], and even chess tournament broadcasts will feature a live engine analysis. To this end, it is not unfair to say that the reliance on engines is commonplace in the chess field, and likely one of the most commonly used mediums by chess players to learn the game. As a result, it is paramount that engine evaluation is as accurate as possible in order to not mislead players and ensure accurate feedback.

[1] A transposition cache is a data cache which stores previously seen positions to avoid redundant computation.

[2] ELO is the baseline system utilized to denote the strength of chess players. Further details are given in section 1.2

[3] FIDE is the International Chess Federation, and is the governing body over the chess competitions.

[4] Lichess and Chess.com are the two largest online chess websites, and both feature engine evaluation.

1.2 ELO Ratings Overview

It is also important to define the skill level of a chess player, for the purposes of this project. Chess players are typically evaluated using a metric known as ELO, which measures their skill level relative to others. In this project, we use ELO ratings as a baseline approximation for skill level, because they are readily available in most open source chess datasets and they provide a reasonably accurate estimation of ability. Typically, ELO ranges anywhere from around 1,000 for a beginner to 2,500 and over for a grandmaster; although, the best computer chess engines can reach levels upwards of 3,000. An average or intermediate ELO is somewhat arbitrary, but for this project we focus on ELO ratings from 1000-2000 - players who range from amateur to just above average.

1.3 Problem & Motivation

Modern chess engines, such as Stockfish [1], have grown increasingly powerful at analyzing and calculating evaluations of given chess positions. Most chess engines are capable of not only providing several sequences of optimal moves which are most likely to yield the highest advantage, but also quantifying the perceived advantage for a side in a given position. Most evaluations of chess positions primarily involve a heuristic approximation of several factors, commonly learned through a deep learning approach, along with a branch-like exploration of possible moves. Although chess engines demonstrate extraordinary playing strength, their evaluations of positions are often limited to the advantage under optimal play, which may not be very applicable to the average player if

those ideal moves are unlikely to be executed. More explicitly, the evaluation given by an engine is unlikely to accurately reflect the objective advantage of a human player, likely to make sub-optimal moves, and thus falls short of achieving an accurate representation of the given positional advantage.

1.4 Primary Contribution

We instead present Elo-Integrated Evaluation (EIE), a novel approach to position evaluation, considering the playing strength of the players, to provide a more accurate metric for assessing practical outcomes and decision-making in real-world games. EIE demonstrates up to a 10% increase in next-move prediction accuracy, and up to a 20% increase in evaluation accuracy over existing chess engines.

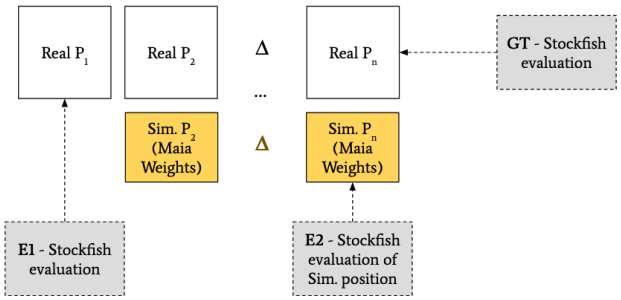


Figure 1: High level EIE methodology.

1.5 Task Allocations

All group members contributed equally to the project. We each worked together to write the reports and create the presentations, as well as write the code for our proposed method and evaluations.

2 Related Work

The models we use to build our evaluation methodology are as follows.

1. Maia-Chess
2. Leela-Chess Zero
3. Stockfish (also discussed in Section 1)

Leela Chess Zero (Lc0) [2] is an open-source chess engine that uses a neural network architecture inspired by AlphaZero to play and improve its chess abilities. It combines deep reinforcement learning with Monte Carlo Tree Search (MCTS), training its network through self-play games without any hard-coded heuristics or opening knowledge. Lc0's design allows it to evaluate board positions and generate moves based on patterns learned from millions of games, enabling it to achieve superhuman performance while providing a unique, human-like playing style. Our project does not use Leela Chess Zero's chess engine itself, but it uses its model infrastructure to plug in Maia chess weights (discussed in the following paragraph).

The Maia-Chess project [3] develops chess engines designed to emulate human play at specific Elo ranges, particularly for lower ratings.These models are based on fine-tuned versions of the Leela Chess Zero (Lc0) neural network architecture. Maia's training process involves adapting Lc0's architecture, which uses a convolutional neural network paired with Monte Carlo tree search for decision-making, to predict human moves rather than optimal ones. By analyzing millions of online chess games played by individuals within targeted Elo brackets, Maia trains its neural network to replicate the move choices made by humans, even when those moves deviate from optimal play. The architecture processes board states and outputs probabilities for moves that players of the given rating range would likely choose. For example, Maia at 1000 ELO captures common blunders and tactical oversights, while the 1500 and 2000 ELO models reflect progressively stronger but still imperfect decision-making, consistent with their respective skill levels. This approach makes Maia an effective training tool, providing players with an engine that mirrors their gameplay and helps them identify areas for improvement in a relatable and realistic manner.

Stockfish is one of the most powerful and widely used chess engines, designed to analyze chess positions and generate highly accurate moves. It relies on a combination of brute-force search and sophisticated heuristic evaluation to explore vast numbers of possible moves and positions at incredible speed. Stockfish uses an alpha-beta pruning algorithm to efficiently search the game tree, considering millions of positions per second, while its evaluation function assesses the quality of each position based on factors like material balance, piece activity, king safety, and pawn structure. Unlike neural-network-based engines, Stockfish focuses on traditional artificial intelligence techniques, making it highly efficient and versatile for tasks such as game analysis, opening preparation, and tactical training for players at all skill levels.

Our proposed methodology is novel because it combines the relative advantages of all of these engines to create a more accurate chess position evaluation for games between players of lower ELO ratings. See Section 3 for more detail.

3 Methodology

3.1 High Level Methodology

At a high level, the EIE methodology constructs a novel evaluation function for chess positions (see Figure 1). We make use of three separate models as part of our evaluation, each covered in Section 2 - the specific uses of these models are also covered later in our report. Our evaluation method attempts to predict the next series of moves more accurately than the canonical engine, Stockfish, and thereby return a better evaluation of the current position. In theory, our evaluation should be more effective since Stockfish is trained to find the most optimal move at a given position while our goal is to find the most likely move - which might not necessarily be the most optimal.

For our evaluation methodology, we must approximate a ground truth evaluation as chess is not a solved game. Since Stockfish is the current leading chess engine, and can likely evaluate a position more accurately than its engine counterparts, we will use its evaluation as our ground truth. In general, the strength of a chess engine is determined by having it play games against other chess engines - a scenario where Stockfish regularly wins major chess engine tournaments, demonstrating its superior playing strength. We then take it as a good choice for approximating the ground truth of a chess position. Our evaluation metric is calculated as follows (see Figure 1):

1. Given a chess position in a game, let Stockfish's evaluation be A.
2. Load the Maia-chess model for the given ELO rating of the two players.
3. Predict Δ number of moves in the future using the loaded model. Calculate the Stockfish evaluation at that position in the game. Let this evaluation be B.
4. Take Stockfish's evaluation Δ moves from the current position in the actual game. Let this evaluation be C. Let us consider C to be the ground-truth evaluation at this point in the game.
5. The evaluation metrics will be $|A-C|$ for Stockfish, or the baseline, performance, and $|B-C|$ for our model. Using this methodology, we can accurately measure the error of both models and determine which is more effective.

An advantage of this evaluation method is that it parallelizes nicely, since games and positions are independent. Thus, we were able to run evaluations on multiple games in parallel. One disadvantage is that our method relies on Stockfish's evaluation as the ground-truth - even though this is the best existing chess engine, it may still make mistakes. However, after researching other available engines, we were unable to find one that can match Stockfish's performance.

3.2 Dataset

First, the dataset we used consists of chess games actually played on Lichess.com's platform - these are publicly available for download on the website. The dataset was then filtered to only include games between players in the rating range of 1000-2000. In our method, we do not deal with cases of mismatched rating. As shown in Figure 4, one particular chess game consists of both metadata and the specific moves that were played in the game, in FEN notation (a particular notation used for expressing chess board positions). Using the WhiteElo and BlackElo fields, we were able to filter games based on the minimum and average ELO rating of the two players before evaluating our methodology. The exact dataset size used was 25 games; although this seems relatively small, based on our evaluation methodology, explained in the next section, the total number of positions evaluated is around 21,000. Due to the time constraints of the class and the restrictions of our hardware, we were limited to this dataset size, though future evaluation is possible. In order to process the data, we used the Python Chess library, which has the functionality to store intermediate board states, push/pop moves, and quickly extract information evaluation information.

3.3 Environment

The environment consists of a standard chess board and its related interactions. A player may interact with the environment by making a move on the chess board. The other player follows with their own move. This sequence repeats indefinitely until the game results in checkmate, in which one player wins, a draw, timeout, or voluntary resignation. All interactions on a chessboard are limited to the pieces. As chess is a well-studied game environment, we will not discuss the intricate details of chess rules in this report.

4 Evaluation

4.1 Testing Environment

We used the following Cloudlab testing environment to run all of our evaluations.

- 2 x Intel Xeon Silver 4114 10-core CPUs @ 2.20 GHz.
- 1 x NVIDIA 12GB PCI P100 GPU.
- 192GB ECC DDR4-2666 MT Memory

4.2 Evaluation Dataset

As mentioned previously, our datasets are real chess games taken from the Lichess.com database of played games. We took 25 games from the rating ranges of 1200, 1500, and 1800. We picked these 3 ELO ratings to encompass low, middle, and high rated players while still being within the

Maia chess dataset range. Since finding games with exact ELO averages of 1200, 1500, and 1800, is challenging, we instead chose to average the rating of the players and round to the nearest multiple of 100 ELO. For each game we run evaluation at each move of the game for delta values 1 to 7 and since the average chess game is about 40 moves, we have approximately 21,000 training samples. We did not have a training set or testing set; we simply used an evaluation dataset that consisted of data points that had not been used to train the underlying Maia-Chess models. Additionally, we did not have a training process of any of the models we used. Instead we used the dataset primarily to evaluate the model we implemented.

4.3 Results: Baseline vs EIE Performance

At the select ELO of 1200, Stockfish performs at an error of 85 centipawns from the "ground truth" at $\Delta = 1$, with a roughly linear increase to an average difference of 200 centipawns by $\Delta = 7$. Our model outperforms Stockfish at $\Delta = 1$ with an average difference of 65 centipawns, but as Δ increases, performance declines, aligning with Stockfish at $\Delta \geq 5$. Theses trends are shown in Figure 3. Additionally, at $\Delta = 1$, our model can consistently outperform Stockfish, seen in Figure 2. For players rated 1500, Stockfish achieves a 65-centipawn difference at $\Delta = 1$, while our model improves to a 50 centipawn difference. However, performance parity occurs at $\Delta = 4$, after which our model slightly underperforms. At 1800, Stockfish records 55 centipawns at $\Delta = 1$ compared to our 45, with similar convergence trends at $\Delta = 4$ and slight underperformance by $\Delta = 7$, where Stockfish averages 140 centipawns and our model 145.

4.4 Discussion

In general, our model performs better at lower Δ and lower ratings. This makes sense since the likelihood of sub-optimal play is inversely proportional to the rating. In general higher-rated players are more likely to play more optimal moves, which favors the base Stockfish evaluation. As Δ increases, both Stockfish and our model's performance are worse. This also makes sense as the further ahead we look into the game, the more variance we are exposed to. Overall, EIE demonstrates up to 10% increase in next-move prediction accuracy, and up to 20% increase in evaulation accuracy over existing chess engines. The 10% improvement in next-move prediction accuracy stems directly from the Maia-Chess models being more calibrated to games between players of lower ELO ratings. Because our methodology incorporates a more accurate next-move prediction than Stockfish in the context of lower ratings, we can see that it exhibits an improvement in evaluation accuracy, especially for lower values of delta.

We use Stockfish's evaluation as a proxy for the ground truth, our evaluations heavily depends on how close Stockfish can evaluate a position to the true theoretical evaluation (which is unknown). As such, Stockfish may make errors. Additionally, we cannot filter out games with players who cheat or play at a rating lower than their true rating, which could possibly disrupt our dataset quality. This could affect the resulting performance metrics. Regardless, our model performed better than the baselines we measured against and is a promising alternative to traditional chess engines for evaluating chess positions of games which are played between players of lower ratings.

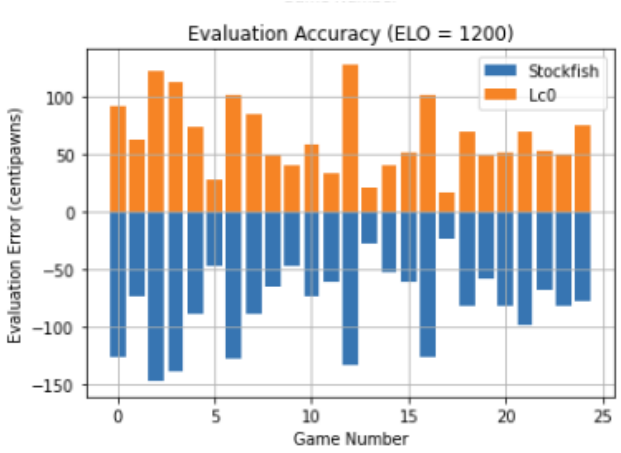


Figure 2: Evaluation accuracy for Stockfish and EIE-Lc0 from baseline truth across 25 games, using delta=1 for evaluation.

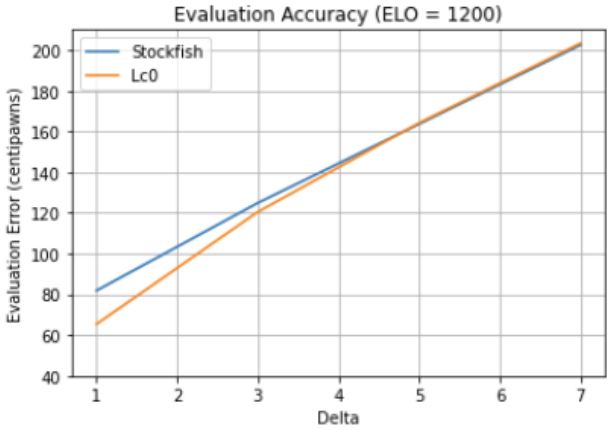


Figure 3: Evaluation accuracy for Stockfish and EIE-Lc0 from baseline truth across 7 values of delta.

References

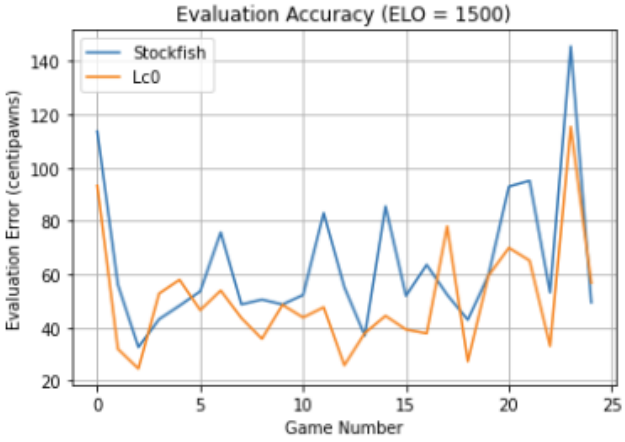
[1] The Stockfish developers (see AUTHORS file). *Stockfish*. 2004. URL: https://stockfishchess.org/.

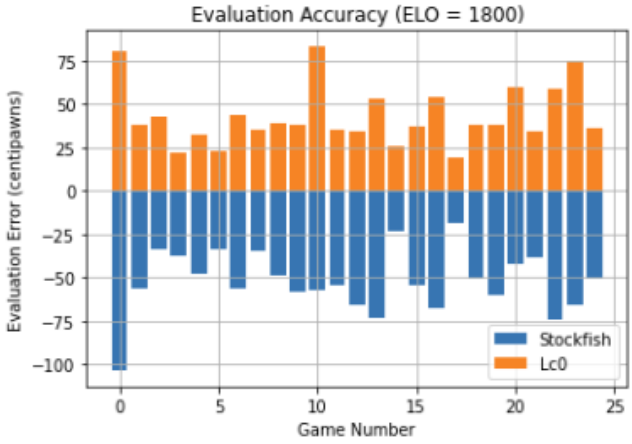
[2] Leela Chess Zero Contributors. *Leela Chess Zero*. Accessed: 2024-12-13. 2018. URL: https://lczero.org/.

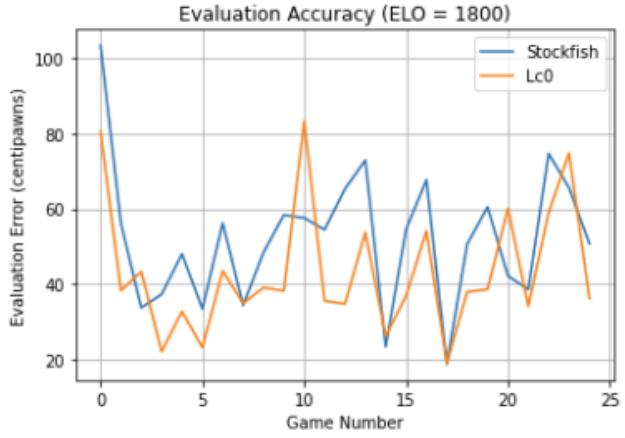
[3] Reid McIlroy-Young et al. "Maia: A Human-Like Chess Engine". In: *Proceedings of the Neural Information Processing Systems (NeurIPS)*. 2020. arXiv: 2006.01855 [cs.AI]. URL: https://arxiv.org/abs/2006.01855.

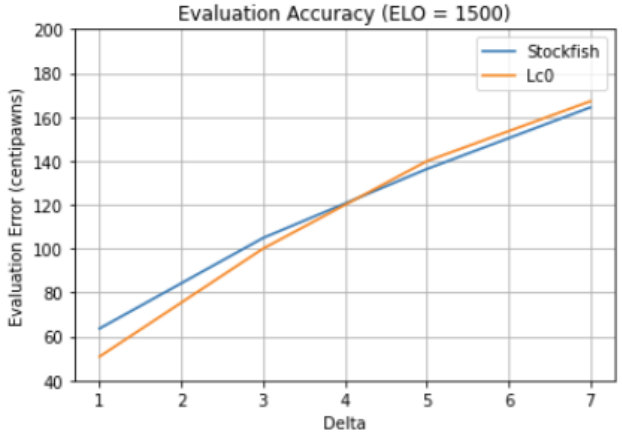
[4] Anian Ruoss et al. *Amortized Planning with Large-Scale Transformers: A Case Study on Chess*. 2024. arXiv: 2402.04494 [cs.LG]. URL: https://arxiv.org/abs/2402.04494.

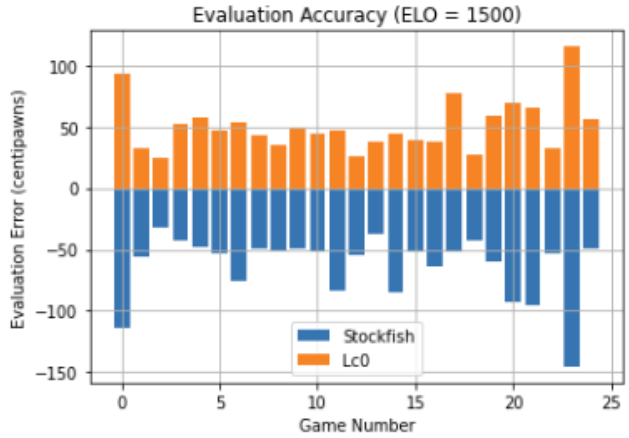
A Appendix

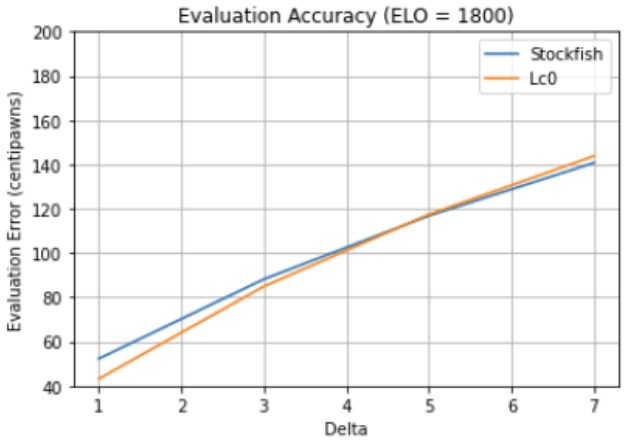

Evaluation Accuracy (ELO = 1500)
Stockfish
Lc0
Evaluation Error (centipawns)
Game Number


Evaluation Accuracy (ELO = 1800)
Evaluation Error (centipawns)
Stockfish
Lc0
Game Number


Evaluation Accuracy (ELO = 1800)
Stockfish
Lc0
Evaluation Error (centipawns)
Game Number


Evaluation Accuracy (ELO = 1500)
Stockfish
Lc0
Evaluation Error (centipawns)
Delta


Evaluation Accuracy (ELO = 1500)
Evaluation Error (centipawns)
Stockfish
Lc0
Game Number


Evaluation Accuracy (ELO = 1800)
Stockfish
Lc0
Evaluation Error (centipawns)
Delta

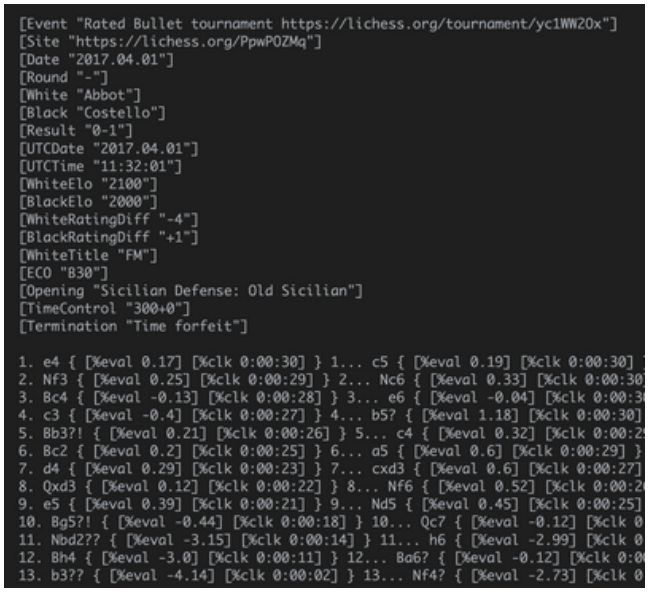


Figure 4: Example data from the Lichess Open Database.