Evaluating Deep Learning-based Chess-Engine Endgame Strategies

Epitácio Pessoa de Brito Neto[1], **Telmo de Menezes e Silva Filho**[2], **Thaís Gaudencio do Rêgo**[1], **Yuri Malheiros**[1]

[1]Informatics Center – Federal University of Paraíba (UFPB)
João Pessoa, Paraíba, Brazil

[2]Department of Engineering Mathematics – University of Bristol
Bristol, United Kingdom

epitaciobneto@gmail.com, yuri@ci.ufpb.br

Abstract. *Artificial Intelligence has been used to challenge human players in chess for decades. In 1997, IBM's Deep Blue won against the best chess player of that time, and since then, the chess engines have continued to improve. However, it is not clear if these new high-performing chess engines are learning how to replicate the way human grandmasters play or if they are devising new strategies to win. Therefore, in this paper, we evaluated two chess engines that use deep learning approaches: StockFish NNUE and Lc0, to compare their moves in endgame situations to the moves in chess theory books. We collected 19 types of endgames and we ran the engines to replicate the books' moves. After that, we computed the similarity, that is, the percentage of equal moves. The Lc0 engine has 40.20% similarity in our experiments and StockFish NNUE 22.50%. These results show that the engines replicate some moves from chess-theory books, but they differ in most parts from what is expected from human players.*

1. Introduction

Artificial Intelligence has been used to challenge human players in many games for decades. Chess is one of the most popular games researchers have been using software to play well and beyond human capabilities. In 1997, IBM's Deep Blue won against Garry Kasparov, the best chess player of that time. The development continued in the following years, and new improvements were achieved [Maharaj et al. 2022]. Deep Blue used a game tree search approach, but most recent chess engines have been using machine learning and neural networks [Silver et al. 2018].

Several chess engines have been created that surpassed the best human players [Acher and Esnault 2016]. The state-of-the-art engines have Elo ratings above 3,000. For instance, StockFish NNUE[1], and Leela Chess Zero (Lc0)[2] are popular open-source engines that use deep learning approaches with versions with 3,190 and 3,250 ratings, respectively. The Leela Chess Zero's neural network is based on the AlphaGo Zero [Silver et al. 2017] and AlphaZero [Silver et al. 2018] architecture. However, there are some changes. The main difference is that Lc0 uses a residual tower with squeeze and

[1]https://stockfishchess.org
[2]https://lczero.org

excitation [Hu et al. 2018] layers in its core. The architecture of Stockfish NNUE is different from the neural networks of the Lc0. It uses a shallow, fully connected network with four hidden layers.

In the beginning, researchers took inspiration from the players to create chess engines. However, recent neural network-based engines have employed machine learning approaches to train their agents [Liu et al. 2021]. Their results have shown that these new agents may learn to play in creative ways, which do not necessarily follow traditional and well-known strategies. Nowadays, many players are studying chess engines to improve their performance and understand the game better [McGrath et al. 2022].

In this paper, we aim to analyze the behavior of the engines StockFish NNUE and Lc0 in endgame situations. We collected 19 endgames from books and used these situations to compare the engines' moves with human players. Thus, we could check the similarities and differences between human players and chess engines.

2. Related Works

In [Haque et al. 2021], the authors evaluated the performance of the engine Lc0 in endgames. This work aimed to evaluate how different neural networks learn and play endgames. They analyzed the prediction of the next moves using neural networks that played at different levels and compared them with moves that humans should make. Also, the paper analyzes the prediction of the result of the game. The results showed that neural networks improved with longer training, and can play close to the perfect game.

[Fayed 2021] trained supervised machine learning algorithms, for instance, logistic regression, decision trees, and neural networks, to predict the result of an endgame situation. In the paper, the dataset used had examples of king and rook vs. king endgames. In their experiments, neural networks had an accuracy of 98%, decision trees 97%, and logistic regression 92%.

[Si and Tang 1999] studied three types of endgames using neural networks. The authors made specific neural networks for each type of endgame, therefore they considered the specificities of the problems in their models. However, each neural network in this paper has three layers (input, hidden, and output). The three endgames tested were: endgames with rookies, endgames with queens, and endgames with pawns. The neural networks was trained with 1,000 instances, and the authors showed that the network replicated the patterns by which they were trained.

3. Endgames

The exact moment when the endgame starts is not well defined [Whitaker et al. 1960]. However, we can think that the endgame is the moment of a game that the board has few pieces. The endgame strategies often include one to three pieces, excluding the king. Thus, the studies of endgames group strategies using combinations of these one-three pieces. In [De La Villa 2015], there are 11 types of endgames: basic endings; knight vs. pawn; queen vs. pawn; rook vs. pawn; rook vs. two pawns; same-colored bishops: bishop and pawn vs. bishop; bishop vs. knight: one pawn on the board; opposite-colored bishops: bishop and two pawns vs. bishop; rook and pawn vs. rook; rook and two pawns vs. rook and pawn endings. Figure 1 shows an endgame example.

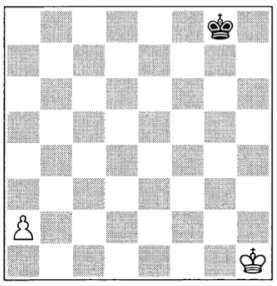

Figure 1. Endgame with king and pawn for the white and king for the black.

4. Elo Ranking

The Elo Ranking is a system used to measure the level of a chess player [Elo 1967]. The International Chess Federation (FIDE) started to use the Elo Ranking in 1970. The FIDE uses these values to select players for tournaments and to give them titles. For instance, a Grandmaster, the highest title for a player, requires an Elo rating above 2,300. After a game, the Elo ratings of both players are updated. The new value depends on the current rating of players and the game's result. The Equation 1 shows how a rating is updated.

$$r'(A) = r(A) + k(S_A - \mu_A) \tag{1}$$

Where $r(A)$ is the current rate of player A, and $r'(A)$ is his updated rate. The constant k defines the size of the updates. A small value means little updates and large values cause big changes. S_A is the result of the game, 1 means player A won, and -1 means he lost. Last, μ_a is the game's expected result when player A competes against player B. If A and B have the same rating, then $\mu_a = 0$. If $\mu_A > 0$, then we expect player A to win, otherwise, if $\mu_A < 0$ we expect player B to win. The range of μ_a is $[-1, 1]$.

5. Method

We collect endgame moves from classic games from two books [De La Villa 2015] [Polgar 2013]. The books show the moves in a graphic format and chess notation. Thus, we transcribed all endgame moves to the FEN (Forsynth-Edwards Notation) format to be able to use them in the engines. The final dataset for our analysis has 19 endgames.

In our experiments, we analyzed two open-source chess engines that use neural networks: StockFish NNUE and Lc0. We used StockFish NNUE with a 3,190 rating and Lc0 with a 3,250 rating. Also, the engines have a depth parameter. This parameter defines how many future moves the engine considers to decide the current move. To achieve our goals, we simulated games using the graphical interface BanksiaGUI[3]. Using the graphical interface, we can analyze all moves and see the board.

The following endgames were examined: (1) Rule of the square; (2) Key squares; (3) The Rook's pawn. Defending King in front of the Pawn; (4) Imprisoning the stronger side's King; (5) Rook vs. Bishop. The wrong corner; (6) Rook vs. Knight. At the edge of the board; (7) King + Knight checkmate; (8) The Knight's dumb square; (9) Queen vs. 7th-rank Rook's Pawn; (10) The strong King is near; (11) The Rook in front of the Pawn;

[3]https://banksiagui.com

Table 1. Simulations results using Lc0 and StockFish NNUE engine.

Endgame	Moves	Equal Moves		Similarity		Depth	
		Lc0	StockFish	Lc0	StockFish	Lc0	StockFish
1	9	3	5	33.33%	55.55%	2	8
2	9	9	2	100.00%	22.00%	2	4
3	2	2	2	100.00%	100.00%	2	2
4	9	6	8	67.00%	89.00%	2	6
5	13	8	0	62.00%	0.00%	6	2
6	13	9	3	69.00%	23.00%	4	2
7	3	3	3	100.00%	100.00%	3	2
8	3	2	1	67.00%	33.00%	2	2
9	10	8	0	80.00%	0.00%	3	2
10	9	8	0	89.00%	0.00%	2	2
11	10	1	0	10.00%	0.00%	2	2
12	19	1	6	5.00%	32.00%	2	2
13	9	1	4	11.00%	44.00%	2	3
14	7	1	1	14.00%	14.00%	2	9
15	13	12	7	92.00%	54.00%	5	9
16	9	1	1	11.00%	11.00%	6	5
17	9	0	0	0.00%	0.00%	2	2
18	25	5	1	20.00%	4.00%	4	7
19	19	1	1	5.00%	5.00%	2	2

(12) Special themes with a Knight's Pawn; (13) Strong side's King on one side of the Pawns; (14) Driving off the defending Bishop; (15) Central Pawn; (16) Pawns on the 6th rank; (17) The Philidor position; (18) Central Pawns; (19) Blocked Pawns. Key squares.

For each endgame situation, we used both engines to try to replicate the moves in the books. For instance, if the books show 9 moves, an engine makes 9 moves as well. Additionally, for each test with an engine, we varied its depth from 2 to 10. Last, with the engines' moves, we compare and check the similarity between the books and the engines. For each move, we compare if it is equal, in the same order, to the move in the books. Thus, the similarity is the percentage of equal moves.

6. Results

The Table 1 shows the results of the tests using the Lc0 and StockFish NNUE engines. The first column of the table shows the endgame type, and the second column shows the number of moves reported in the books. The following columns show the number of moves the engine made equal to the books, the similarity percentage, and the minimum depth that achieved the best result, respectively, for each engine.

The total number of moves in the dataset is 200. The Lc0 engine made 81 identical moves (40.50%), and the average depth used was 2.8947. In endgame 17, Lc0 did not match any move. In 3 tests (endgames 2, 3, and 7), the Lc0 made 100% of identical moves. The maximum depth used was 6 for the endgames 5 and 16. The depth level 2, the minimum used, appeared in 12 tests. The StockFish NNUe engine made 41 identical moves (22.50%), and the average depth used was 3.8421. The engine did not match any move in 5 tests (endgames 5, 9, 10, 11, and 17). In 2 tests (endgames 3 and 7), the StockFish engine made 100% identical moves. The maximum depth used was 9 for the endgames 14 and 15. The depth level 2 appeared in 11 tests. Notice that for endgame 17,

the Philidor position, both engines failed to reproduce the moves. In this endgame, the game ends in a draw if the players follow all correct moves. Therefore, the engines may try to deviate from the theory of the books, and then they may win the game.

Last, following or not following the moves of an endgame differs from being correct or wrong. A move or a sequence of moves deviating from the books may also achieve victory. Additionally, the Elo ratings of these engines are higher than the best human players. Thus, human players can learn when an engine plays differently to try to improve their performance. Also, researchers can learn when machines need to follow or deviate from humans, hence they can improve the engines even more.

7. Conclusion

This paper aims to compare how StockFish NNUE and Lc0 engines play endgames and the moves presented in chess books. The results show that, besides the engines having greater Elo ratings than the best human players, they have some similarities in their moves. The Lc0 has 40.50% similarity in its moves, and StockFish NNUE has 22.50%.

It is worth highlighting that similarities or differences do not mean good or bad moves. Analyzing the results, we can conclude that the engines follow some human moves in the endgames, but they differ for the most part. This result gives us two ways for future studies. First, because engines have high Elo ratings, human players can learn from them to improve their games. Second, researchers can analyze if the similarities are creating worse or better results for the engine, so improving the current software.

To better understand the engines, testing more endgames and other phases, such as openings and middlegames, would be interesting. Also, a detailed qualitative study of some examples can clarify and assist the interpretation of engines' moves. Further, we can measure similarity to players or classic games because the players do not necessarily follow all book moves. Another significant analysis is to compare other Elo ratings for the engines. Both StockFish and Lc0 have versions of different ratings. Last, there are other engines that we can include in future studies, for instance, Maia Chess.

References

Acher, M. and Esnault, F. (2016). Large-scale analysis of chess games with chess engines: A preliminary report. *arXiv preprint arXiv:1607.04186*.

De La Villa, J. (2015). *100 Endgames You Must Know: Vital Lessons for Every Chess Player Improved and Expanded*. New In Chess.

Elo, A. E. (1967). The proposed uscf rating system. its development, theory, and applications. *Chess Life*, 22(8):242–247.

Fayed, M. S. (2021). Classification of the chess endgame problem using logistic regression, decision trees, and neural networks. *arXiv preprint arXiv:2111.05976*.

Haque, R., Wei, T. H., and Müller, M. (2021). On the road to perfection? evaluating leela chess zero against endgame tablebases. In *Advances in Computer Games*, pages 142–152. Springer.

Hu, J., Shen, L., and Sun, G. (2018). Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141.

Liu, S., Cao, J., Wang, Y., Chen, W., and Liu, Y. (2021). Self-play reinforcement learning with comprehensive critic in computer games. *Neurocomputing*, 449:207–213.

Maharaj, S., Polson, N., and Turk, A. (2022). Chess ai: competing paradigms for machine intelligence. *Entropy*, 24(4):550.

McGrath, T., Kapishnikov, A., Tomašev, N., Pearce, A., Wattenberg, M., Hassabis, D., Kim, B., Paquet, U., and Kramnik, V. (2022). Acquisition of chess knowledge in alphazero. *Proceedings of the National Academy of Sciences*, 119(47):e2206625119.

Polgar, L. (2013). *Chess: 5334 problems, combinations and games*. Hachette UK.

Si, J. and Tang, R. (1999). Trained neural networks play chess endgames. In *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)*, volume 6, pages 3730–3733. IEEE.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., Driessche, G. v. d., Graepel, T., and Hassabis, D. (2017). Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359.

Whitaker, N., , and Hartleb, G. (1960). *365 Selected Endings, one for each day of the year*. Self published.