Natural Language Processing in Chess: A Comprehensive Exploration of the Abilities of Language Models in Game-Playing

Mateusz Tabaszewski **Bartłomiej Pukacki** **Krzysztof Weber** **Adam Mielniczuk**

Abstract

Recent developments in the field of natural language processing (NLP) in tasks requiring logical thought, like those of question answering[1] and code generation[2], have forced researchers to consider the limitations and abilities of such models in tasks related to game-playing. Games have been used to test human logical abilities in a predefined setting with a limited set of actions for thousands of years[3]. As such, it makes sense to extend this challenge to the currently available NLP models to assess their logical limitations and to compare them with those exhibited by humans. In this paper, we would like to explore the abilities of NLP models for the task of game-playing using chess as a means to study these models' abilities. Chess was chosen thanks to its popularity and ready-made implementations which allow for a more comprehensive overview of the model's abilities thanks to the existence of solutions trained specifically for chess[4].

Introduction

The following paper is meant to be an overview of multiple aspects related to NLP models used for playing chess. As such, each section presented below can be treated in large part independently with its own focus, experiments, and conclusions, however, the final conclusions have also been provided at the end. An additional section at the start was also dedicated to the exploration of human chess-playing abilities to allow for further comparison with the NLP model's abilities.

Exploration of Human Chess Games

Assessing the NLP model's performance in the context of game-playing requires a good understanding of what differentiates a good chess player from a bad player. This task may require a good definition of metrics and comprehensive visualizations. Of course, the most efficient way would be to compare the models by having them play against increasingly higher-rated human players. However, this may not be feasible due to the number of games that would be required of the human players to accurately compare the abilities of NLP models. Furthermore, the small scale of our

study did not allow us to gather a sufficient number of highly rated players for this comparison. As such, to better understand the task of automatically exploring the model's abilities and to better understand the means of evaluating the quality of moves made by a player, we have turned to the recordings of human games publicly available on the Lichess website[5].

Our code and the corresponding data analysis were based on the datasets from August 2013, February 2013, and January 2013 for Train, Test, and Val datasets respectively although the approach could be generalized for newer datasets available on the website. Furthermore, we only focused on a subset of 20,000 games for the training set and 1000 games for test and validation sets. This was necessitated by the volume of the data making it hard to process quickly and efficiently for our study, however we believe that this number of games should be more than enough seeing as each game is composed of multiple moves, leading to a dataset composed of hundreds of thousands of examples. From this dataset, we have utilized the information about the state of the board represented using Forsyth–Edwards Notation (FEN), which move was just chosen causing the board to end up in this state, the legality of the move as "True" or "False" (more on this later), stockfish_2, stockfish_5, stockfish_10 as Stockfish engine's evaluations[6] for depths 2, 5, 10 respectively; move_quality_2, move_quality_5, move_quality_10 as the difference between the previous stockfish evaluation of a given depth and the new evaluation after performing a move (i.e. how beneficial was the move for the player); ELO for both players and lastly whether the move was real or randomly generated (more on this later in this section). Furthermore, we have also parsed the information from the FEN representation into substrings representing things like piece placement, en_passant, and other game information. For a more detailed description of the available data at this stage please see the project's repository[7].

Furthermore, to better understand moves made by humans and for the sake of the task of legal/illegal move classification described in the next section, we have enriched the dataset by simulating a random move from the uniform distribution of moves for every real move made. This way we obtained a parallel "what if" scenario for a move done at random. The information if the move is real or simulated in our data is presented in our dataset along with the legal attribute showcasing if our simulated move is legal or only pseudo-legal. Pseudo-legal moves involve moving an existing figure but doing so in such a way that is usually not allowed in chess, like a knight going straight, etc. Furthermore, we cannot estimate Stockfish scores for illegal moves, but random moves also include legal moves, so we can base our calculations on those as well. Lastly, it is important to note that the dataset was approximately balanced in terms of the number of real vs random moves as well as legal and illegal moves.

One of the best ways of achieving an estimation of a chess player's ability is the stockfish score estimation of the positions these players find themselves in, at least

in comparison to the position they would have found themselves in, had they played randomly. As such, below we can see a visualization of both the move quality (Figure 1) as a difference between stockfish score before and after and the position quality itself (Figure 2) for both human and random moves. Humans consistently perform better than random players, though most players seem to find themselves in a somewhat worse position than before according to StockFish.

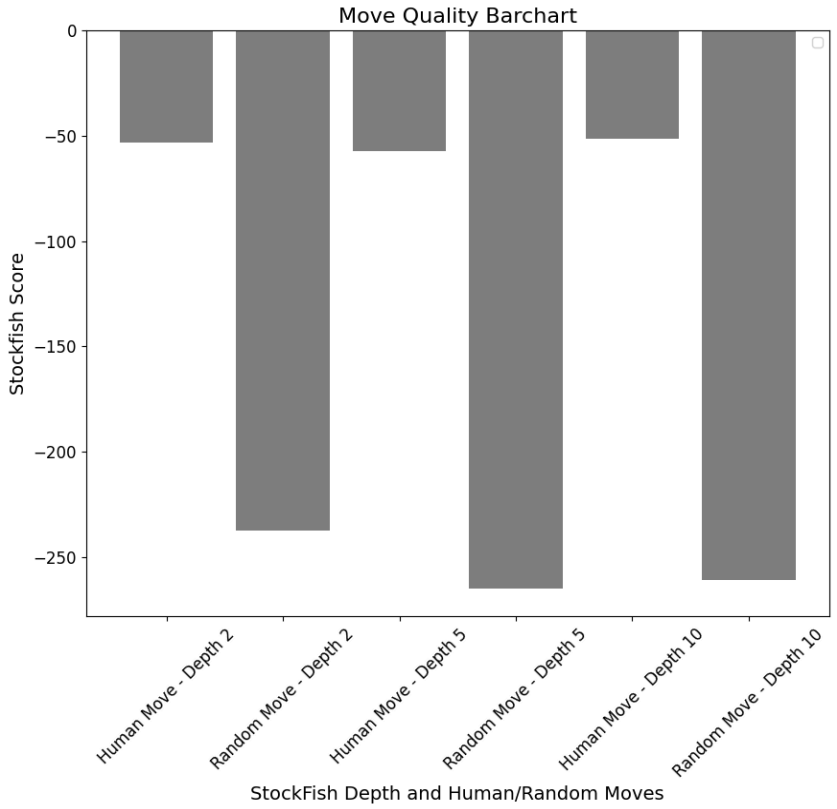

Figure 1. Bar chart showing the move quality as judged by the Stockfish engine for both human and random moves for different search depths for the Stockfish engine.

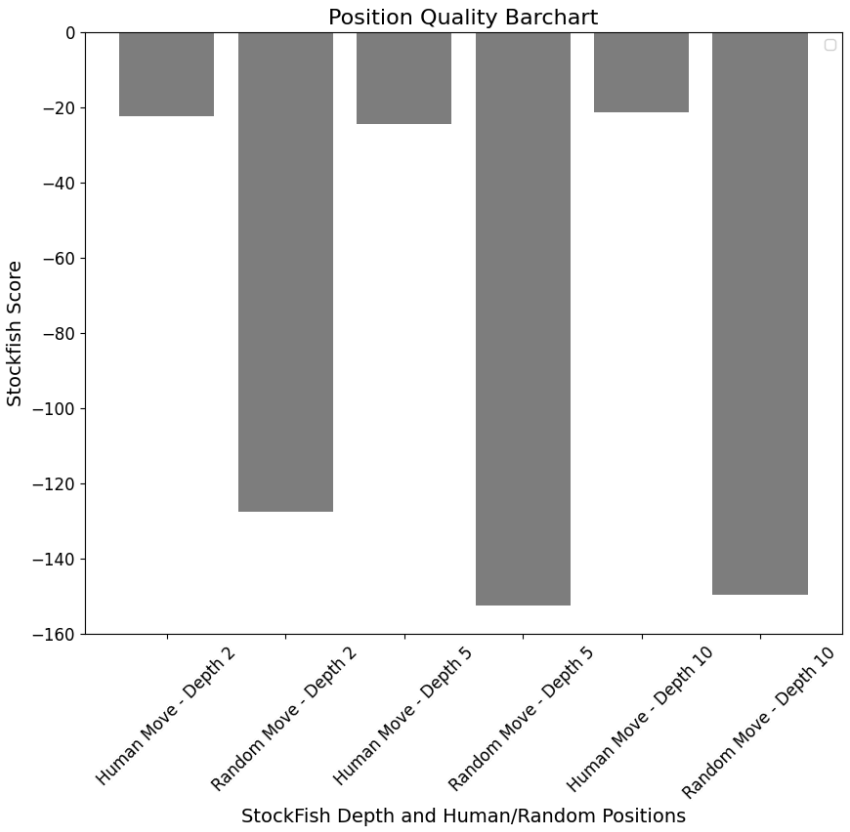

Figure 2. Bar chart showing the quality of the position the player has found themselves in as judged by the Stockfish engine for both human and random players for different search depths for the Stockfish engine.

The distribution of moves' and positions' quality estimation using the Stockfish engine comparing random and human players can be seen in Figure 3 and Figure 4 respectively. Humans tend to perform fewer very bad moves and the distribution seems more concentrated.

Figure 3. Violin plot comparing the move quality, as judged by the Stockfish engine for both human and random players. Each of the subplots shows a different Stockfish engine depth.

Figure 4. Violin plot comparing the chess position quality the player has found themselves in, as judged by the Stockfish engine for both human and random players. Each of the subplots shows a different Stockfish engine depth.

Furthermore, it can be beneficial to understand which moves the players perform most frequently within our dataset. As such Figure 5 showcases these most common moves from all human players within our dataset. As can be seen, move g1f3 is particularly popular with e2e4 and g8f6 all being within the top 3 most commonly detected moves. All of those moves correspond to the most common chess opening moves (king's knight, king's pawn openings for white, and the black's Indian defense respectively).

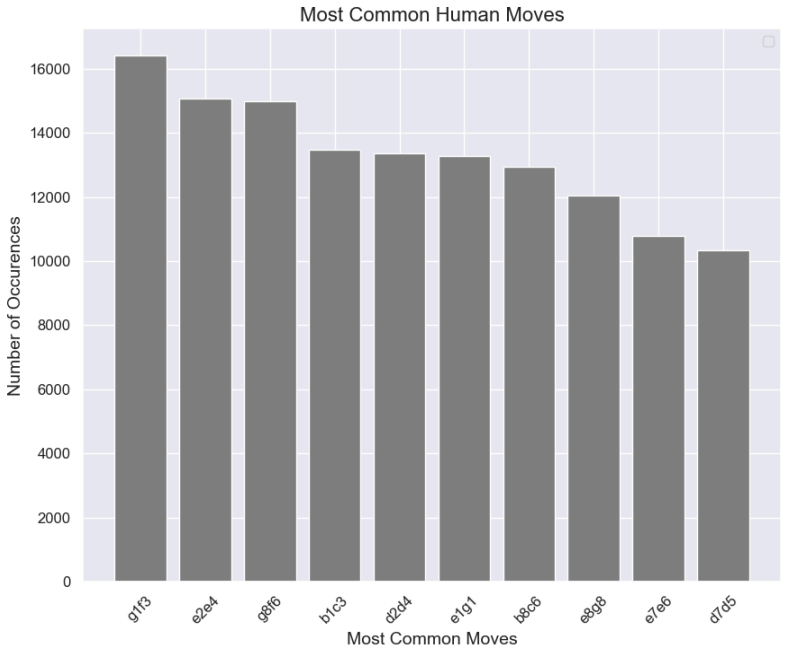

Figure 5. Histogram showing the most common moves done by humans in chess.

Lastly, we have decided to discretize the players into four categories: novices, intermediate players, expert players, and master players. The binarization was done using the following ELO thresholds: 0 - 1000 for novices, 1000 - 1500 for intermediate players, 1500 to 2000 for expert players, and 2000+ for masters. Using this binarized data we created a plot showcasing move quality as judged by the most common moves for each player depending on the experience level (Figure 6). As can be seen, the playstyle changes depending on the player's experience level. Master-level players tend to perform moves that give negative evaluation by Stockfish but, their moves appear to still have better Stockfish evaluation than most common moves by novices, with the exception of g7g6 which is the only move to be assessed positively by Stockfish across all common moves for all player's experience levels. Regardless, we believe that this visualization has showcased that the Stockfish score can provide important information about the player's abilities.

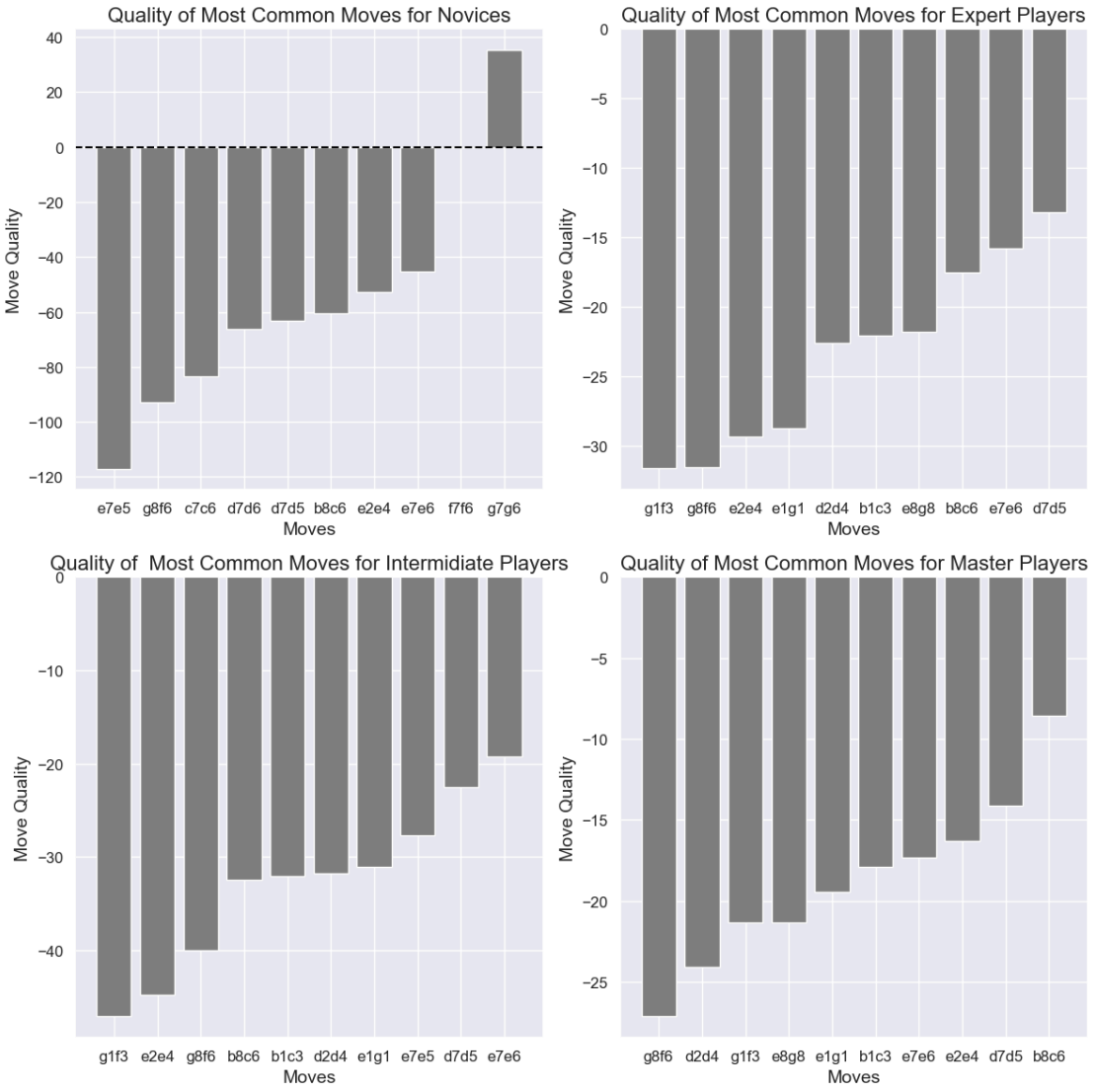

Figure 6. Bar charts showing the most common moves in chess by humans based on experience level judged by the player's ELO and the move's quality as judged by the Stockfish engine.

In conclusion, the presented figures seem to indicate that players tend to pick moves that cause a negative Stockfish evaluation. However, there is a difference between moves picked by novices as opposed to those picked by masters. As such, the Stockfish scores end up reflecting it by showing more favorable scores for moves frequently performed by masters. As such, we believe that we have presented that metrics related to stockfish score and picked moves can be beneficial in assessing both a human's and an NLP model's chess-playing abilities.

BERT for Move Legality Classification

It is important to test if NLP models are capable of understanding the internal rules governing chess. Although nowadays with the rise of popularity of websites created for playing chess, knowing how a certain figure moves is not as crucial due to the impossibility of performing an illegal move, in reality knowing the rules of chess is crucial to how humans plan their games, test if a certain move is beneficial (i.e. it is not a blunder) and more. As such, we have set out to fine-tune a BERT[15] model for the task of classifying certain moves as legal or illegal. The dataset used for this task was described in more detail in the previous section. Now we will only mention that our dataset includes both human and random moves with approximately the same number of human and random moves and approximately the same number of legal/illegal moves. As such, only a small percentage of random moves was legal but we believe that it was enough to accurately assess the data in the previous section. Regardless, fine-tuning BERT[15] for chess was already done before[8] although not specifically for the task of classifying the legality of moves. The model was presented with an FEN representation of the board and a move to be performed (also as a string), as output we expected the class to correspond to a legal or illegal move. Furthermore, this required some changes to the tokenization process, however, more details are available in the project's repository[7].

The training process forced us to restrict the data to only 10000 for the training process, and 500 examples for both validation and test set, this was done due to computational restrictions, however, as we will show later, a small dataset in this case still allowed our model to achieve satisfying results. The training was done on the Google Colab platform with the use of CUDA GPU for 40 epochs. The loss used was binary cross entropy. The optimizer used was AdamW[9] with the learning rate set to 1e-5. At the end of the training process, the model managed to achieve an accuracy of **98.67%**, **88.6%**, and **85.0%** for training, validation, and test sets respectively.

In conclusion, the BERT[15] model managed to obtain a performance far above the random baseline of 50%, showcasing a limited ability to represent the possible moves made by figures on the chessboard, as such we believe that NLP models may be capable of achieving good game-playing rule understanding when provided

enough examples. However, future research in this section could focus on further improving the model's performance by providing more examples or trying to create more difficult adversarial examples of non-obvious illegal moves.

NLP for Chess-Playing

As we have managed to prove in the previous section, NLP models can gain a limited understanding of the rules of chess, which we tested by forcing a BERT[15] model to learn to classify the moves as legal or illegal, as such the next step is to evaluate the NLP model's chess-playing abilities by making them play the game and assessing their abilities in this context. In this section, we focus on the GPT-2[10], GPT-2-large, and chessGPT[4] models and compare their performance for the task of chess-playing.

For the task, a small subset of a second dataset[11] was used. The dataset contained sequences of moves in SAN notation which is a simpler and more compact notation than FEN which could help the model in providing adequate output. The quality of moves in the dataset was evaluated using Stockfish to determine the difference in scores between the current state and all possible legal moves.

The task of the model was to output the next possible move in the sequence according to the prompt: "*Provide the next move in the chess game. Only provide the move, no move numbers. {move sequence}"*. The output was filtered assuming the beginning sequence of tokens before a space should be the next move in the game and checked for validity and quality. Tests were performed multiple times for a small subset of games at various prior game sequence lengths i.e. after 5, 10, 15, and 20 moves. The number of games and model answers considered depends on the hardware availability of the student performing the task.

Additionally, a separate test was performed for each model with a custom transformers.TextGenerationPipeline pipeline, forcing the model to produce output with tokens included in theoretical legal moves in an effort to help the model produce valid moves. In this task, a single answer to a higher number of games was evaluated due to generating output from a probability distribution not being possible with this setup. It is important to note that the below results for human play may not be completely reliable due to the small number of samples of human gameplay for this particular series of experiments. However, we still believe that the results effectively showcase the differences between human and model-based chess gameplay.

GPT-2

Complete results can be viewed in the evaluate_gpt2.ipynb notebook at our repository. In the figure below (Figure 7) we can see a result of comparing the GPT-2 model to a randomly picked legal move (Figure 8) score quality aggregated from 20 games after performing 5 moves and evaluating 200 answers for the next move. In the tables (Table 1, Table 2) we are presenting statistics from the experiments described in the above section.

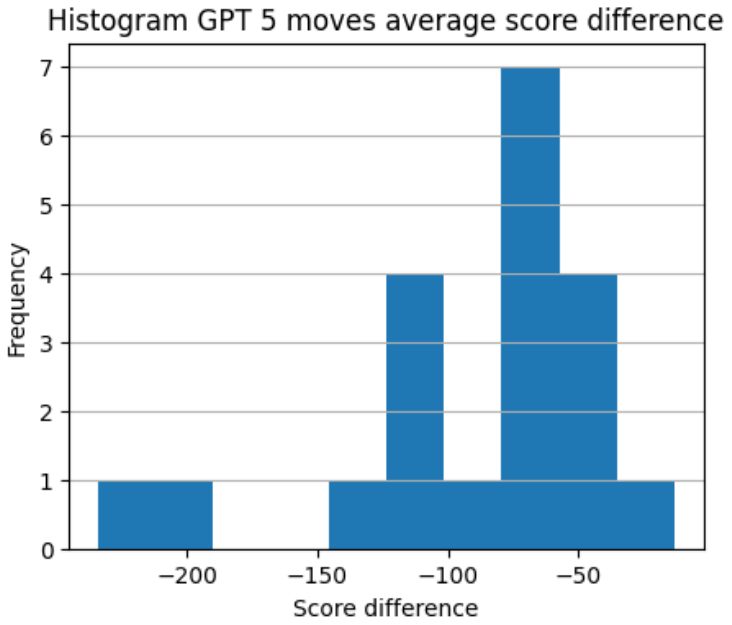

Figure 7. Histogram of quality of moves generated by the model. The result represents the number of games for which the average score difference of all legal moves out of 200 evaluated next moves fell into an appropriate range.

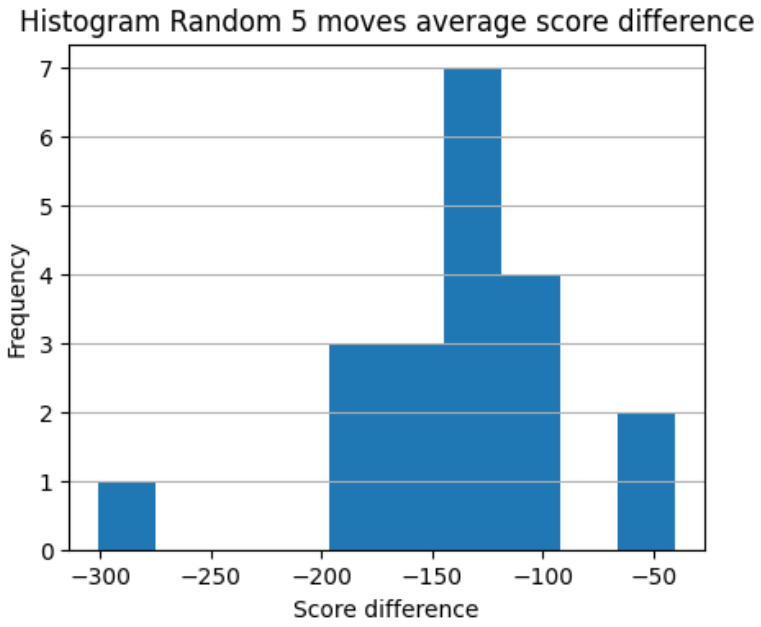

Figure 8. Histogram of quality of moves generated by the model. The result represents the number of games for which the average score difference of *X* randomly selected legal moves fell into an appropriate range, where *X* is the number of legal moves made by the model.

No. prev. moves	Player	Average score difference	% legal moves	% best moves	% above average moves	% worst moves
5	GPT-2	-88.88	30.52%	8.43%	86.07%	0.90%
	Random	-138.15	-	3.85%	73.62%	3.03%
	Human	-29.00	-	25.00%	95.00%	0.00%
10	GPT-2	-203.07	25.63%	7.31%	73.66%	0.98%
	Random	-265.06	-	2.83%	61.4%	2.34%
	Human	-72.55	-	15.00%	95.00%	0.00%
15	GPT-2	-242.67	18.19%	5.65%	63.81%	1.37%
	Random	-270.49	-	2.14%	57.70%	3.05%
	Human	-77.67	-	38.89%	94.44%	0.00%
20	GPT-2	-323.55	14.89%	5.78%	54.48%	1.49%
	Random	-365.89	-	3.73%	52.24%	3.36%
	Human	-92.39	-	16.67%	88.89%	0.00%

Aggregate results	GPT-2	-214.54	22.77%	6.76%	72.78%	1.10%
	Random	-259.90	-	3.09%	62.64%	3.00%
	Human	-67.90	-	22.39%	93.42%	0.00%

Table 1. Comparison of total aggregate results for GPT-2, Random, and Human players across different game sequence lengths. The number of moves evaluated for the Random player is equal to the total number of legal moves of the model, the number of Human moves is just one move per game, meaning a total of 20 per previous move sequence length.

No. prev. moves	Player	Average score difference	% legal moves	% best moves	% above average moves	% worst moves
5	GPT-2 ALT	-47.35	34.69%	11.76%	94.12%	0.00%
	GPT-2	-83.56	32.65%	0.00%	87.50%	0.00%
	Random	-157.23	-	0.00%	76.47	5.89%
	Human	-18.24	-	29.41%	100.00%	0.00%
10	GPT-2 ALT	-	0.00%	-	-	-
	GPT-2	-90.83	12.24%	0.00%	100.00%	0.00%
	Random	-	-	-	-	-
	Human	-	-	-	-	-
15	GPT-2 ALT	-134.45	23.91%	0.00%	90.91%	0.00%
	GPT-2	-191.54	23.91%	9.09%	81.82%	0.00%
	Random	-192.54	-	0.00%	81.82%	9.09%
	Human	-59.09	-	18.18%	100.00%	0.00%
20	GPT-2 ALT	-59.5	4.55%	0.00%	100.00%	0.00%
	GPT-2	-263.14	15.91%	0.00%	71.42%	0.00%
	Random	-75.0	-	0.00%	100.00%	0.00%
	Human	-105.5	-	0.00%	50.00%	0.00%
Aggregate results	GPT-2 ALT	-80.436	15.96%	6.66%	93.33%	0.00%

	GPT-2	-157.271	21.28%	2.5%	85.00%	0.00%
	Random	-141.594	-	0.00%	80.00%	6.66%
	Human	-60.942	-	23.33%	96.67%	0.00%

Table 2. Comparison of total aggregate results for GPT-2 with forced legal move alternative tokens (GPT-2 ALT), GPT-2, Random and Human players across different game sequence lengths. For the Random and Human players, only the games where the GPT-2 ALT model has made a legal move were evaluated.

According to the presented results the rather simple for today's standards and not fine-tuned GPT-2 model is consistently better in the quality of moves made than the Random choice player. Although the number of legal moves is not high (30% falling to 14% for later parts of the game), the results are still better than expected. The better quality of moves might be explained by the inability of the model to explore a higher number of possible moves. Still, the results of this model are much worse than the ones of human players which was expected.

Forcing legal moves alternatives using the *force_words_ids* argument in the pipeline handling the text generator unfortunately does not seem to help with finding legal moves. This is due to the fact that the default tokenizer is not appropriate for the task and splits the move string into separate elements which seems to make it harder for the model to create valid moves than if this limitation did not exist. Despite that, in the limited experiment, we can observe that the quality of moves could be better at times than that of the generic GPT-2 model however we would not consider these results very reliable and treat them rather as possible research possibilities in the future experiments.

GPT-2 - Large

In further experiments, we have also set out to compare the GPT-2 - Large model to see how the size of the model impacts its quality in the task of chess-playing. GPT-2 - Large has around 774M parameters compared to GPT-2's 124M parameters. We would like to check if this substantial difference in the number of parameters will impact the model significantly in terms of its abilities relating to chess. The results for the GPT-2 - Large model are available in the evaluate_gpt2-large.ipynb notebook from the project's repository[7]. The following Figures 9 and 10 show the results of our analysis, however, a more comprehensive comparison can also be seen in Table 3 and Table 4. Please note, that unless explicitly mentioned otherwise, the means of evaluating the model were the same as in the previous section.

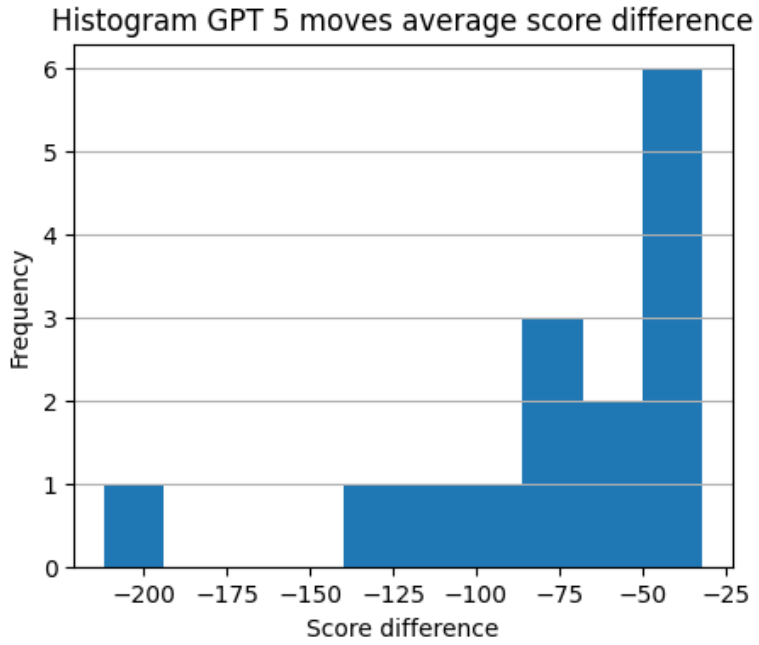

Figure 9. Histogram of quality of moves selected by the model for GPT-2 - Large. The result represents the number of games for which the average score difference of all legal moves out of 200 evaluated next moves fell into an appropriate range.

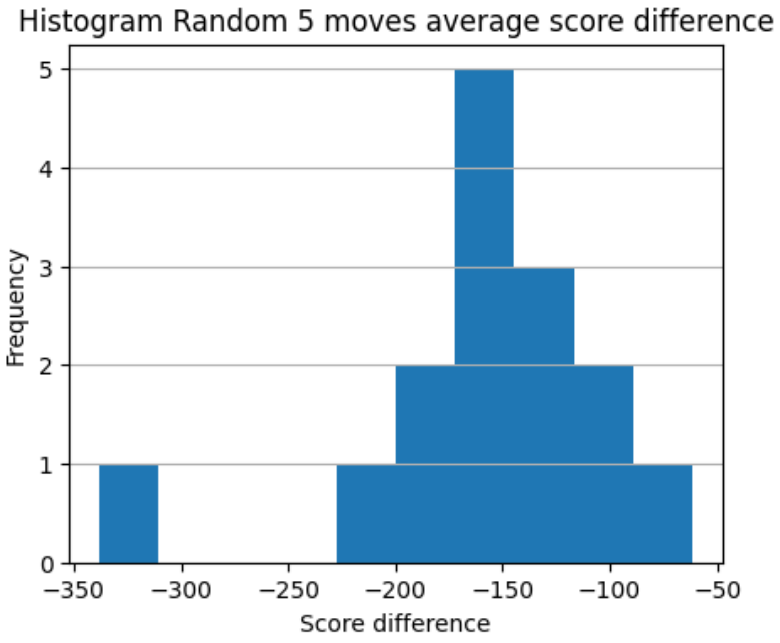

Figure 10. Histogram of quality of moves selected by the model. The result represents the number of games for which the average score difference of *X* randomly selected legal moves fell into an appropriate range, where *X* is the number of legal moves made by the model.

No. prev. moves	Player	Average score difference	% legal moves	% best moves	% above average moves	% worst moves
5	GPT-2 - Large	-74.15	59.1%	13.53%	91.32%	0.00%
	Random	-159.91	-	3.83%	71.36%	2.15%
	Human	-31.6	-	26.67%	100.00%	0.00%
10	GPT-2 - Large	-199.52	48.33%	9.10%	75.86%	0.96%
	Random	-269.36	-	2.62%	61.38%	2.76%
	Human	-46.33	-	20.00%	100.00%	0.00%
15	GPT-2 - Large	-235.78	27.15%	8.50%	64.02%	1.98%
	Random	-278.94	-	5.10%	51.27%	2.55%
	Human	-84.08	-	38.46%	100.00%	0.00%
20	GPT-2 - Large	-344.27	24.92%	2.16%	59.56%	0.92%
	Random	-368.23	-	3.08%	49.07%	3.40%
	Human	-105.46	-	15.38%	92.30%	0.00%
Aggregate results	GPT-2 - Large	-213.43	40.87%	9.74%	77.72%	0.74%
	Random	-269.11	-	3.53%	61.95%	2.57%
	Human	-66.87	-	25.00%	98.21%	0.00%

Table 3. Comparison of total aggregate results for GPT-2 - Large, Random, and Human players across different game sequence lengths. The number of moves evaluated for the Random player is equal to the total number of legal moves of the model, the number of Human moves is just one move per game, meaning a total of 20 per previous move sequence length.

No. prev. moves	Player	Average score difference	% legal moves	% best moves	% above average moves	% worst moves
5	GPT-2 - Large ALT	-22.23	34.69%	5.88%	100.00%	0.00%

	GPT-2 - Large	-60.59	65.31%	15.62%	90.62%	0.00%
	Random	-96.94	-	5.88%	76.47%	0.00%
	Human	-27.82	-	35.29%	94.11%	0.00%
10	GPT-2 - Large ALT	-188.00	10.20%	0.00%	100.00%	0.00%
	GPT-2 - Large	-144.18	44.90%	9.09%	81.82%	0.00%
	Random	-228.4	-	0.00%	60.00%	0.00%
	Human	-7.6	-	20.00%	100.00%	0.00%
15	GPT-2 - Large ALT	-128.64	30.43%	14.28%	85.71%	0.00%
	GPT-2 - Large	-137.67	26.08%	8.33%	83.33%	0.00%
	Random	-310.92	-	7.14%	50.00%	7.14%
	Human	-61.28	-	21.42%	100.00%	0.00%
20	GPT-2 - Large ALT	-163.25	9.09%	0.00%	75.00%	0.00%
	GPT-2 - Large	-199.3	22.73%	0.00%	70.00%	10.00%
	Random	-89.0	-	0.00%	75.00%	25.00%
	Human	-48.75	-	0.00%	100.00%	0.00%
Aggregate results	GPT-2 - Large ALT	-125.532	21.28%	7.50%	92.50%	0.00%
	GPT-2 - Large	-135.436	40.42%	10.53%	84.21%	1.31%
	Random	-181.317	-	5.00%	65.00%	2.50%
	Human	-36.365	-	25.00%	97.50%	0.00%

Table 4. Comparison of total aggregate results for GPT-2 - Large with forced legal move alternative tokens (GPT-2 - Large ALT), GPT-2 - Large, Random, and Human players across different game sequence lengths. For the Random and Human players, only the games where the GPT-2 - Large ALT model has made a legal move were evaluated.

The presented results showcase that the GPT-2 - Large model consistently achieves better results than a random player. Furthermore, it appears to achieve slightly better results than the considerably smaller GPT-2 model. Of course, to definitively prove

that the number of parameters positively impacts the model's chess-playing abilities more detailed studies would need to be performed, which could be a target of future research in the area. However, as it stands right now, it appears that there might be a positive link between the number of parameters and the model's ability to play chess as evaluated using the Stockfish engine. However, the increase of parameters necessary to reflect in the model's performance seems quite high, based on the numbers of parameters between GPT-2 and GPT-2 - Large models. Furthermore, forcing alternatives on the GPT-2 - Large model did not improve the model's ability to choose legal moves, however, it did improve the model's evaluation according to StockFish.

ChessGPT

Lastly, it might be beneficial to test if solutions specifically trained on chess-related data might be able to outperform the previously tested models and possibly even human players. As such in this section, we have utilized the chessGPT model and compared it against human and random players to test the model's abilities. The results have been obtained by means similar to the previous models and have been presented in Figure 11, as well as Table 5. However, one notable exception this time was the lack of comparison with a human player, as such for the sake of comparison results from previous comparisons have been used for the human player, since the results should be analogous.

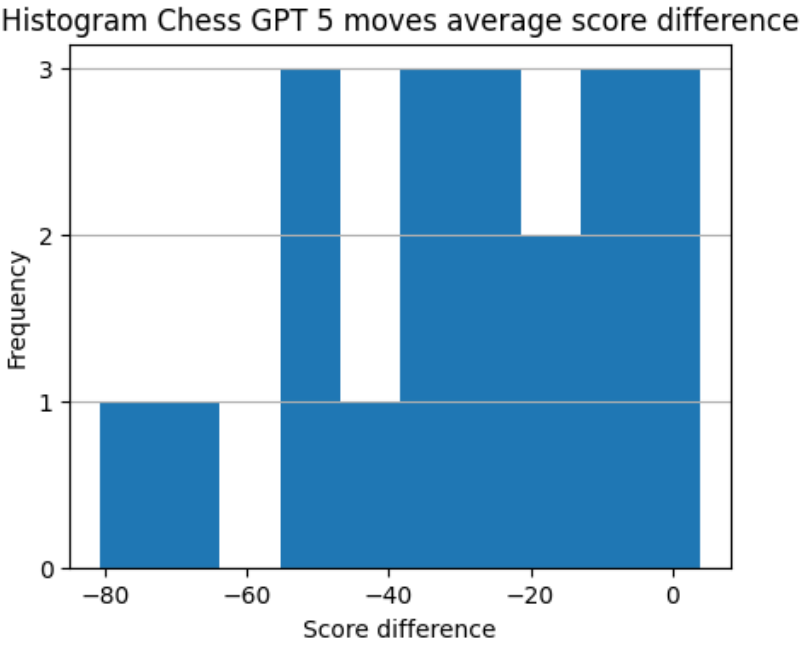

Figure 11. Histogram of quality of moves selected by the model chessGPT. The result represents the number of games for which the average score difference of all legal moves out of 200 evaluated next moves fell into an appropriate range.

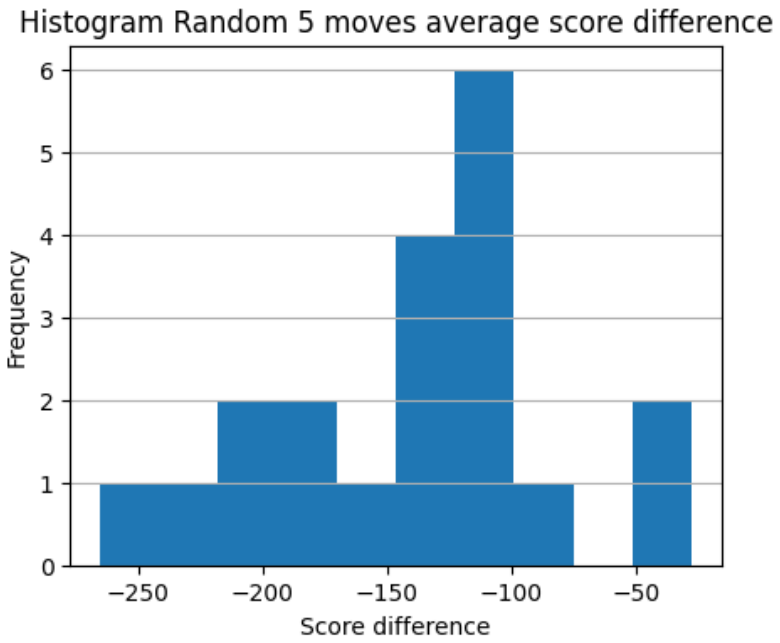

Figure 12. Histogram of quality of moves selected by the model. The result represents the number of games for which the average score difference of *X* randomly selected legal moves fell into an appropriate range, where *X* is the number of legal moves made by the model.

No. prev. moves	Player	Average score difference	% legal moves	% best moves	% above average moves	% worst moves
5	chessGPT	-28.35	98.00%	41.07%	97.44%	0.51%
	Random	-138.06	-	2.29%	72.44%	3.06%
	Human	-26.85	-	35.00%	100.00%	0.00%
10	chessGPT	-47.70	99.5%	26.88%	97.99%	0.00%
	Random	-249.01	-	3.02%	63.07%	3.27%
	Human	-76.00	-	20.00%	95.00%	0.00%
15	chessGPT	-102.12	98.61%	31.55%	92.39%	0.00%
	Random	-277.78	-	2.82%	55.77%	2.25%
	Human	-89.67	-	44.44%	88.89%	0.00%
20	chessGPT	-101.49	99.17%	26.89%	93.27%	0.00%
	Random	-358.71	-	4.48%	49.58%	4.76%
	Human	-90.00	-	22.22%	94.44%	0.00%

Aggregate results	chessGPT	-69.92	98.82%	31.69%	95.41%	0.13%
	Random	-255.89	-	3.13%	60.59%	3.33%
	Human	-70.63	-	30.26%	94.74%	0.00%

Table 5. Comparison of total aggregate results for chessGPT, Random, and Human players across different game sequence lengths. The number of moves evaluated for the Random player is equal to the total number of legal moves of the model, the number of Human moves is just one move per game, meaning a total of 20 per previous move sequence length

The results show that a language model trained on chess data can achieve much better results than a random player. The presented chessGPT[4] model has managed to achieve a performance comparable to or sometimes even exceeding the human level. This was reflected in both the Stockfish[6] engine's evaluation as well as percentages of best-picked moves. However, it must be stated that sometimes the model did perform illegal moves but the percentage of illegal moves in comparison to the legal ones performed by the model during the testing process has remained very low throughout the whole experiment.

Summary

The conducted experiments aimed to discover the difference between the human, random, and large language model players. All tested language models outperformed the random players by a significant margin. However, both GPT-2 and GPT-2 - Large were much worse than the human players. On the other hand, the chessGPT model has managed to match and sometimes even outperform the human players from the available dataset. This shows that NLP models trained on chess data can significantly outperform more general models. Furthermore, to compare the models more thoroughly we have created Table 6, which shows the aggregated results for all three models. As can be seen, chessGPT significantly outperforms the remaining models with GPT-2 and GPT-2 - Large having relatively similar scores, although when taking the percentage of legal moves into consideration, GPT-2 - Large can be considered a better model for generating chess moves, as such hinting that there might be a link between number of parameters and model's performance even for models not explicitly focusing on chess-based data. However, more detailed research, possibly including the current state-of-the-art models would need to be conducted to prove the link.

	Player	Average score difference	% legal moves	% best moves	% above average moves	% worst moves
Aggregate results	GPT-2	-214.54	22.77%	6.76%	72.78%	1.10%
	GPT-2 - Large	-213.43	40.87%	9.74%	77.72%	0.74%
	chessGPT	-69.92	98.82%	31.69%	95.41%	0.13%

Table 6. Comparison of total aggregate results for GPT-2, GPT-2 - Large, and chessGPT models for chess-playing.

NLP Models for Opening Recognition

The final set of experiments performed in assessing the abilities of NLP models for playing chess was related to the model's ability to recognize chess openings. This experiment was performed based on the chessGPT[4] model. To appropriately test the model's abilities to recognize chess openings, we have utilized the data available on the chess openings website - chessopenings.com[14].

The results of this analysis are presented in the evaluate_chessgpt.ipynb notebook, available in the project's repository[7]. During the experiments, the model has been presented with the prompt *"Q: {notation} what is the name of this opening?A:".* This way the model was forced to provide the name of the opening based on an initial move or a series of moves. The answers of the model were oftentimes very verbose but generally, they were often correct for more popular openings and the chance of an error increased with less-known chess openings. Figure 13 summarizes the results by showing the histogram of the edit distances for the longest common sequence between the prediction and the ground truth (opening's name).

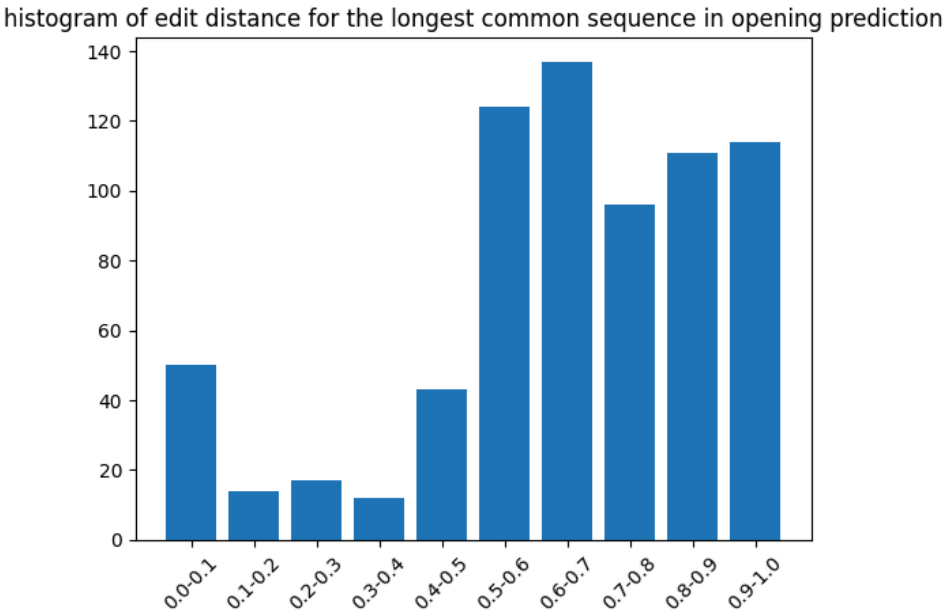

Figure 13. Histogram of edit distance between the longest common sequence in the chess opening name's prediction and the ground truth (real opening name) for the chessGPT model.

Furthermore, in this section, we have also included a small analysis of what happens if the start of a chess opening is given to the model, which is then asked to continue the series of moves. This experiment was also conducted on the chessGPT[4] model. It appears that the model tends to perform illegal moves or declare that the game is finished when asked to continuously produce the continuation of the sequence for the whole game. These results are shown in Figure 14.

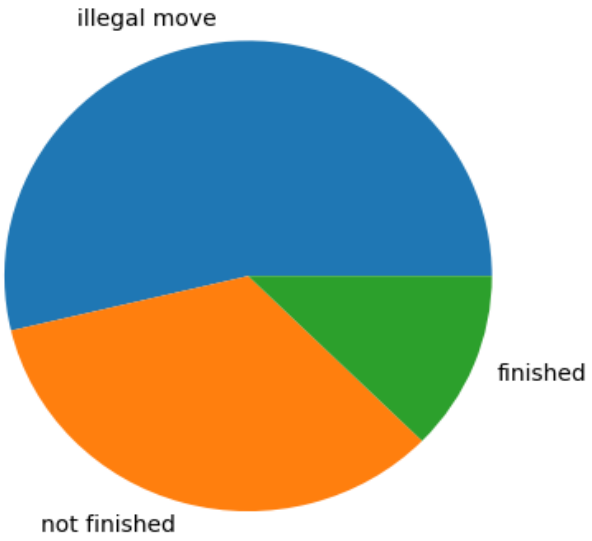

Figure 13. Pie chart comparing the times the chessGPT model has managed to finish the game on its own versus the number of times it has not finished the game vs the number of times it has performed an illegal move.

Results indicate that although the transformer-based models trained for the chess data like chessGPT[4] significantly outperform more general models like GPT-2[10] and GPT-2 - Large, they have their limitations, particularly when queried for long move sequences or when tasked with recognizing a rare opening. As such, the restrictions of these models could be addressed in further research.

Conclusions

In conclusion, in this paper, we have described several experiments and their results pertaining to the topic of using NLP models for the task of understanding the rules of chess, playing chess, and recognizing chess openings. Our results have shown that the quality of the moves made during a game of chess can be assessed using the Stockfish engine and that this also applies to the assessment of the quality of moves made by the models, alongside factors like the legality of the move. We have shown that models previously used for NLP-related tasks like the BERT[15] model can be made to “understand” certain rules of chess and classify moves as legal or illegal. Furthermore, we have shown that even general models not specifically trained on chess data like GPT-2[12] and GPT-2 - Large[13] seem to perform better than random players. Lastly, it appears that the chessGPT[4] model outperforms both GPT-2 and GPT-2 - Large models, however, it also tends to make errors when asked to generate very long sequences of moves. As such, improving the language-model-based solutions for the tasks of long move sequence generation could be a worthwhile topic for further research.

References:

GitHub repository: https://github.com/MatTheTab/NLP-Chess/

[1] Pereira, J., Fidalgo, R., Lotufo, R., & Nogueira, R. (2022). Visconde: Multi-document QA with GPT-3 and Neural Reranking. ArXiv. /abs/2212.09656
[2] Huang, D., Bu, Q., Zhang, J. M., Luck, M., & Cui, H. (2023). AgentCoder: Multi-Agent-based Code Generation with Iterative Testing and Optimisation. ArXiv. /abs/2312.13010
[3] Pietroszek, Krzysztof & Agraraharja, Zaki & Eckhardt, Christian. (2021). The Royal Game of Ur: Virtual Reality Prototype of the Board Game Played in Ancient Mesopotamia. 647-648. 10.1109/VRW52623.2021.00206.
[4] Feng, X., Luo, Y., Wang, Z., Tang, H., Yang, M., Shao, K., Mguni, D., Du, Y., & Wang, J. (2023). ChessGPT: Bridging Policy Learning and Language Modeling. ArXiv. /abs/2306.09200
[5] Lichess dataset: https://database.lichess.org
[6] Stockfish website: https://stockfishchess.org
[7] NLP-Chess Project Repository: https://github.com/MatTheTab/NLP-Chess
[8] DeLeo, M., & Guven, E. (2022). Learning Chess With Language Models and Transformers. ArXiv. https://doi.org/10.5121/csit.2022.121515

[9] Loshchilov, I., & Hutter, F. (2017). Decoupled Weight Decay Regularization. ArXiv. /abs/1711.05101
[10] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.
[11] Huggingface SAN dataset: https://huggingface.co/datasets/mlabonne/chessllm
[12] GPT-2 - Large on Huggingface: https://huggingface.co/openai-community/gpt2-large
[13] GPT-2: https://huggingface.co/openai-community/gpt2
[14] Chess openings website: chessopenings.com
[15] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv*. /abs/1810.04805