

Machine Learning to Study Patterns in Chess Games

Student Number: 690065435

Academic Year 2022/2023

Abstract

Abstract here.

I certify that all material in this report which is not my own work has been identified.

Signature: _____

Contents

1	Introduction	3
2	Preliminary Research	3
2.1	Early History of Computer Chess	3
2.2	Modern Developments in Computer Chess	3
2.3	Growth in the Popularity of Chess	4
2.4	Practical Uses of Chess Databases	4
3	Project Specification	4
4	Design, Methods, and Implementation	4
4.1	Downloading the Data	4
4.2	Data Processing	4
5	Data Exploration	6
5.1	Structure of Data	6
5.2	Distribution of Player Ratings	6
5.3	Effect of Rating Differential on Win Rate	7
5.4	Most Popular Openings by Category	8
5.5	Most Popular Base Openings	9
6	Results and Evaluation	10
7	Conclusion	10
7.1	Limitations and Future Work	10
	Appendices	12
A	Additional Outputs	12
B	Additional Plots	12
C	Examples	15

1 Introduction

Chess is one of the oldest and most popular board games globally – it has a rich history and vast literature dating back centuries, with the first comprehensive book about chess called ‘Book of the Chess’ appearing in the year 840, discussing chess openings, endings, and hundreds of chess problems [1, 2]. However, the large-scale exploration of chess games has only caught on recently – the most well-known and highest-rated chess engine [3], Stockfish, has used over 9,400 years of CPU time to analyse over 5.6 billion self-play chess games as of April 2023 [4]. Websites such as Lichess and Chess.com have provided an online platform for people to play chess – users can create a free account and search for a game with other players with various time controls, where their matchmaking system will pair them with a player of a similar skill level. These platforms have made the game accessible to more people in standardised formats – this has enabled the collection of data on a large scale, which makes it a fascinating domain for data mining and machine learning. Furthermore, chess games involve cognition and human behaviour that we can apply more generally – there have been studies in areas like social learning theory on other games such as Go [5], but they also apply to chess.

In this dissertation, we explore data mining and machine learning techniques to analyse large-scale data from the Lichess Open Database, which contains monthly records on tens of millions of chess games played by users of various skill levels. Our goal is to discover patterns and insights into how people play chess, which could contribute to further studies in chess such as cheat detection algorithms to improve the integrity of chess tournaments and promote positive growth in the game.

2 Preliminary Research

2.1 Early History of Computer Chess

Since the first digital computer switched on in 1945, programmers have been interested in using computers to play chess. In 1948, Alan Turing worked with David Champnowne to create the ‘Turochamp’, a machine routine for playing chess [6]. Claude Shannon wrote a paper in 1949 describing a computer that could play chess by analysing the seven most likely moves from the current positions and branching these into the seven most likely replies from the opposition continuously until the computer ran out of memory [7]. Dietrich Prinz eventually implemented the first chess program in November 1951, though it could not complete an entire chess game, and it used exhaustive search to play moves rather than a heuristic [6].

IBM made a significant breakthrough with their Deep Blue supercomputer starting with their initial prototype in 1995 [8]. It was released in 1996 [9] and was able to perform 200 million calculations per second [10]. Deep Blue was the first computer to use a heuristic search algorithm, searching for a solution to a problem by evaluating possible solutions and choosing the best one. In 1997, Deep Blue became the first computer to beat a world champion as it defeated Garry Kasparov over a six-game match [11], indicating that chess engines were better than human players. However, a lack of a deep understanding of the game was evident – Deep Blue accepted Kasparov’s sacrifice of a rook for a bishop in the first game, only to lose 16 moves later [10].

2.2 Modern Developments in Computer Chess

Stockfish is the strongest chess engine in the world. Both Lichess and Chess.com use Stockfish to provide game analysis on their platforms, and it is also used by professional players to analyse their games. It was released as a free, open-source chess engine in November 2008 [12], with volunteers donating CPU time to test improvements to the engine using a distributed framework, Fishtest [13]. Stockfish uses the alpha-beta pruning search algorithm which improves minimax search [14] by avoiding variations that will never be reached in optimal play due to either player redirecting the game [15].

However, other chess engines have used different algorithms in recent history. Most notably, DeepMind released AlphaZero in 2017 [16] based solely on reinforcement learning – it had no knowledge of chess beyond the basic rules and learned within hours by playing millions of games [10]. AlphaZero used a general-purpose Monte Carlo tree search algorithm, a heuristic search algorithm [16]. It uses

a tree to represent the game state, then randomly selects moves and uses the results of each game to weigh the tree nodes accordingly [17]. In December 2017, AlphaZero beat Stockfish in a 100-game match with 28 wins, 72 draws, and zero losses [18], showing the feasibility of alternative approaches, although Stockfish has since improved.

2.3 Growth in the Popularity of Chess

2.4 Practical Uses of Chess Databases

3 Project Specification

I will be investigating the following question: can we use data mining and machine learning techniques on historical data to discover patterns and insights in how people play chess? Our project will be broken down into the following objectives:

- Create a data pipeline for downloading and processing the data efficiently
- Explore the data to find interesting patterns and insights
- Implement machine learning to create models based on patterns in chess games
- Analyse and evaluate the usefulness of the models

In the next section, we will discuss how we will achieve these objectives by downloading the data, processing it, and analysing it.

4 Design, Methods, and Implementation

4.1 Downloading the Data

Collecting the data correctly was paramount to the success of our project, and it was important to use a large sample size to ensure that our insights represent the general population of chess games. We used the Lichess Open Database [19] of standard rated games for our data source – they upload tens of millions of games every month in PGN format, and they are easily accessible to the public. We decided to focus on games in 2022, as this enables me to capture the latest trends in chess.

The time controls in Lichess are decided based on the estimated game duration, using the formula: $total\ time\ in\ seconds = (initial\ clock\ time) + 40 \times (clock\ increment)$. Games between 29 seconds and 179 seconds are Bullet; games between 179 seconds and 479 seconds are Blitz, games between 479 seconds and 1499 seconds are Rapid, and games over 1500 seconds are Classical. Furthermore, games are either Rated or Unrated – the former results in rating points changing based on the game outcome, whereas the latter does not. The Lichess Open Database only includes Rated games. We will only be analysing Rated Bullet, Rated Blitz, and Rated Rapid games – we will not be including Rated Classical games, as they have a significantly smaller sample size and the longer time controls result in a vastly different playstyle.

We started by downloading the PGN files for each month, which introduced difficulties with big data. Each month’s file is around 30 GB to download and they are compressed, which means each month is around 210 GB when uncompressed, resulting in approximately 2.5 TB of data for the year. My laptop only had 1 TB of storage, and the sheer size of the data would make it unrealistic to process in a reasonable amount of time. Therefore, we collected a sample of 6 million chess games from each month which we would later filter down.

4.2 Data Processing

PGN (Portable Game Notation) files are a standard format for recording chess games – each game has headers containing information like the white player, black player, their ratings, and the result of the game. In addition, it stores a list of all the moves in each game. While PGN files are well-structured, they are not easy to process. The pgn2data Python library [20] provides a simple way to convert PGN

files into CSV files, but it exported both the moves and the metadata of each game in separate CSV files. We did not need the moves, so it was using up unnecessary space and taking a long time to process each game. It was also designed for generic PGN files so it did not include additional useful metadata provided by Lichess such as the opening of the game.

Thus, we wrote our own Python script to convert each PGN file into a single CSV file containing only the metadata of each game. We used the python-chess library to parse the PGN files and iterate over each game, skipping Rated Bullet games, Rated Classical games, games that involved a non-human player, and games that were abandoned. Our sample reduced from 60 million games to slightly over 40 million games after filtering it. To optimise performance, we split each month’s PGN into six files each containing 1 million games with pgn-extract (a PGN manipulator written in C) [21], and we used Python’s multiprocessing library to process each month in parallel – this reduced the processing time from 120 minutes to 20 minutes for each month.

Once we had a CSV for each month, we combined them into a single CSV file containing our final data set. We converted this CSV to a folder of Parquet files using Dask [22], which is a library for parallel computing in Python and extends common interfaces like pandas [23] to handle big data in a larger-than-memory environment. This significantly reduced the load time for the data, and it meant we could use the Dask library to perform parallel computations on the data.

For low-level, turn-based analysis of games, we created a .scout file using Scoutfish (a tool for querying chess databases written with C++) [24] to enable fast queries. For example, we can query the database to find all games where the board state is a certain position using the FEN (Forsyth-Edwards Notation) of the position, which is the standard string used in chess to represent the board state – this may be useful for finding occurrences of a specific variation of an opening. It provides the offset line number of each matching game in the PGN file, which we can use to find more information about the game. However, Scoutfish queries are not intuitive to write, so it can take a lot of effort to get insights using this tool. Therefore, we will primarily focus on high-level analysis of game metadata using Dask DataFrames.

Figure 1: Diagram of the Data Pipeline

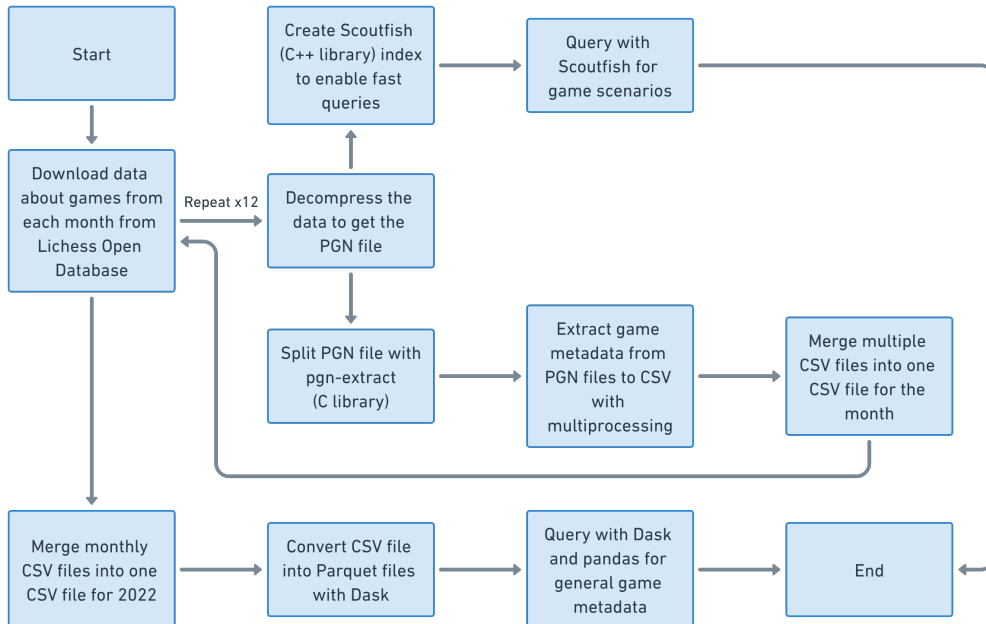


Figure 1 shows an overview of our data pipeline. It shows how we used the Lichess Open Database to collect the data over 12 months. To analyse game metadata, we aggregated the data into a single CSV file, and converted it to a folder of Parquet files to query with Dask. Meanwhile, to perform more detailed analysis of game scenarios, we used this data to create a Scout index to enable fast queries.

We also created a Python script to convert the CSV file to an SQLite3 database – this enables us to perform queries on the data over SQL for manual data exploration, which is easier and more

interactive than using Dask functions.

5 Data Exploration

5.1 Structure of Data

Our data consists of 40,121,728 Rated Blitz and Rated Rapid games that were played on Lichess in 2022. Each game is described by: the datetime of when the game started, what type of game (Rated Blitz or Rapid) it was, the specific time control, outcome of the game (1-0 for a White win, 1/2-1/2 for a draw, and 0-1 for a Black win), how the game ended, ECO category of the opening, the specific opening that was played, who played White and Black, their Lichess ratings, and a URL to the game on Lichess. Figure 2 shows the head of the DataFrame, which contains the first five games in the data sample.

Figure 2: Structure of Data

	UTCDateTime	Event	TimeControl	Result	Termination	ECO	Opening	White	Black	WhiteElo	BlackElo	Site
0	2022-01-01 00:00:11	Rated Rapid game	600+0	1-0	Normal	B00	Nimzowitsch Defense: Scandinavian Variation, A...	shahzad97	mpasha98	1500	1205	https://lichess.org/UMMGcaGz
1	2022-01-01 00:00:11	Rated Blitz game	300+3	0-1	Normal	D00	Queen's Pawn Game: Mason Variation	nicolas_clos	pangma	1430	1412	https://lichess.org/yJl78ET0
2	2022-01-01 00:00:12	Rated Blitz game	300+0	1-0	Normal	E97	King's Indian Defense: Orthodox Variation, Aro...	oyuka79	JUAN121	2422	2292	https://lichess.org/mrPOJU0W
3	2022-01-01 00:00:12	Rated Blitz game	300+0	1-0	Normal	C22	Center Game: Berger Variation	manoharkatwa	RigoBam	1241	1221	https://lichess.org/YJ1Ain03
4	2022-01-01 00:00:12	Rated Blitz game	300+0	0-1	Time forfeit	A40	Modern Defense	raffrantic1	Carly001	1044	1091	https://lichess.org/RATgKCCm

5.2 Distribution of Player Ratings

We counted the ratings of players in our data set. Rating systems are designed to predict the outcome of games so that players can be matched fairly. There are various rating systems used in chess – the most common is the Elo rating system, which is used by various sports. Lichess uses the Glicko 2 rating system, which accounts for volatility amongst other factors to provide better prediction accuracy than the Glicko 1 and Elo rating systems [25, 26]. The Glicko 2 rating system starts at 1500, and it is artificially limited to a minimum of 600 in Lichess. We created bins of 200 rating point ranges starting from 600 – this handles the representation of new players well, as those who have played few games will not have changed their rating significantly and likely get placed in the 1400-1600 bin. Note that the last four bins on the right of the bar chart do exist, but they are difficult to see – they contain totals of 11,622 for 2800-3000, 382 for 3000-3200, 459 for 3200-3400, and 1 player for 3400-3600. In addition, the counts are based on each game played, so some players may be overrepresented in the data set – a player who plays 100 games within the year will have 100 entries in the data set, whereas a player who plays 1 game will have 1 entry. However, this is not a significant issue because the data set is large enough to represent the distribution of player ratings well.

Figure 3: Distribution of Player Ratings on Lichess in 2022

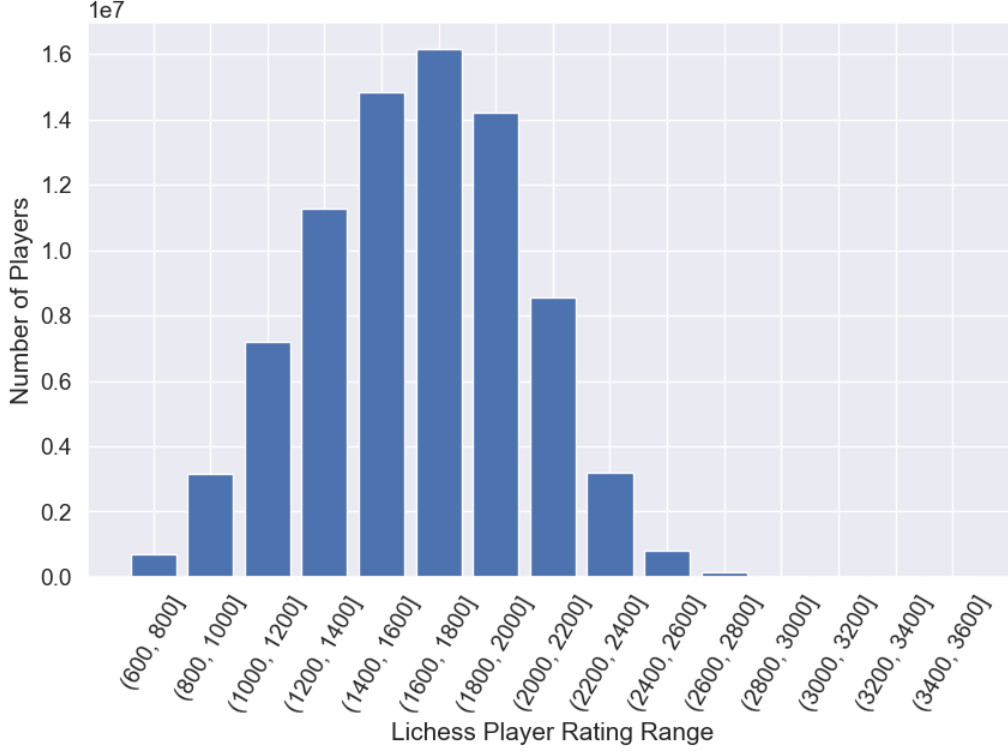


Figure 3 shows that the distribution of player ratings tends towards a normal distribution. We could explain this based on the concept that a player’s performance is similar to a random walk, which states that an object moves randomly with equal probability in different directions. In Lichess ratings, a player’s rating can be seen as the result of a series of random walks, where the player’s rating constantly changes as they win and lose games. Over time, their rating will tend to stabilise to represent their true skill level. Moreover, the central limit theorem states that the sum of a large number of independent and identically distributed random variables tends towards a normal distribution [27]. In this case, the ratings of individual players can be seen as independent and identically distributed random variables, with each game representing a single trial. Therefore, the central limit theorem predicts that the distribution of ratings will tend towards a normal distribution as the number of games increases. This is supported by how largest bin is the 1400-1600 bin, which contains the starting rating of 1500. Lichess’ matchmaking system could contribute to this – it matches players with similar skill, so games theoretically end up with a roughly equal number of wins and losses.

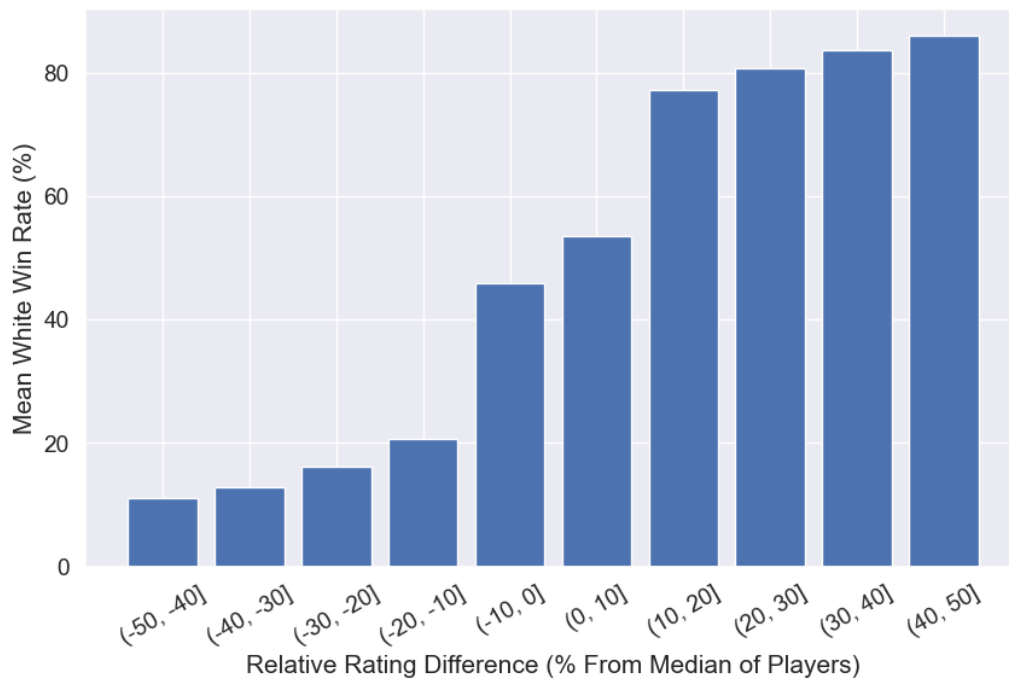
5.3 Effect of Rating Differential on Win Rate

Next, we investigated the impact of player rating differentials on White’s win rate. To begin with, we counted the outcomes of all games, and then counted the outcomes of games when White had a higher rating – White won 49.69% of all games, which rose to 53.98% of games when they had a higher rating than their opponent (as shown in figure 7 from the appendices). Then, we sought to quantify the impact of rating differential in both absolute and relative terms – the former was the difference in rating between the two players, and the latter was the difference in rating divided by the mean rating of the two players. Figure 8 and figure 9 from the appendices show the distribution of the absolute and relative rating differences. We grouped the data by rating differential and calculated the win rate for White. For example, when White is rated 1100 and Black is rated 900, the absolute rating differential is 200, whereas the relative rating differential is 10%. We focused on the win rate for White because the opening of the game is usually decided by White.

The results showed that there is a clear positive correlation between rating differential and White’s win rate. Both the absolute and relative differential metrics corroborate this, but we saw that relative rating difference highlighted the impact of rating on White’s win rate more clearly. Absolute rating

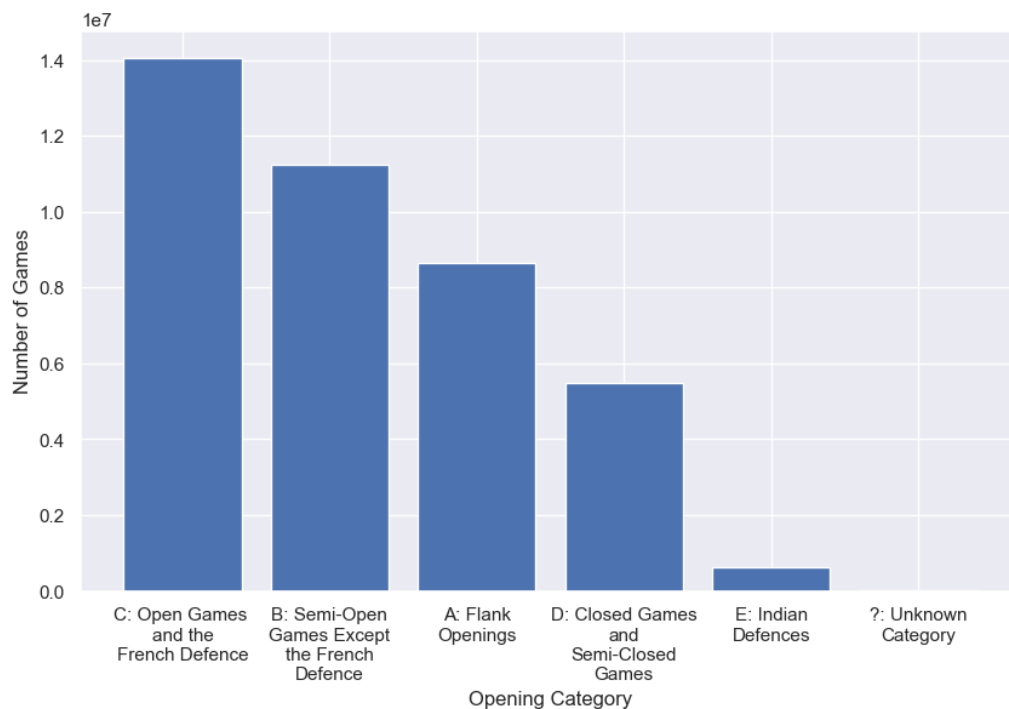
differential (shown in figure 10 from the appendices) is more intuitive to understand, but it is less accurate than relative rating differential (shown in figure 4) because it is not normalised.

Figure 4: Mean White Win Rate for Each RelativeEloDiff Bin



5.4 Most Popular Openings by Category

Figure 5: Most Popular Openings by Category on Lichess in 2022



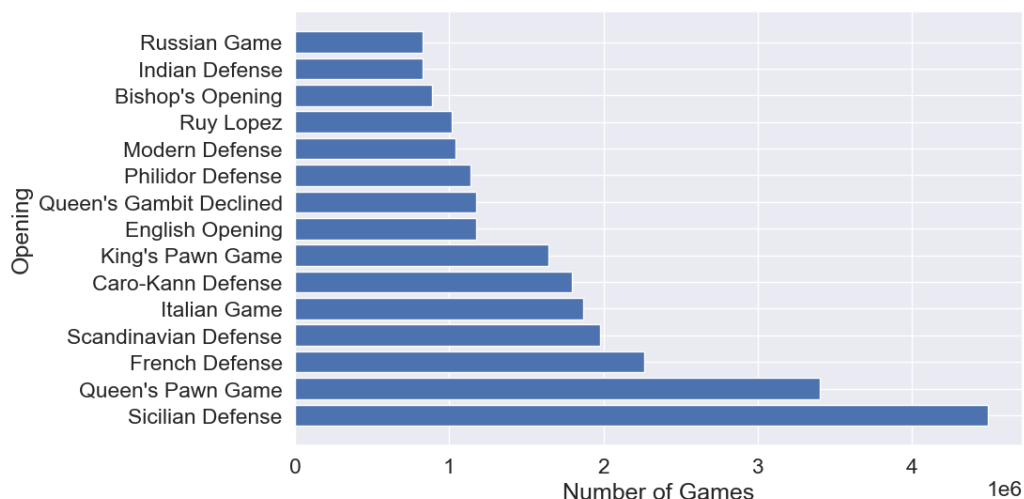
We also counted the most popular openings by category in figure 5, as this is represented by the ECO column provided in the Lichess PGN data. ECO (Encyclopedia of Chess Openings) is a classification of chess openings based on the first few moves of the game. Editors who mostly consist of chess

grandmasters select critical opening lines and assign them a code [28]. The ECO system consists of five categories, A to E, each of which represent a category of opening. Within each category, there are further subcategories to group openings more explicitly. For example, a game with an ECO code of A41 means that the opening is a flank opening as it is in the A category, and the 41 means that the opening was the Queen’s Pawn Game. The ECO system is not perfect as it does not include all openings (as demonstrated by the 32,799 games with the unknown category in figure 5), but it is useful to analyse opening trends.

Figure 5 shows that the most popular category of openings are open games and semi-open games, whereas the Indian defences and closed games are the least popular. This makes sense for the general population of games – the ideas behind these openings are simple and straightforward, as they focus on controlling the centre and developing pieces quickly. However, we see that the pattern is different for players rated 2000 and above – semi-open games and flank openings are more popular than open games in this category (shown in figure 11 from the appendices). At higher rated games, players often prefer to play more complex and dynamic openings that enable opportunities to create imbalances in the position and therefore gain an advantage at an earlier stage of the game.

5.5 Most Popular Base Openings

Figure 6: Most Popular Base Openings on Lichess in 2022



The data set also includes a column for the specific opening detected in each game based on Lichess’ opening detection. We grouped openings that were different variations of each other into base openings to get a more accurate representation – figure 14 from the appendices contains examples of base openings on the chess board. The top 15 most popular base openings account for 61.79% of all games in our data set. Figure 6 shows that the most popular openings are the Sicilian Defence and the Queen’s Pawn Game by a significant margin. The Sicilian Defence is the most well-studied response to White moving the king’s pawn forward by two, which could explain why it is the most popular opening. However, the Queen’s Pawn Game is slightly misleading, as it describes any opening beginning with White moving the queen’s pawn where they do not play the Queen’s Gambit and therefore encompasses a large number of openings such as the London System which are not labelled separately.

Delving deeper into the data, we found that the most popular base openings varied by rating groups. The top players rated 2000 and above strongly preferred the Sicilian Defense compared to other openings (shown in figure 12 from the appendices). This may be because the Sicilian Defense is a complex opening that requires a lot of preparation due to the number of variations, and it is unanimously accepted as the best response to White opening with the king’s pawn. Conversely, the Sicilian Defense becomes much less popular for players rated 1200 and below for the same reasons. Figure 13 from the appendices shows that the two most popular base openings for this group are the

Queen’s Pawn Game and the King’s Pawn Game, which represent generic openings starting with the queen’s pawn and the king’s pawn respectively – this is likely because they are simple and easy for beginners to play.

6 Results and Evaluation

7 Conclusion

7.1 Limitations and Future Work

References

- [1] B. Wall, “Earliest Chess Books and References.” <http://billwall.phpwebhosting.com/articles/oldtexts.htm>, 2013. Date Accessed: 3 April 2023.
- [2] P. R. Wonnig, *A short history of the game of chess: Chess history in brief*, vol. 6. Mossy Feet Books, 2014.
- [3] “Computer Chess Rating Lists: April 2023.” <https://ccrl.chessdom.com/ccrl/4040/>, 2023. Date Accessed: 3 April 2023.
- [4] Stockfish, “Stockfish Testing Framework.” <https://tests.stockfishchess.org/users>, 2023. Date Accessed: 3 April 2023.
- [5] B. A. Beheim, C. Thigpen, and R. Mcelreath, Strategic social learning and the population dynamics of human behavior: The game of Go *Evolution and Human Behavior*, vol. 35, no. 5, pp. 351–357, 2014.
- [6] B. J. Copeland and D. Proudfoot, Turing and the computer *Alan Turing’s Automatic Computing Engine: The Master Codebreaker’s Struggle to Build the Modern Computer*, Oxford University Press, Oxford, pp. 107–148, 2005.
- [7] C. E. Shannon, XXII. Programming a computer for playing chess *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 41, no. 314, pp. 256–275, 1950.
- [8] F.-h. Hsu, M. S. Campbell, and A. J. Hoane Jr, Deep Blue system overview in *Proceedings of the 9th international conference on Supercomputing*, pp. 240–244, 1995.
- [9] F.-h. Hsu, IBM’s deep blue chess grandmaster chips *IEEE micro*, vol. 19, no. 2, pp. 70–81, 1999.
- [10] S. Strogatz, One giant step for a chess-playing machine *New York Times*, pp. 1–6, 2018.
- [11] Y. Seirawan, H. A. Simon, and T. Munakata, The implications of Kasparov vs. Deep Blue *Communications of the ACM*, vol. 40, no. 8, pp. 21–25, 1997.
- [12] “About Stockfish.” <https://stockfishchess.org/about/>, 2022. Date Accessed: 22 November 2022.
- [13] “Fishtest Distributed Testing Framework.” <https://www.talkchess.com/forum3/viewtopic.php?start=0&t=47885>, 2022. Date Accessed: 22 November 2022.
- [14] J. v. Neumann, Zur theorie der gesellschaftsspiele *Mathematische annalen*, vol. 100, no. 1, pp. 295–320, 1928.
- [15] S. Maharaj, N. Polson, and A. Turk, Chess AI: competing paradigms for machine intelligence *Entropy*, vol. 24, no. 4, p. 550, 2022.
- [16] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, Mastering chess and shogi by self-play with a general reinforcement learning algorithm *arXiv preprint arXiv:1712.01815*, 2017.

- [17] G. Chaslot, S. Bakkes, I. Szita, and P. Spronck, Monte-Carlo tree search: A new framework for game ai in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 4, pp. 216–217, 2008.
- [18] M. Klein, Google’s AlphaZero destroys Stockfish in 100-game match *Chess. com*, vol. 6, 2017.
- [19] Lichess, “lichess.org open database.” https://database.lichess.org/#standard_games, 2022. Date Accessed: 1 March 2023.
- [20] Z. Qureshi, “pgn2data: A library that converts a chess pgn file into a tabulated CSV data set.” <https://github.com/zq99/pgn2data>, February 2021. Date Accessed: 1 March 2023.
- [21] D. J. Barnes, “pgn-extract: A Portable Game Notation PGN Manipulator for Chess Games.” <https://www.cs.kent.ac.uk/people/staff/djb/pgn-extract/>. Date Accessed: 1 March 2023.
- [22] Dask Development Team, “Dask: Library for dynamic task scheduling.” <https://dask.org>, 2016. Date Accessed: 1 March 2023.
- [23] pandas Development Team, “pandas-dev/pandas: Pandas.” <https://doi.org/10.5281/zenodo.7549438>, January 2023.
- [24] M. Costalba, “scoutfish: Chess Query Engine.” <https://github.com/mcostalba/scoutfish>, December 2016. Date Accessed: 1 March 2023.
- [25] Lichess, “Chess rating systems.” <https://lichess.org/page/rating-systems>. Date Accessed: 1 March 2023.
- [26] FIDE, “Deloitte/FIDE Chess Rating Challenge.” <https://www.kaggle.com/competitions/ChessRatings2/overview>, May 2011. Date Accessed: 1 March 2023.
- [27] L. Le Cam, The central limit theorem around 1935 *Statistical science*, pp. 78–91, 1986.
- [28] A. Matanović, M. Molorović, and A. Božić, Classification of chess openings 1971.

Appendices

A Additional Outputs

Figure 7: Effect of Higher Rating on White Win Rate on Lichess in 2022

```
Results of All Games:

1-0      19938052
0-1      18450502
1/2-1/2  1733174
Name: Result, dtype: int64

Out of 40121728 games:
  White won 19938052 games (49.69%).
  White lost 18450502 games (45.99%).

Results of Games When White Had a Higher Rating:

1-0      10662552
0-1      8242031
1/2-1/2  846503
Name: Result, dtype: int64

Out of 19751086 games when White had a higher rating:
  White won 10662552 games (53.98%).
  White lost 8242031 games (41.73%).
```

B Additional Plots

Figure 8: Distribution of Absolute Rating Difference on Lichess in 2022

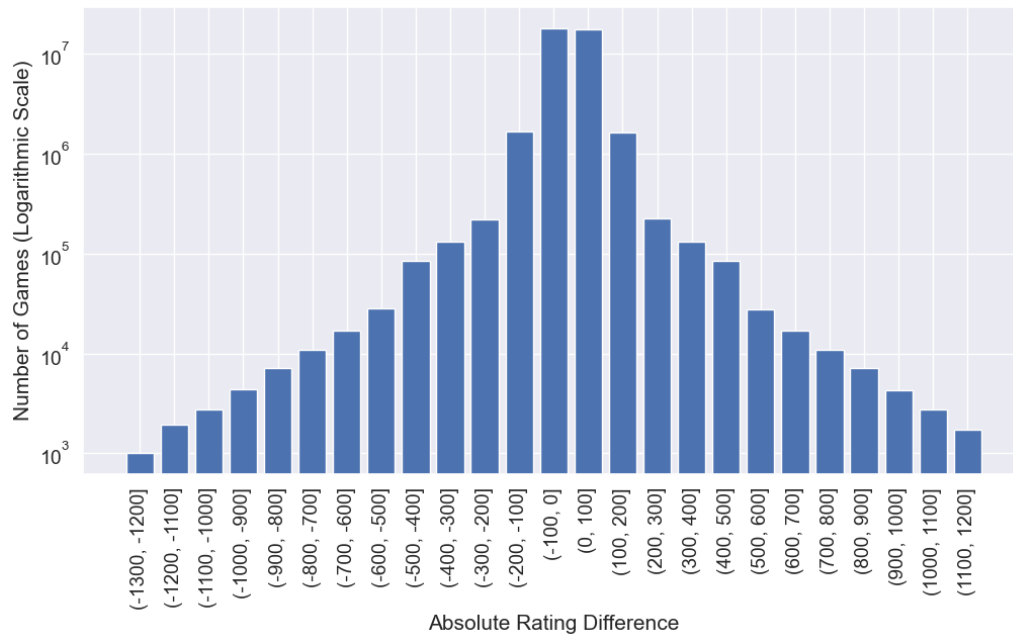


Figure 9: Distribution of Relative Rating Difference on Lichess in 2022

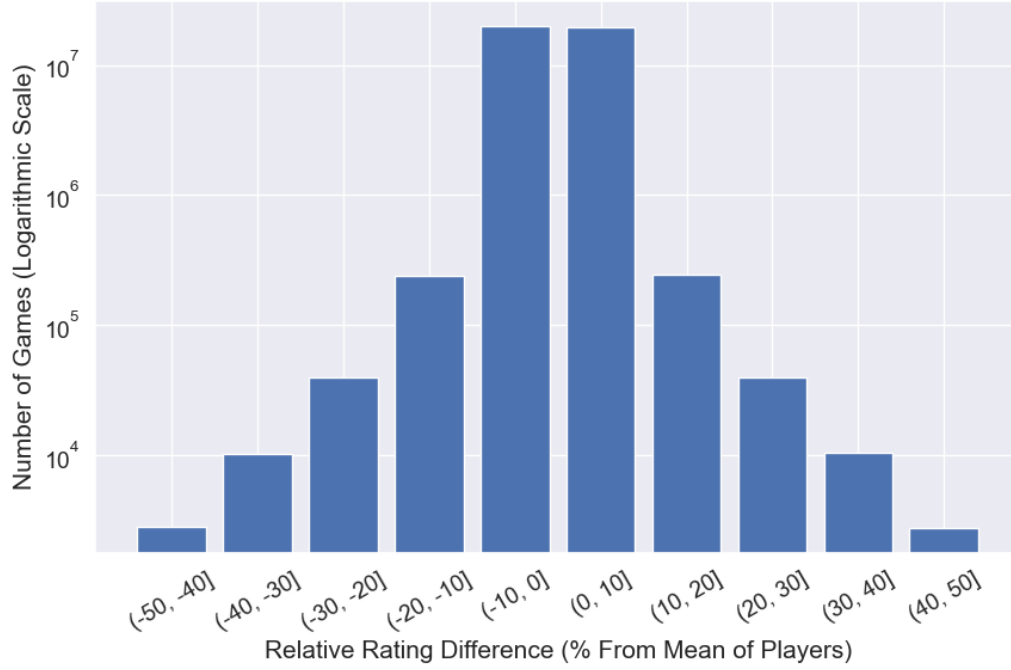


Figure 10: Mean White Win Rate for Each EloDiff Bin

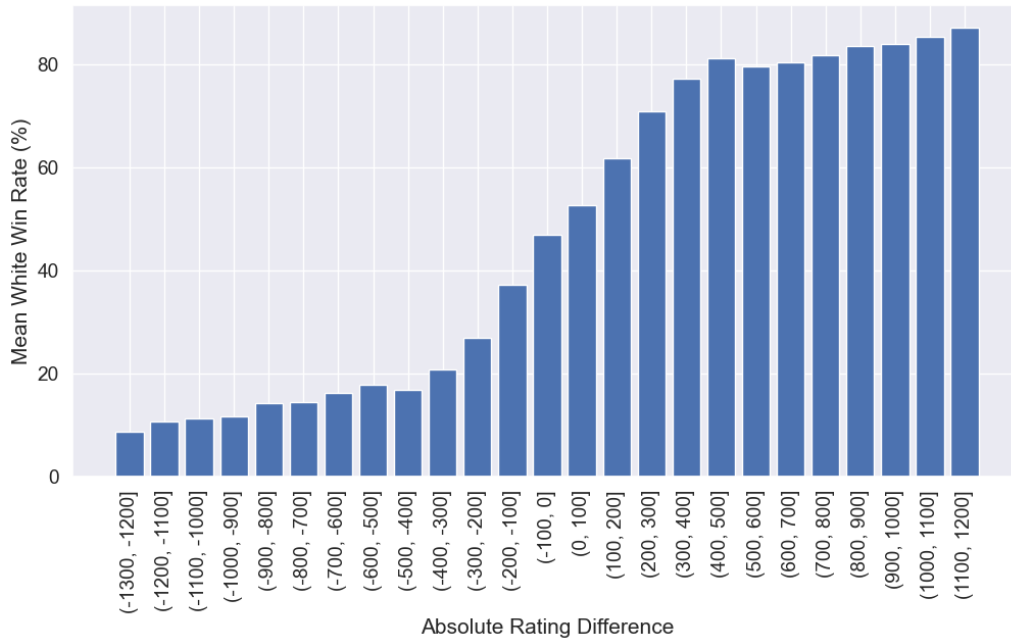


Figure 11: Most Popular Openings by Category on Lichess in 2022 (Rated 2000 and Above)

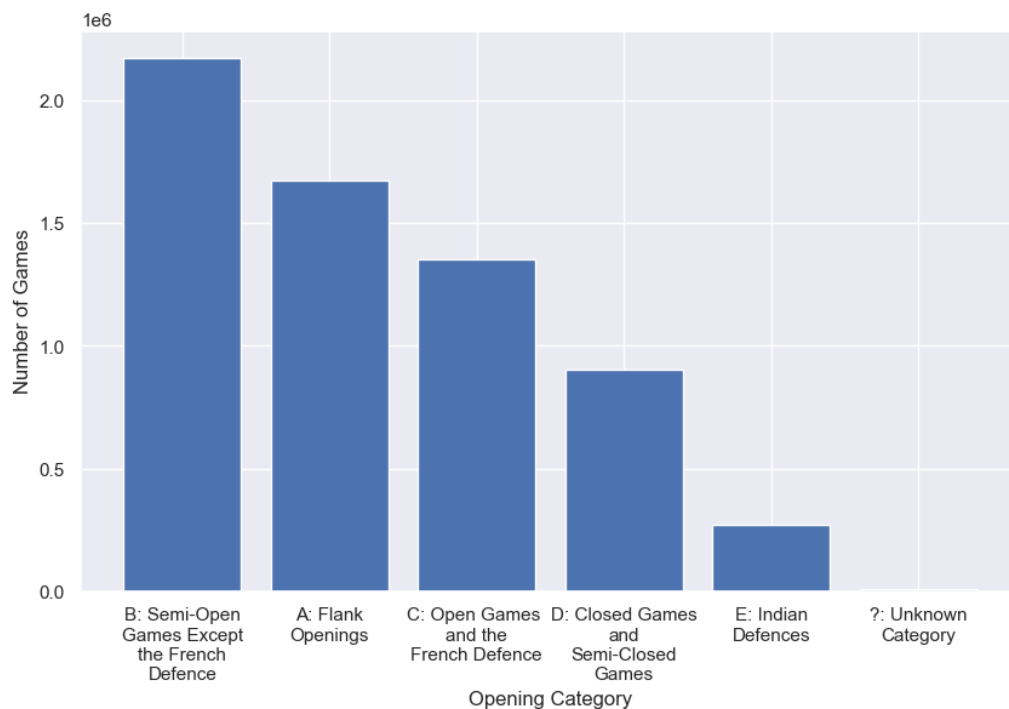


Figure 12: Most Popular Base Openings on Lichess in 2022 (Rated 2000 and Above)

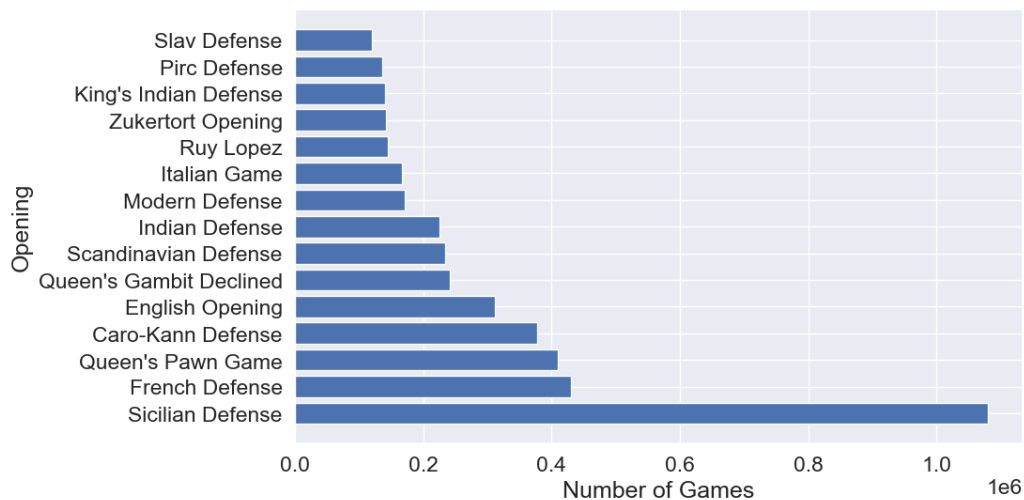
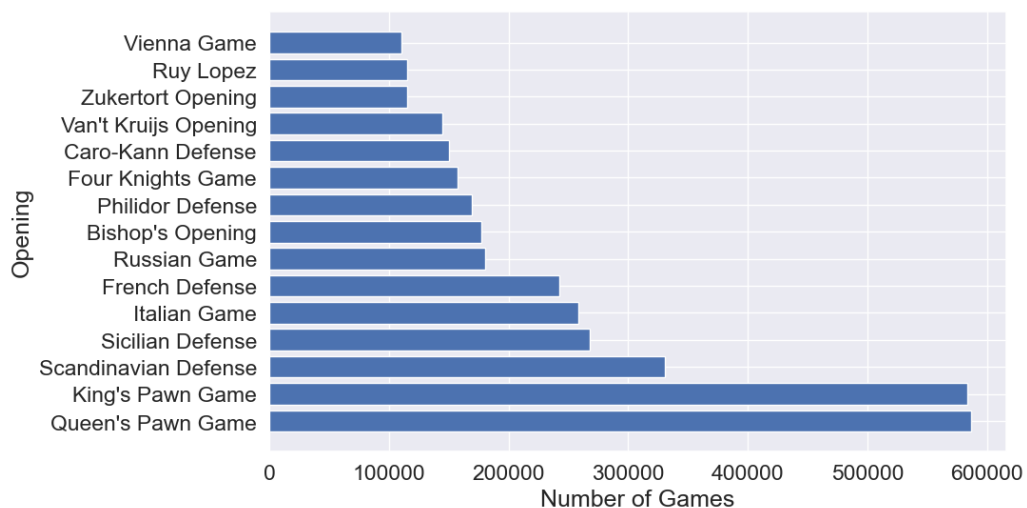


Figure 13: Most Popular Base Openings on Lichess in 2022 (Rated 1200 and Below)



C Examples

Figure 14: Examples of Chess Openings

