Predicting Chess Player Rating Based on a Single Game

Tijhuis, Tim ; Mavromoustakos Blom, Paris; Spronck, Pieter

Published in:
2023 IEEE Conference on Games (CoG)

DOI:
10.1109/CoG57401.2023.10333133

Publication date:
2023

Predicting Chess Player Rating Based on a Single Game

Tim Tijhuis*, Paris Mavromoustakos Blom† and Pieter Spronck‡
Department of Cognitive Sciences and Artificial Intelligence, Tilburg University
The Netherlands
Email: *t.tijhuis@uvt.nl, †p.blom@uvt.nl, ‡p.spronck@uvt.nl

***Abstract*—Traditionally, the relative strength of a chess player within a competitive pool is identified by a rating number. In order to reach a fair rating that best represents their level of play, chess players are required to play numerous games against various opponents within that pool. However, intuitively, experienced chess players are capable of extracting a rough estimate of a player's strength by looking at the moves they made in a single game. How accurately could a machine learning model based on a large dataset of chess games predict player ratings from a single game, and what would these predictions depend on? This paper presents an attempt to identify, encode and model chess gameplay features in order to predict a player's rating from a single game played. If successful, such a model could be employed to attach a fair initial rating to a new player within a pool before any games are played. We use an extensive dataset of chess games downloaded from a popular online chess platform, from which we extract a set of 30 features which are used to model and ultimately predict players' ratings. Our findings show that we are capable of predicting the rating bracket of a player with 79.3% accuracy when considering the extreme ends of the dataset (lowest vs. highest rated players), while the accuracy consistently drops as we increase the respective bracket width. We discovered that the most important features of our predictive models are both theory- and engine-related; most importantly, the features that we have extracted lead to explainable, quantifiable predictions of chess player strength.**

***Index Terms*—Chess, Player rating, Rating prediction, Predictive modelling**

I. Introduction

Chess is arguably the most popular board game in the history of humanity, with its origins being traced back to ancient Asian civilisations [15]. In modern days, chess is played casually and competitively both "over the board" as well as online, through numerous web and mobile applications. The game of chess has evolved into a multi-million dollar market; the prize money for the 2019 world chess championship reached 1,600,000$ [21], meanwhile the annual profit of the most popular online chess platform *chess.com* is over 50 million $ [1].

The relationship between computers, chess and Artificial Intelligence (AI) is synergetic. The implementation of chess-playing agents has been at the forefront of AI research and has set influential milestones, such as IBM's *DeepBlue* beating the world champion Garry Kasparov in 1997 [9]. Nowadays, chess engines have greatly surpassed the level of human play; they are mostly used to analyse and evaluate chess moves accurately and efficiently. Furthermore, the vast volume of publicly available chess game data has enabled chess research beyond the scope of chess-playing agents, such as computer vision [7], cognitive science [10] and human performance analysis [11].

In this paper, we use a large dataset of online chess games through which we attempt to model human players' level of play. Our predictive models take the moves from an unknown single chess game as input and produce a bin-based estimation of player skill rating as output. More specifically, we divide our dataset in 10 bins, each containing 10% of the dataset's games, sorted by player rating. Of the total dataset, a portion equally sampled from all bins is used for model training, while each individual game of the remaining data is used for the model's evaluation. To build our models, we extracted and analysed a total of 30 chess gameplay features which derive both from fundamental chess theory as well as state-of-the-art chess engine analysis. These features were employed in order to encode each game in the dataset.

This study presents a novel approach regarding the analysis of chess games from a machine learning perspective. If performed accurately, the prediction of a player's chess skill rating from a single game's data could be used to position a new player into an existing pool of chess players, even before that player competes within the pool. Moreover, beyond predicting a player's chess skill level, our models could be used to interpret which aspects of chess gameplay correlate to which skill rating bracket, providing a tool for players to detect mistakes and improve their overall gameplay. Lastly, reliable rating estimation could be used to detect players whose gameplay patterns do not correspond to their actual rating, contributing towards an automatic cheating detection system.

II. Related Work

A. Development of Chess AI

In the recent past, game-playing agents have set benchmarks for AI research in different categories of games, such as video games (e.g. Starcraft [32] and Dota2 [8]), card games (e.g. Hanabi [6]) and board games (e.g. Go, Shogi and Chess [30]). In chess (and similar board games) research particularly, a recent breakthrough was the introduction of *self play* [30], where a game-playing agent achieved superhuman performance in the game of chess through reinforcement

979-8-3503-2277-4/23/$31.00 ©2023 IEEE

learning; the agent, which was only fed the rules of the game, learned how to play by starting with random moves and iteratively optimised its decisions. Alternatively, "traditional" chess engines such as *Stockfish* use game-tree search and a heuristic evaluation function to determine whether a specific chess move is favorable to the player or not [22].

However, achieving mastery in the game of chess is not the only context within which the game is studied. McIlroy-Young et al. [23] argue that the level of mastery chess engines have achieved is not always explainable and understandable by human players, even of the highest level. Simply put: humans do not perceive (and play) chess the way engines do. Therefore, the researchers presented a series of studies on *Maia* [23]–[25], a chess-playing agent based on *AlphaZero* [30] which attempts to distinguish engine-like from human-like chess gameplay.

The aforementioned studies often require vast amounts of computational power for model training. Fundamentally, the game of chess is estimated to have 10^{46} possible positions, without including pawn promotion [31]. Therefore, at the moment of writing, chess remains an "unsolved" game. For that reason, the focus of this study shifts from chess-playing agents to the estimation of human player skill level. Our main goal is to derive a set of features, which are rooted both in established chess theory and post-game engine evaluation, in order to encode and interpret various levels of chess gameplay. Ultimately, we attempt to estimate the skill level of a chess player by only looking at a single game they played.

B. Player rating and rating prediction in games

Rating systems have been designed to extract a ranking within a pool of entities that are compared to each other in a pairwise fashion. Historically, the most popular rating system is the Elo system, invented by Aprad Elo and adopted by the World Chess Federation in 1970 [13]. Since, various improvements and adaptations of the Elo system have been implemented and used both in chess [16], [17], sports [19] and online video games [28]. Player rating is mostly used in matchmaking, i.e., determining opponent pairs or teams that are at an approximately similar level of play. It has been shown that playing against opponents of disproportional skill level can negatively affect engagement [34].

Rating systems such as Elo require a relatively high number of pairings (and results) to converge to an appropriate rating number. Furthermore, rating numbers are only relevant within the pool in which players or teams compete; they are not generalisable beyond the pool. Zhang et al. [34] recently presented a framework called *QuickSkill*, which attempts to rapidly extract player ratings in multiplayer online battle arena games. To that end, the authors propose a profiling mechanism which receives snapshots of the game state every three minutes and builds a detailed profile of each player based on their in-game performance. This system was built to address the cold-start problem of traditional rating algorithms (i.e., the uncertainty about a player's skill level in the early stages of competition). The method we propose in the present paper contributes in that direction as well; we analyse a single chess game from an unknown player and extract an estimation of which rating bracket that particular player belongs to.

While, to our knowledge, modelling player rating has been sparsely studied within the context of chess [12], [18], it has been a topic of growing interest in the video game domain. Notably, Aung et al. [4] present a longitudinal dataset from the game League of Legends, through which they study the relationship between early skill learning rate and end-of-season player performance. Their results show that with high accuracy, their system can predict which players will achieve master-level rating at the end of a competition, only by looking at their initial 10 games played. Pradhan and Abourazakou [26] introduced a multi-criteria decision-making tool in order to extract power rankings of teams in the game Dota2. They argue that power ranking systems are appropriate for competitive video games (Esports) given the vast amounts of generated data from both casual and competitive games. Their resulting rankings are strongly correlated to traditional rating systems such as Elo, Glicko-1 and Glicko-2.

C. Chess theory

Given the long history of the game of chess, a large volume of resources has accumulated over the years. These resources span from books written by chess masters [20], chess learning books for children [14], online articles with chess tips, tricks and principles [29], [33] to online video tutorials and analyses of past games. In the past, IBM's *DeepBlue* used chess opening and gameplay theory principles as features [9]. Similarly, the models presented in this paper are based on features that are extracted from fundamental chess theory. Additionally, we used post-game evaluations of chess positions using the *Stockfish* engine [27] to enrich the feature set. A more detailed explanation of the features used in this study is presented in Section III.

III. Experiment

A. Dataset

The dataset used in this study consists of two Portable Game Notation (PGN) files downloaded from *Lichess*[1]. PGN files contain the notation of the chess moves of each chess game, as well as meta-data such as the players' nicknames, player ratings and time constraints (total time available per player – also called time control). Each file contains all the games played on Lichess for one month. We used the available data for the months of January and February 2013. These were the earliest data available for download, and were preferred for feasibility reasons; memory and computational power constraints. The two files combined represent a total of 240000 games, which are then split into games played by black and white, resulting in a total of 480000 datapoints. An overview of the dataset is illustrated in Table II.

[1]Lichess (https://www.lichess.org) is a popular open-source online chess platform that provides public chess game data from games played on the platform.

TABLE I
RATING RANGE AND PLAYER DISTRIBUTION PER BIN.

Bin	1	2	3	4	5	6	7	8	9	10
Rating Range	800-1339	1340-1429	1430-1492	1493-1541	1542-1593	1594-1647	1648-1706	1706-1773	1774-1870	1871-2341
Number of players	1829	2132	2038	2858	1981	1982	1869	1607	1294	778

TABLE II
DATASET DEMOGRAPHICS AND META-DATA.

Total number of unique players	3632
Average number of games per player	125.9 (σ^2 = 218.9)
Minimum and maximum games per player	10 – 3139
Total number of bullet games ([1-2] mins total time)	141735
Total number of blitz games ([3-5] mins total time)	197360
Total number of rapid games ([10-15] mins total time)	117626
Total number of games in other time controls	743

From the PGN files, we kept players' nicknames, ratings, piece colour and the algebraic notation of the moves. Algebraic chess notation is a method of recording the moves played that allows the post-hoc reproducibility of a chess game. Other features such as match outcomes and time control were discarded from the dataset; the features used for model implementation were encoded in such a way that they can be considered time-invariant. More specifically, several features were extracted at four distinct timestamps: after 25, 50, 75 and 100% of the game's total moves. These features are indicated by (x4) in Table III.

Before extracting the features and feeding them into machine learning models, the data went through several pre-processing steps. Even though almost all the games from the Lichess PGN files are consistent in terms of formatting, two exceptions were detected and removed from the dataset. First, 1290 games (0.27% of the total dataset) without a player nickname or rating were deleted. Then, 7473 games (1.56% of remaining dataset) were deleted because the algebraic notation contained engine evaluation after every move. These games were discarded because engine evaluation will be applied at a later step for all games in the dataset. Finally, 13773 (2.92% of remaining dataset) games were removed, containing players which played 10 or less total games. As mentioned in Chapter II, chess rating is calculated over the result of all previous games of a player. Therefore, in this research we apply a heuristic minimum threshold of 10 games. This minimises the number of games where a player is playing with an inaccurate rating. The final dataset after pre-processing consists of a total of 457464 datapoints.

Finally, the dataset was split into 10 equally sized bins, each containing 10% of the total amount of players, sorted by rating. Therefore, bin 1 contained the bottom 10% of players and bin 10 contained the top 10% of players in terms of rating. An overview of the rating ranges and the number of players per bin is illustrated in Table I. Note that the same player can be part of more than one bin, since their rating before each game in the dataset is considered. Hence, game results can cause a player to move across bins. After pre-processing, all of the dataset's games were evaluated by the *StockFish* engine ($depth = 8$).

B. Feature Extraction

A total of 30 features (including features that were sampled four times per game) were extracted from the dataset. These features are based on chess theory and principles, as well as *StockFish* engine evaluations. An overview of the features extracted and a brief explanation is illustrated in Table III.

From the extracted features, several are only available post-game, such as the total game length and engine evaluation-related features. From engine evaluations, we labelled specific moves as "mistakes" or "blunders". Specifically, based on *StockFish*'s centipawn evaluation mechanism, moves that resulted in loss of value between 100 and 300 centipawns (approximately the value of one to three pawns) were labelled as mistakes, whereas moves that resulted in loss of value greater than 300 centipawns (more than the value of a minor piece) were labelled as blunders. Furthermore, we extracted forced checkmate-in-3 positions (positions in which the player could force a checkmate or be checkmated by their opponent in the next three moves).

Moreover, four features were extracted at 25, 50, 75 and 100% of a game's total moves. These features are isolated, doubled and tripled pawns, as well as total piece mobility. We decided to extract the above features at distinct timestamps in order to encode gameplay in the opening, mid-game and end-game phases of a chess game. All features that contain temporal information were standardised by the total number of moves in a game, allowing for comparison across games of different length. For example, the "first blunder" feature is represented as a percentage of the total moves made by the player, instead of the move's absolute index.

Finally, the remaining features represent various principles of chess theory, such as placing (or not placing) knights at the edges of the chess board, placing rooks on the 7^{th} or 2^{nd} rank, developing minor pieces (knights and bishops) before major pieces (rooks and queen), castling (protecting the king with a rook and pawns on the side of the board), as well as defending the centre of the board in the opening phase of the game.

C. Model Selection and Evaluation

In order to predict chess rating, the aforementioned features are fed into machine learning classifiers, namely Random Forest (RF) and Support Vector Machine (SVM), which are compared accuracy- and efficiency-wise. RFs have been shown to perform well in predicting skill learning in video games [4], while SVMs were used to accurately predict match results in

TABLE III

FEATURES EXTRACTED FROM THE DATASET. FEATURES THAT WERE EXTRACTED MULTIPLE TIMES THROUGHOUT THE COURSE OF A GAME (AT 25, 50, 75 AND 100 % OF THE GAME'S TOTAL MOVES) ARE HIGHLIGHTED BY (X4).

Feature	Description
Game length	The total number of moves made during a game
Moves before castling	The number of moves made by a player before the player castles. If no castling takes place, it is equal to the total number of moves
Isolated pawns (x4)	The number of isolated pawns (pawns that have no neighbouring pawn on either side). Measured at 25, 50, 75 and 100% of the game's total moves
Doubled pawns (x4)	Number of doubled pawns (two pawns of the same colour on the same file). Measured at 25, 50, 75 and 100% of the game's total moves
Tripled pawns (x4)	Number of tripled pawns (three pawns of the same colour on the same file). Measured at 25, 50, 75 and 100% of the game's total moves
Knights or bishops	A binary variable that equals 1 when bishops and knights are developed (moved from their starting square) before the queen and rooks
Defending centre	The number of minor pieces (knights and bishops) that directly attack or defend the four centre squares (d4, d5, e4, e5) after the first 5 moves
Pieces moved	The total number of unique pieces moved after 10 moves
Blunders	The total number of blunders made by a player. Blunders are detected through chess engine analysis (see Section III-B)
First blunder	The move where the first blunder is made by a player
Mistakes	The total number of mistakes made by a player. Mistakes are detected through chess engine analysis (see Section III-B)
First mistake	The move where the first mistake is made by a player
First mate opportunity	The move at which the player had the first opportunity to win the game within 3 moves
First opponent mate	The move at which the opponent had the first opportunity to win the game within 3 moves
Mobility (x4)	The difference in total number of legal moves between the player in perspective and the opponent. Measured at 25, 50, 75 and 100% of the game's total moves
First knight on edge	The move after which the player put a knight on the edge of the board (files 1 or 8) for the first time
Total knights on edge	The total amount of times a player put a knight on the edge of the board
Rook On 7th	The first time a player moves the rook to the 7th (or 2nd) rank

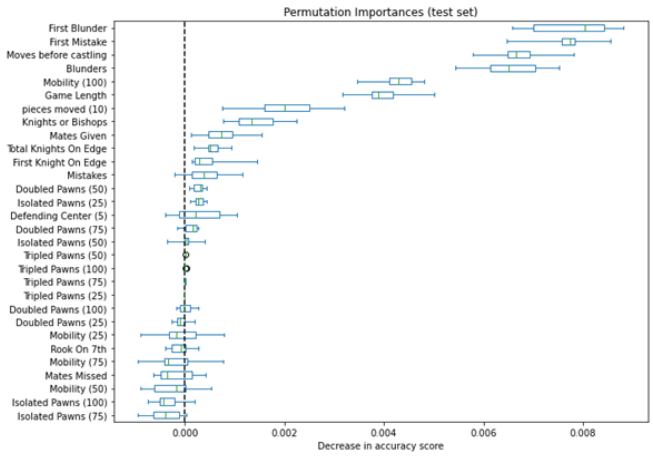

Fig. 1. Permutation feature Importance in a 10-class classification task, using the RF classifier.

Dota2 games [3]. While deep learning architectures could be considered at this stage, we prefer "traditional" machine learning methods that allow the extraction of feature importances and can be run efficiently on a consumer-grade laptop.

As target variable, we use the ID of the bin that the player in perspective falls under. Initially, this approach translates into a 10-class classification problem, when the entire dataset is considered. Since the dataset is balanced in terms of total games per bin, a random baseline yields an accuracy of 10%. In order to extract more fine-grained results, we also treat the problem from a binary classification perspective. To that end, we consider two bins at a time and symmetrically increment the width of the bins step-by-step. Initially we consider either the two extreme ends of the dataset (bins 1 and 10) or the two middle bins (bins 5 and 6) and iteratively add a neighbouring bin on each end. The classifier ultimately predicts whether the game in consideration belongs to a player from class 1 (lower-rated players) or class 2 (higher-rated players).

A 10-class classification is run using both RF and SVM. Depending on accuracy and efficiency, the best performing model is chosen to perform the binary classification. Consecutively, the best performing model's hyperparameters are tuned through a grid search method. From the 10-class classification task, a list of feature importances of each model is extracted. RF classifiers specifically, provide two measures for feature importance, namely Gini and permutation importance. Research has shown that permutation feature importance addresses the bias that is detected in Gini feature importance and will thus be preferred [2].

IV. RESULTS

A. 10-class classification

Initially, all 10 bins were considered, constituting a 10-class classification task. Using a subset of 80000 randomly selected games in total, equally distributed accross all bins and using all 30 features, we trained and evaluated both a RF and a SVM classifier. The 80000 game subset was used in place

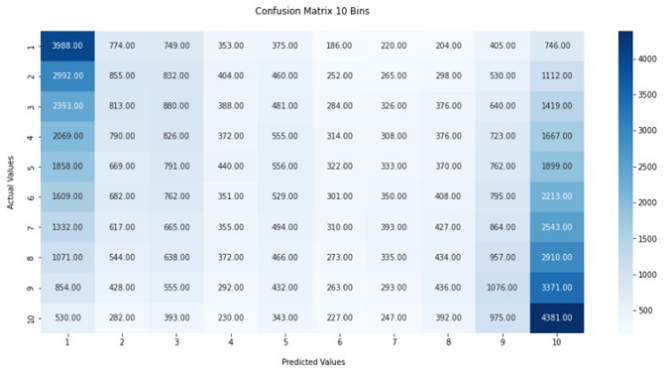

Fig. 2. RF predictions confusion matrix, based on a 10-class model (one class per bin)

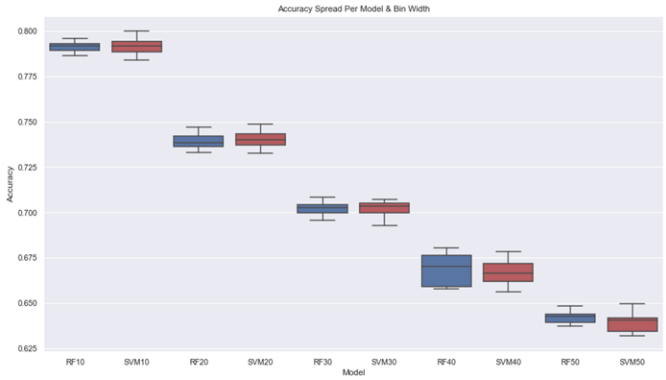

Fig. 3. Accuracy distribution per model and bin width. From left to right, the top- and bottom-most bins of the 80000 game subset are considered, while one neighbouring bin is added to each end at every step.

of the entire dataset for feasibility reasons. In particular, the SVM required 37 minutes per fold while the RF required 46 seconds per fold in a 10-fold cross validation task (using the 80000 game subset). The 10-fold cross validation ensured no overlap between games in the training and test set was present. Per fold, each game in the test set was fed as an unknown test datapoint into the model and the average accuracy over all test games was extracted. The bin ID was set as the target variable.

We compared our results against a random baseline (10% accuracy); both the RF and SVM model predicted the player in perspective's rating with an accuracy of approximately 17%. While this is an improvement relative to the baseline, Figure 2 shows that bins 1 (lowest rated players) and 10 (highest rated players) were predicted the most. This result is not surprising, expecting that players at the two extreme ends of the dataset approach the game of chess differently. This phenomenon has also been observed in other (video) games, for example STARCRAFT II [5]. Moreover, as illustrated in Table I, the width and difference between middle bins is marginal; the upper bound of bin 5 is only 1 rating point away from the lower bound of bin 6, meanwhile the two bins have a width of approximately 50 rating points each. This potentially explains the low prediction accuracy of the 10-class classifier.

Lastly, Figure 1 shows the permutation feature importance extracted from the RF model during the 10-class classification task. As illustrated, the most important features are both theory and engine evaluation-based. The most important feature (first blunder) has a permutation score of .008, which means that removing this feature from the set would decrease classification accuracy by approximately .8%. Lastly, features with negative permutation importance could be negatively affecting classification accuracy by a rather small margin.

B. Binary Classification

Next, we ran binary classification tasks by defining two classes (low vs. high rating) and iteratively increasing the class rating width. More specifically, we ran 10-fold classification tasks starting at bins 1 and 10, randomly sampling an equal amount of games from both bins as training and test set. Then at each consecutive task, we added a symmetrically neighbouring bin (2 and 9, 3 and 8, etc.) under the respective class label. Figure 3 illustrates the classification accuracy of each task, starting at the extreme ends of the 80000 game subset per class and ending up at 50% of the subset per class (bins 1-5 vs. 6-10). We compare each task to a random baseline which yields 50% accuracy, since the class data is balanced.

Looking at Figure 3, we observe that both models reach an accuracy of approximately 79% when the two extreme ends of the subset are considered. The classification accuracy consistently drops as more bins are added to the tasks on both ends of the subset. When the top and bottom 50% of the subset are considered, the accuracy of both models is approximately 64%. This iterative process shows that our models can discriminate between the highest- and lowest-rated players with relatively high accuracy, but at the same time show a significant accuracy loss when each class contains 50% of the subset.

From the above results, we conclude that accuracy-wise, our models perform at a similar level. However, there is significant difference in the runtime of the two models. Therefore, for the remainder of this section, we will use the RF model to extract further results. The hyperparameters of the RF classifier were tuned through grid search, resulting in the following final setup: *bootstrap = True, max_depth = 70, max_features = 'auto', min_samples_leaf = 4, min_samples_split = 3, n_estimators = 1200*.

Regarding feature importance, the same features as in the 10-class classification task were found to be most important, although in a different order. The most important feature is still the first blunder, followed by total number of blunders, first mistake, game length, moves before castling, and mobility (100). For the binary model the first blunder has an average impact of around 3.2%. Table IV illustrates the average value per bin for each of the six most important features.

C. Alternative Bin Configurations

Given the accuracy and speed of the RF classifier, we expanded our system on the entire dataset (457464 datapoints). Similarly to the tasks described in the previous section, our

TABLE IV
AVERAGE FEATURE VALUE PER BIN, IN A BINARY CLASSIFICATION TASK USING THE RF CLASSIFIER.

Feature / Bin	First Blunder	First Mistake	Moves Before Castling	Total Blunders	Mobility (100)	Game Length
1	10.81	6.12	6.32	3.66	-8.51	30.23
2	11.94	6.85	7.39	3.33	-3.65	32.20
3	12.76	7.28	7.94	3.16	-1.94	33.37
4	13.08	7.56	7.97	3.05	-0.82	33.63
5	13.74	7.88	8.22	2.96	0.01	34.60
6	14.04	8.26	8.23	2.84	1.15	35.02
7	14.50	8.55	8.28	2.71	1.74	35.63
8	14.93	8.93	8.35	2.60	2.97	36.22
9	15.50	9.47	8.46	2.51	3.61	37.01
10	15.78	10.43	8.32	2.21	7.12	36.98

TABLE V
PREDICTION ACCURACY OF THE RF CLASSIFIER PER BIN WIDTH. STARTING WITH GAMES FROM THE HIGHER- AND LOWER-MOST RATED PLAYERS (BINS 1 AND 10) AND ITERATIVELY INCREASING THE RATING RANGE WIDTH.

Bins considered	1 vs. 10	1-2 vs. 9-10	1-3 vs. 8-10	1-4 vs. 7-10	1-5 vs. 6-10
Accuracy	79.29%	74.44%	70.48%	67.21%	64.26%

TABLE VI
PREDICTION ACCURACY OF THE RF CLASSIFIER PER BIN WIDTH. STARTING WITH GAMES FROM THE MEDIAN-RATED PLAYERS (BINS 5 AND 6) AND ITERATIVELY INCREASING THE RATING RANGE WIDTH.

Bins considered	5 vs. 6	4-5 vs. 6-7	3-5 vs. 6-8	2-5 vs. 6-9	1-5 vs. 6-10
Accuracy	51.65%	54.68%	57.28%	60.36%	64.26%

dataset in two classes (high- vs. low-rated players) and starting at the extreme ends (bins 1 and 10) we performed 10-fold cross validation tasks, iteratively increasing class width. Table V shows the accuracy of the RF model under this setup.

Alternatively, we designed a different configuration where classification tasks started at the two middle-most bins (5 and 6) and iteratively more data was added towards the extreme ends of the dataset (5-4 vs. 6-7, 5-3 vs. 6-8, etc.). This configuration ensures that the two classes will always contain "neighbouring" datapoints, in terms of player rating. Table VI shows the accuracy of the RF classifier under this setup. We observe that the initial accuracy when only bins 5 and 6 are considered is only marginally above chance (51%), reaching a maximum of 64% when each of the two classes contains 50% of the entire dataset. We assume that the overall low accuracy observed in this configuration is caused by the similarity in playstyle of players that belong to the middle bins of the dataset.

D. Considering More Games

The results mentioned above were all extracted based on a single unknown game. Intuitively, we expect prediction accuracy to increase when more than one games (of the same player) are considered in the test set. To achieve this, we simply average the feature and rating values over N games. Three separate tasks were run using the RF model, to test this hypothesis for $N = 5, 10$ and 20 games. With an average of 5 games, the accuracy for the lowest and highest rated bin went up to 91.1%. The accuracy based on the same bins but averaged over 20 games even went up to 96.5%. Respectively, the accuracy of the 10-class RF classifier reached 25.1% based on the average of 20 games.

V. DISCUSSION

This paper presents a novel study towards the predictive modelling of chess player rating by analysing a single game played. The models presented were based on a total of 30 features, deriving from fundamental chess theory and state-of-the-art chess engine evaluation. The main results show that the highest- and lowest-rated players can be distinguished with fairly high accuracy; however, the closer the rating brackets considered for classification, the lower the accuracy observed.

An important contribution of this study is the set of features that was used for the encoding of chess games. Most importantly, Table IV illustrates that the most important features in a binary classification task derive both from engine evaluation and established chess theory. Taking a closer look, we can conclude that the set of features we have employed leads to understandable and quantifiable explanations of gameplay patterns and its effect on player rating. It is expected that higher rated players make less mistakes and blunders than lower rated players. Beyond the obvious, it is notable to see that higher rated players tend to castle later in the game and focus a lot more on the mobility of their pieces. These observations could be used to analyse one's own patterns of gameplay and ultimately become a better chess player.

Furthermore, even though our models do not perform at a satisfactory level across all rating brackets, such a system could be used to position new players in already existing pools, even before the new players start competing. In the present paper, our models are built on Lichess's database, but depending on the availability of data, this system could be adapted to work on any pool of players.

This study does not come without limitations; we acknowledge that a limited set of 30 features cannot produce generalisable, high-fidelity results. Allegedly, IBM's *Deep Blue* was implemented using a set of 8000 features [9]. Therefore, the existing feature set can only constitute a pilot study and requires further exploration. The addition of more detailed features could not only increase the overall accuracy of the system, but further boost the explainability of the predictions in terms of gameplay patterns. Moreover, having achieved a maximum accuracy of 79.3% (when considering extremal rating bins), our models can only be considered moderately reliable. For a system such as the one proposed in this study to be commercially used as a player rating estimator or even a cheating detector, the overall accuracy needs to be significantly increased.

Lastly, since this first study yielded promising results, the dataset should be enriched with more (and more recent) data. Despite the rules not having changed for decades, chess is

an ever-evolving game; since casual and competitive players have gained access to superhuman chess engines, novel rule-breaking gameplay patterns have emerged. Furthermore, the addition of more players to the existing dataset would not only lead to more robust results, but could also potentially increase the rating ranges of the proposed bins.

Future analyses could shift towards studying specific time controls separately, as we expect players who play shorter (in terms of total time available) chess games to base their playstyle more on intuition than deep analytical thinking. Moreover, depending on the feasibility constraints, studies like this could be transferred to the deep learning domain; neural networks have been proven to work exceptionally well in the game of chess.

VI. Conclusion

To conclude, we have shown that traditional machine learning algorithms can be employed to model and predict chess player rating based on a single game. Specifically, a Random Forest classifier was capable of predicting a player's chess rating bin with a maximum of 79.3% accuracy when only the lowest- and highest-rated players were considered. Increasing the rating bin width did cause a consistent decrease in classification accuracy.

For the purposes of this study, we have presented a set of 30 features through which we encode and analyse chess games. Several of these features contribute towards an explainable model of gameplay behaviour with respect to player rating. Systems such as the one proposed in this paper can identify patterns indicative of a specific skill level, and could be used to both improve the matchmaking accuracy of new players within an existing pool of players and facilitate the improvement of chess gameplay.

References

[1] Allerbest, E. How chess.com scaled a massive community. mixergy. com/interviews/chess-com-with-erik-allebest/, 2021. Mixergy.com.

[2] Altmann, A., Toloşi, L., Sander, O., and Lengauer, T. Permutation importance: a corrected feature importance measure. *Bioinformatics 26*, 10 (2010), 1340–1347.

[3] Anshori, M., Mar'i, F., Alauddin, M. W., and Abdurrahman Bachtiar, F. Prediction result of dota 2 games using improved svm classifier based on particle swarm optimization. In *2018 International Conference on Sustainable Information Engineering and Technology (SIET)* (2018), pp. 121–126.

[4] Aung, M., Bonometti, V., Drachen, A., Cowling, P., Kokkinakis, A. V., Yoder, C., and Wade, A. Predicting skill learning in a large, longitudinal moba dataset. In *2018 IEEE conference on computational intelligence and games (CIG)* (2018), IEEE, pp. 1–7.

[5] Avontuur, T., Spronck, P., and Van Zaanen, M. Player skill modeling in starcraft ii. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* (2013), vol. 9, pp. 2–8.

[6] Bard, N., Foerster, J. N., Chandar, S., Burch, N., Lanctot, M., Song, H. F., Parisotto, E., Dumoulin, V., Moitra, S., Hughes, E., et al. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence 280* (2020), 103216.

[7] Bennett, S., and Lasenby, J. Chess–quick and robust detection of chess-board features. *Computer Vision and Image Understanding 118* (2014), 197–210.

[8] Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680* (2019).

[9] Campbell, M., Hoane, A., and hsiung Hsu, F. Deep blue. *Artificial Intelligence 134*, 1 (2002), 57–83.

[10] Charness, N. The impact of chess research on cognitive science. *Psychological research 54*, 1 (1992), 4–9.

[11] Chowdhary, S., Iacopini, I., and Battiston, F. Quantifying human performance in chess. *arXiv preprint arXiv:2207.07780* (2022).

[12] Dangauthier, P., Herbrich, R., Minka, T., and Graepel, T. Trueskill through time: Revisiting the history of chess. *Advances in neural information processing systems 20* (2007).

[13] Elo, A. *The Rating of Chessplayers Past & Present*. 1986.

[14] Engqvist, T. *Chess Strategy for Kids*. 2016.

[15] Ferlito, G., and Sanvito, A. On the origins of chess, 1990.

[16] Glickman, M. E. The glicko system. *Boston University 16* (1995), 16–17.

[17] Glickman, M. E., and Jones, A. C. Rating the chess rating system. *CHANCE-BERLIN THEN NEW YORK- 12* (1999), 21–28.

[18] Haworth, G., Regan, K., and Di Fatta, G. Performance and prediction: Bayesian modelling of fallible choice in chess. In *Advances in Computer Games: 12th International Conference, ACG 2009, Pamplona Spain, May 11-13, 2009. Revised Papers 12* (2010), Springer, pp. 99–110.

[19] Hvattum, L. M., and Arntzen, H. Using elo ratings for match result prediction in association football. *International Journal of forecasting 26*, 3 (2010), 460–470.

[20] Kasparov, G. *Garry Kasparov on Modern Chess - Part 1*. Everyman Chess, 2022.

[21] Lewis, L. Chess world cup winner name, runner-up, and prize money. sportunfold.com/chess-world-cup-winner/, 2021.

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

[23] McIlroy-Young, R., Sen, S., Kleinberg, J., and Anderson, A. Aligning superhuman ai with human behavior: Chess as a model system. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2020), pp. 1677–1687.

[24] McIlroy-Young, R., Wang, R., Sen, S., Kleinberg, J., and Anderson, A. Learning personalized models of human behavior in chess. *arXiv preprint arXiv:2008.10086* (2020).

[25] McIlroy-Young, R., Wang, Y., Sen, S., Kleinberg, J., and Anderson, A. Detecting individual decision-making style: Exploring behavioral stylometry in chess. *Advances in Neural Information Processing Systems 34* (2021), 24482–24497.

[26] Pradhan, S., and Abdourazakou, Y. "power ranking" professional circuit esports teams using multi-criteria decision-making (mcdm). *Journal of Sports Analytics 6*, 1 (2020), 61–73.

[27] Romstad, T., Costalba, M., and Kiiski, J. Stockfish 13. *URl: https://stockfishchess. org*.

[28] Semenov, A., Romov, P., Korolev, S., Yashkov, D., and Neklyudov, K. Performance of machine learning algorithms in predicting game outcome from drafts in dota 2. In *International Conference on Analysis of Images, Social Networks and Texts* (2017), Springer, pp. 26–37.

[29] Sgîrcea, R., and Castellanos, R. 10 classic chess principles you need to know. https://thechessworld.com/articles/general-information/10-classic-chess-principles-you-need-to-know/, 2017.

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

[31] Steinerberger, S. On the number of positions in chess without promotion. *International Journal of Game Theory 44*, 3 (2015), 761–767.

[32] Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature 575*, 7782 (2019), 350–354.

[33] Wall, B. 101 essential chess tips. https://archive.org/details/101EssentialChessTipsByBillWall, 2016.

[34] ZHANG, C., WANG, K., CHEN, H., FAN, G., LI, Y., WU, L., AND ZHENG, B. Quickskill: Novice skill estimation in online multiplayer games. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (2022), pp. 3644–3653.