Predicting Level of Game Play (Chess) Using Transformer

Pratyaksh Mathur, Vandit Goel

April 2025

1 Abstract

This research aims to develop an approach for predicting chess player skill levels (Elo ratings) by analyzing their performance in individual games. We explore comprehensive feature extraction using engine-derived evaluations and methodologies covering move accuracy, strategic understanding, tactical awareness, time management, and opening knowledge. Our approach aims to provide more accurate initial ratings for new players, offering a nuanced assessment of performance beyond simple win/loss outcomes, and potentially aiding in the detection of anomalous play. This paper lays the groundwork for such predictive modeling by providing a comprehensive review of existing literature on chess skill evaluation, outlining key considerations for dataset construction, and proposing a structured methodological framework for future research. The aim is to establish a robust foundation for subsequent modeling efforts focused on predicting Elo rating from single chess games.

2 Introduction

The Elo rating system, developed by Arpad Elo, serves as a widely adopted standard for ranking chess players based on the outcomes of their games [1]. Officially adopted by the U.S. Chess Federation in 1960 Later, by FIDE (Fédération Internationale des Échecs) in 1970, this system became the standard. This system calculates ratings based on game outcomes between players, adjusting ratings according to expected probabilities. While effective in providing a general hierarchy of player strength that adjusts based on game outcomes, research indicates that Elo ratings may not be perfectly accurate in predicting the result of any single game [12]

This outcome-based approach presents limitations when attempting to evaluate performance within a single game or when seeking to provide an initial rating for players who have not yet played enough rated games. Furthermore, while Elo ratings effectively categorize players by skill, they do not offer an absolute

measure of playing strength, nor do they readily facilitate direct comparisons of players across different historical periods [14] This research aims to address the following questions: **Can a multi-feature approach accurately predict a chess player's Elo rating from a single game using sequential model?**

Our approach focuses on extracting simpler, interpretable features such as material type, centipawn loss, and blunders using the Stockfish engine. This method aims to provide a lightweight yet effective alternative for rating prediction. Predicting a player's Elo rating directly from the characteristics of a single game offers several potential benefits. Firstly, it could provide a more immediate and nuanced assessment of a player's performance in a specific game, going beyond the binary outcome of a win or loss [5] Secondly, such a predictive capability could be instrumental in assigning fair initial ratings to new players [13] Additionally, by analysing the statistical probability of certain game features occurring at different Elo levels, this approach could potentially contribute to the detection of anomalous playing patterns indicative of unfair play[6]

3 Literature Reveiw

The Elo rating system's statistical foundation and its ability to reflect relative player strength have made it a dominant force in competitive chess. However, studies have shown that its predictive power for individual game outcomes is not absolute, with empirical data sometimes deviating from theoretical expectations [5] This has spurred the development of alternative and improved rating systems, such as Glicko and Glicko- 2, which are designed to converge more rapidly to a player's true skill level and account for rating uncertainty [11]

Beyond traditional rating systems, researchers have explored methods for estimating chess player strength based on the quality of moves played, often leveraging the evaluations provided by powerful chess engines like Stockfish. The concept of Average Centipawn Loss (ACPL), which quantifies the average deviation of a player's moves from the engine's top recommendations, has emerged as a potential indicator of playing strength [2]. Studies have shown a correlation between lower ACPL and higher Elo ratings, suggesting that players with better ratings tend to make moves that result in smaller decreases in positional evaluation [6] Another prominent approach involves leveraging the analytical power of strong chess engines to evaluate the quality of human moves [5] Chess engines provide an objective benchmark against which human moves can be compared, offering a more direct assessment of skill than relying solely on game outcomes. The concept of "gain per move," which measures the change in the engine's evaluation of a position after a player makes a move, has been proposed as a key indicator of playing strength [4]. Analysing the distribution of this "gain per move" over the course of a game or multiple games can provide insights into a player's consistency and ability to make positionally sound moves. Research suggests that even relatively weaker chess engines can produce sensible rankings

of players based on the average deviation between their moves and the engine's preferred choices [7]. Furthermore, formulas incorporating a player's sensitivity to move quality differences and consistency in choosing better moves have been developed and tested against Elo ratings, revealing a smooth relationship between these parameters and skill level [10]. However, relying solely on engine evaluation at specific time points during a game has been shown to be insufficient for accurately determining Elo, as it lacks crucial context such as the move number and can be affected by how checkmate values are handled [9].

The advent of powerful machine learning techniques has opened new avenues for predicting chess ratings from game data. Models trained on large datasets of chess games can learn complex patterns and relationships between various game features and player ratings. The RatingNet model, for instance, utilizes Convolutional Neural Networks (CNNs) to extract positional features from game moves and Bidirectional Long Short-Term Memory (LSTM) networks to incorporate temporal information and clock times, achieving a notable level of accuracy in predicting Elo after each move [8]. The MAIA Chess project focuses on a related but distinct goal: training neural networks to predict the moves a human player would make at specific Elo rating levels, highlighting the connection between playing style and skill [15]. Studies have also explored the prediction of rating brackets from single games using a diverse set of features, including both theoretical opening knowledge and engine-based evaluations, achieving promising results [13]. However, research employing sequence models like Recurrent Neural Networks (RNNs), LSTMs, Gated Recurrent Units (GRUs), and Transformers to directly predict Elo from move quality has encountered challenges such as high prediction error and overfitting [3]. This suggests that while these models can capture the sequential nature of chess, accurately mapping move sequences to precise Elo ratings remains a complex task, potentially due to inherent noise in the data and the variability of human play. Attempts to predict the average Elo rating of players in a game by incorporating more variables like time, evaluation, and evaluation changes have also shown limitations, tending to over-predict lower ratings and under-predict higher ones [9].

The existing literature reveals both the potential and the challenges in predicting Elo ratings from single chess games. While simple game statistics offer limited predictive power on their own, methods leveraging chess engine evaluations to assess move quality show greater promise. The application of machine learning, particularly deep learning techniques, has demonstrated the ability to learn complex relationships between game features and Elo ratings. However, achieving high accuracy in single-game prediction remains an open research question, and various limitations, such as the context-dependence of engine evaluations and the difficulties in training robust sequence models, need to be addressed. Future research directions include incorporating more nuanced features like position complexity and time spent on decisions, utilizing stronger chess engines for analysis, and developing more sophisticated machine learning models to capture the intricacies of chess skill [10].

4 Methodology

This research leverages the Lichess open database, a vast collection of chess games from the Lichess.org platform, which hosts millions of games daily across various time controls and formats. We use the Lichess 2025 February dataset in PGN format, including moves and metadata like ratings and time controls, covering players rated 800 to 2800+ Elo. Our methodology aims to predict gameplay skill levels across the entire spectrum of player abilities.

4.1 Pre-processing

The raw PGN data is inherently unstructured, rendering it unsuitable for direct application in machine learning models. Consequently, an initial pre-processing phase is required to extract meaningful features and transform the move sequences into encoded vectors suitable for model input. The first step involves parsing the compressed PGN file and integrating engine annotations to facilitate feature extraction. The PGN dataset, provided as a .zst file, is processed in chunks to enable efficient handling. Each chunk is ingested into a SQLite database, which facilitates efficient querying and retrieval of game data for subsequent feature extraction.

For each game, engine evaluations are generated to annotate moves and extract features such as time spent, result (win/draw/loss), blunder/ inaccuracy/mistake counts, and additional metadata. Features are categorized into two groups: game-level features and move-level features (see Table 1). Following feature extraction, the data is normalized and encoded, with features unsuitable for direct model input being omitted. The normalized and encoded features are subsequently converted to TensorRecord files and saved in batches for model training.

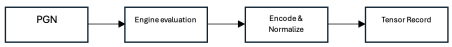

Figure 1: Pre-processing data pipeline

4.2 Feature Creation

Key features extracted include temporal context, move number, spatial data (FEN), and additional metadata, all of which undergo normalization and encoding prior to storage in tensor records. A challenge arises in representing time-related features, such as time remaining on the clock or time taken per move. To address this, a composite feature is constructed that expresses the time taken for a move as a fraction of the total available time, as well as the remaining time as a proportion of the total game time. This approach yields a normalized representation that is robust to variations in time controls.

Centipawn loss, a critical indicator of move quality, is normalized using a strategy that accounts for the magnitude of evaluation shifts, with special handling for cases where mate is imminent. If a mate is not possible, centipawn values are normalized accordingly; otherwise, a fixed value is assigned. Handling missing or incomplete features is addressed by omitting or imputing such values as appropriate. The final feature set comprises five game-level features and multiple move-level features, including a spatial encoding of the board state. The spatial encoding is represented as a 64-vector space, which is subsequently projected to a 10-feature vector.

4.3 Model Architecture

In order to potentially address longer-range dependencies in our data, we turn to transformer models. provide a powerful way of tackling sequence problems such as language modeling and machine translation, which have been difficult for traditional RNNs. It comprises an encoder stack and decoder stack, both of which implement a multi-headed self-attention mechanism. Through the attention mechanism, we hope that the model can learn to "focus" on the most salient parts of the game through learning an adequate embedding (with the use of positional encoding to tell us "Where" in the game these moves are played). Note that the decoder is required so that the model can piece together an output for a generation task (e.g., translating a sentence into English). Since our output value is a scalar value, not a sequence, we replace the decoder with two dense feedforward layers. We use a transformer architecture with 6 attention heads, a 512-dimensional feedforward layer, and 6 sub-encoder layers. We give our move-level features to the encoder and game-level features to a dense layer. The pooled output of the encoder is concatenated with the dense layer output and given again to a dense layer to predict 2-player performance in terms of Elo.

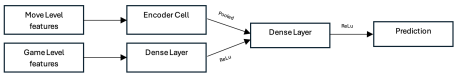

Figure 2: Model Architecture

5 Metrics and Results

The evaluation framework for our predictive model incorporated three primary metrics: Mean Absolute Error (MAE), Mean Squared Error (MSE), and a custom metric designated as elo_under_N, which quantifies the percentage of predictions within N Elo points of the actual rating. Given that the target variable (Elo ratings) underwent normalization during pre-processing, the interpretation of raw model outputs necessitated de-normalization. Consequently, all evalu-

ation metrics were multiplied by the normalization factor to facilitate human-interpretable assessment of model deviation from true ratings.

5.1 Performance Metrics

The model was trained over 20 epochs (Figure 6), achieving progressive reduction in loss function values. Post-training evaluation revealed an average MAE of ± 212 Elo points. The significance of this error margin warrants contextual interpretation within chess rating systems. While a 100-point differential at higher rating tiers (e.g., between 2700 and 2600) represents relatively minor skill discrepancy, the same differential at lower ratings (e.g., between 400 and 600) corresponds to substantially divergent skill levels with significant implications for win probability.

Our custom metric elo_under_100 yielded a value of 25%, indicating that one-quarter of all predictions fell within 100 Elo points of their true value. This demonstrates reasonable precision for a substantial portion of the dataset, though considerable room for improvement remains.

5.2 Distribution Analysis

Subsequent analysis of the dataset revealed several significant patterns with implications for model refinement. The player rating distribution centered around 1500 Elo (Figure 3), suggesting that implementation of alternative normalization techniques for Elo ratings could potentially enhance predictive accuracy. This concentration of samples at mid-range ratings may have influenced the model's training dynamics, potentially reducing sensitivity at rating extremes.

Further examination revealed a positive correlation between player rating and drawing probability (Figure 4). This observation aligns with theoretical expectations, as higher-rated players typically commit fewer tactical errors, resulting in more balanced contests with less decisive outcomes. This phenomenon can be attributed to the inverse relationship between player rating and centipawn loss (Figure 5), whereby stronger players demonstrate smaller evaluation differentials between their moves and engine-recommended alternatives.

These distributional insights suggest potential avenues for model improvement, particularly through stratified sampling approaches or specialized normalization techniques accounting for the non-linear relationship between rating differences and expected outcomes.

6 Conclusion

The findings of this research demonstrate that predicting chess performance based on gameplay data can be effectively formulated as a sequence modeling

problem. By leveraging transformer-based architectures, which are well-suited for capturing long-range dependencies in sequential data, we were able to process and interpret large-scale chess datasets with reasonable accuracy.

Despite employing state-of-the-art modeling techniques and integrating engine-based move evaluations, the model's predictive performance remained within a ± 200 Elo point range. This level of deviation highlights the inherent difficulty of the task, attributable to several factors. First, the prediction of a player's "true Elo" from a single game is intrinsically challenging due to substantial sources of noise. These include the potential for players to use assistance, natural fluctuations in player performance, and the volatility of unofficial online chess ratings.

In summary, while sequence models such as transformers show promise in modeling chess performance from large-scale game data, the results underscore the need for more robust normalization techniques, deeper contextual features, and potentially the integration of additional metadata to further improve predictive accuracy. The persistent challenge of noise and variability in unofficial online ratings remains a significant barrier to achieving fine-grained Elo prediction from single-game data.

7 Future Work

Several potential avenues for future research could build upon the findings presented here. Enhancements to feature engineering, such as incorporating additional contextual cues from chess games (e.g., material balance, multi-engine evaluations), could yield more nuanced representations of game states. A more rigorous framework for model selection and validation, potentially through cross-validation or ensemble methods, may improve the generalizability and robustness of the model. Finally, exploring non-sequential modeling techniques, such as Convolutional Neural Networks (CNNs), could offer alternative approaches to feature extraction and pattern recognition in chess gameplay data.

References

[1] Elo rating system. `https://www.chess.com/terms/elo-rating-chess`, 2025. Accessed: 2025-04-25.

[2] Approximation. How to estimate your elo for a game, using acpl, and what it realistically means? `https://lichess.org/forum/general-chess-discussion/how-to-estimate-your-elo-for-a-game-using-acpl-and-what-it-realistically-means`, 2022. Accessed: 2025-04-25.

[3] S. Cao and A. Lee. Performance prediction in chess using sequence. Stanford University, 2025. Accessed: 2025-04-25.

[4] D. Ferreira. Determining the strength of chess players based on actual play. *ICGA Journal*, 35(1):3–19, 2012.

[5] Dennis Golomazov. Chess stack exchange. https://chess.stackexchange.com/questions/4132/how-to-automatically-evaluate-a-players-performance-in-a-game, 2014. Accessed: 2025-04-25.

[6] E. L. S. Gonzalez. Medium. https://medium.com/@enzo.leon/data-science-and-chess-centipawn-loss-elo-correlation-e06089efd8b8, 2023. Accessed: 2025-04-25.

[7] M. Guid and I. Bratko. Using chess engines to estimate human skill. https://en.chessbase.com/post/using-che-engines-to-estimate-human-skill#icga, 2011. Accessed: 2025-04-25.

[8] M. Omori and P. Tadepalli. Chess rating estimation from moves and clock times using a cnn-lstm. *arXiv*, 2024. Accessed: 2025-04-25.

[9] oortcloud_o. Is eval by time enough to determine elo? https://lichess.org/@/oortcloud_o/blog/is-eval-by-time-enough-to-determine-elo/7jFsN3a2, 2023. Accessed: 2025-04-25.

[10] K. Regan and G. Haworth. Intrinsic chess ratings. In *AAAI*, volume 25, pages 7–11, 2011.

[11] A. Rizzoli. Predicting chess games results using lightgbm. https://medium.com/@alunariz/predicting-chess-games-results-using-lightgbm-818f30b5a7c3, 2023. Accessed: 2025-04-25.

[12] J. Sonas. Chessbase. https://en.chessbase.com/post/the-elo-rating-system-correcting-the-expectancy-tables, 2011. Accessed: 2025-04-25.

[13] T. Tijhuis, P. B. Mavromoustakos, and P. Spronck. Predicting chess player rating based on a single game. In *2023 IEEE Conference on Games (CoG)*. IEEE, 2023.

[14] S. Toshniwal, S. Wiseman, K. Livescu, and K. Gimpel. Chess as a testbed for language model state tracking. *arXiv*, 2022. Accessed: 2025-04-25.

[15] R. M. Young, S. Sen, J. Kleinberg, and A. Anderson. Aligning superhuman ai with human behavior: Chess as a model system. In *ACM SIGKDD International Conference on Knowledge Discovery*, volume Data Mining, 2020.

A Appendix: Additional Figures and tabels

Features	Normalization Needed	Normalization factor	Encoding Needed	Encoding Type
Game level				
Estimated time	Yes	Estimated time of game type	No	
Total_Black_errors	Yes	Total moves in the game	No	
Total_white_errors	Yes	Total moves in the game	No	
Outcome	No		No	
Move Level				
Halfmove_count	Yes	Total moves in the game	No	
Move	Yes	Normalised when encoded	Yes	5 feature encoding
fen	Yes	Normalised when encoded	Yes	70 feature encoding
turn	No		No	
Is_error	No		No	
cp	Yes	divided by 1500 and clipped	No	
time_ratio	No		No	
Mate_in_n	Yes	divided by 20 and clipped	No	

Tabel 1

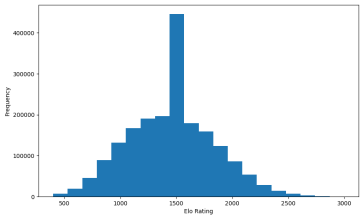

Figure 3: Elo Distribution

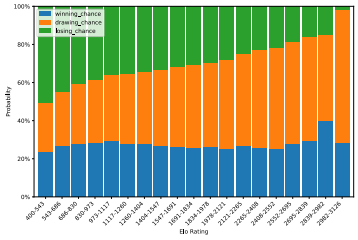

Figure 4: W/D/L distribution Vs Elo

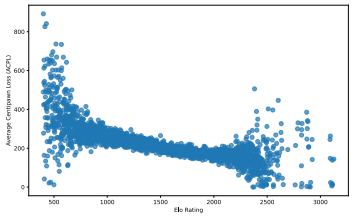

Figure 5: centipawn loss vs Elo

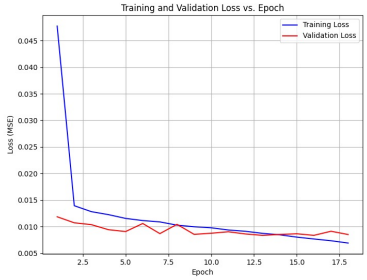

Figure 6: Train and Validation Loss