ChessRec: An Embedding-Based Recommender System for Chess Opening Suggestions

Juan José Alonso
Pontificia Universidad Católica de Chile
Santiago, Chile
jalonsol@uc.cl

Joaquín Domínguez
Pontificia Universidad Católica de Chile
Santiago, Chile
joaquin.dominguez@uc.cl

José Tomás Lledó
Pontificia Universidad Católica de Chile
Santiago, Chile
jtlledo@uc.cl

Figure 1: White Pieces on a Chess Board.

Abstract

Chess is one of the most popular board games in the world, with a **history spanning over 1,400 years** [8]. Its popularity continues to grow, with approximately **605 million adults playing the game regularly** [11]. One of the key aspects of chess is the opening phase: in the *Oxford Companion to Chess*, there are **at least 1,327 named openings and variations** [2]. This diversity highlights the strategic depth of the game but can be **intimidating for beginner players seeking to improve**.

ChessRec proposes a recommendation system designed to **facilitate player learning** through **personalized opening recommendations**. The system utilizes embeddings to analyze the user's game history, providing suggestions for openings that align with their playing style. Additionally, it considers factors such as chess engine evaluations, the performance of other players, and the popularity of openings.

The results demonstrate the **system's strong performance**, validated by robust evaluation metrics. Finally, the potential impact of this approach on skill development in chess and opportunities for future research are discussed.

Keywords

Chess, Recommendation, Personalized, Openings, FEN, Similarity, ELO, Embeddings

ACM Reference Format:
Juan José Alonso, Joaquín Domínguez, and José Tomás Lledó. 2024. *ChessRec*: An Embedding-Based Recommender System for Chess Opening Suggestions. In . ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Source Code

The source code for ChessRec can be **publicly accessed** at *ChessRec*'s GitHub repository. [1]

2 State-of-the-Art Review

To date, **no related work or recommender systems exist** that achieve what *ChessRec* is capable of: recommending personalized chess openings based on a player's past games.

However, there are some areas of research and tools that, while not directly aligned with *ChessRec*'s approach, address relevant topics within the fields of chess and recommender systems. These include:

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2.1 Chess Position Evaluation Engines

Tools such as **Stockfish**, **Dragon**, and **Komodo** are widely used to analyze chess games. Although these engines focus on evaluating positions and generating optimal moves, they **lack a personalized approach** that considers the play style or specific preferences of an individual player.

2.2 Generic Recommender Systems

Traditional recommender systems, like those used in e-commerce or streaming platforms (e.g., **Amazon**, **Netflix**), typically rely on algorithms such as **Collaborative Filtering**, **Content-Based Filtering**, or hybrid models. Although these algorithms have broad applications, **their adaptation to the domain of chess and personalized opening recommendations remains unexplored**.

2.3 Chess Opening Databases

Tools like **ChessBase** and platforms such as **Lichess** or **Chess.com** offer extensive databases of openings and related statistics (e.g., win rates, popularity). However, these platforms do not provide dynamic recommendations tailored to an individual player's characteristics.

2.4 Deep Learning Models Applied to Chess

Recent works, such as those involving **AlphaZero**, have demonstrated the potential of deep learning models to autonomously learn game strategies. While these approaches are impressive, they focus on optimizing overall gameplay rather than personalizing opening recommendations [7].

3 Proposed Solution to the Addressed Problem

To address the described scenario, *ChessRec* was developed, a chess opening recommendation model focused on personalization.

ChessRec positions itself as an **innovative solution** by integrating **modern embedding techniques** to analyze player games and generate personalized opening recommendations. This approach offers several **advantages over the state-of-the-art**:

- **Personalization:** *ChessRec* leverages a player's previous games to understand their playing style, skill level, and strategic preferences.
- **Dynamic Recommendations:** The recommendations are neither static nor solely based on global statistics. Instead, they are adapted to the player's context and time control.
- **Integration of Multiple Metrics:** *ChessRec* combines information from chess engines (position evaluations), historical win rates, and cosine similarity metrics, among others, to ensure robust and effective recommendations.
- **Scalability:** By utilizing embeddings, *ChessRec* can process large volumes of data and generate recommendations in real time, something traditional tools cannot achieve.

4 Dataset

This study utilizes a subset of data extracted from the public archive of games provided by **Lichess.org**, specifically from **July 2014** [5]. Lichess.org is a widely recognized free platform for playing and studying chess. It provides access to its game history, which is stored in **Portable Game Notation** (PGN) format. The dataset comprises a total of **1,048,400 games**, each including the following relevant features:

- **Event:** Name and type of the event.
- **Date and Time:** Precise start time of the game in UTC format.
- **Players:** Name and title of the players for both white and black pieces.
- **ELO:** Players' ELO ratings at the start of the game.
- **Result:** Outcome of the game.
- **ECO Code and Opening:** Classification of the opening according to the Encyclopedia of Chess Openings (ECO) and its name.
- **Time Control:** Time format of the game.

5 Methodology

To emulate the recommendation scenario, all games played in the last month are selected. For the development and evaluation of *ChessRec*, all games played on **Lichess** in July 2014 were chosen, comprising a total of 1,048,400 games. To narrow the scope of the recommendation context, only three types of game (Bullet, Blitz, Classical) played by white-piece players with an ELO between 600 and 1,600 were considered. This selection represented a total of 457,879 games, approximately 43.7% of all games played on Lichess in July 2014.

To properly modularize the recommendation context, the games were separated according to their time control (game mode). The distribution was as follows: 124,534 Bullet games (27.2% of the total), 178,306 Blitz games (38.9% of the total), and 155,309 Classical games (33.9% of the total).

During the month and for each player, the following data were extracted from the dataset: the variation in their ELO (calculated as the difference between the ELO of their last and first games of the month), their win rate (percentage of victories), and their most frequently used opening. Subsequently, for each opening, the average ELO variation and the win rate of the players who used it most frequently were calculated.

To extract as much information as possible from an opening (for example, pawn structure), embeddings were created using the Sentence Transformers library for Python, using a pre-trained model named `'all-mpnet-base-v2'` [6]. This model takes as input the FEN of the ten most common first moves of each player during the last month, as well as the FEN of all the relevant openings, producing a vector associated with each position as output.

Using all this information, the five openings most similar to the ten most common moves of each player were selected for each game mode (if applicable) and ranked based on a score calculated as follows:

$$\begin{aligned}\text{score}(p, o) = {} & a \cdot \text{cosineSimilarity}(p, o) \\ & + b \cdot \text{averageELODelta}(o) \\ & + c \cdot \text{winRate}(o) \\ & + d \cdot \text{popularity}(o) \\ & + e \cdot \text{engineEval}(o)\end{aligned}$$

Where:

- a, b, c, d, e are **weights**, whose values are defined for $R \in [0, 1]$.

- *cosineSimilarity*(p, o) represents the **cosine similarity** between the embedding of the player's ten most common first moves (p) and the embedding of a given opening (o). This similarity is defined as follows:

$$\text{cosineSimilarity}(p, o) = \frac{\vec{p} \cdot \vec{o}}{\|\vec{p}\|\|\vec{o}\|}$$

  where the value is defined for $R \in [0, 1]$, with higher values indicating a greater similarity between the two embeddings.

- *averageELODelta*(o) represents the **average ELO variation**, during the last month, of players whose most common opening in that period was opening o. This average is defined as follows:

$$\text{averageELODelta}(o) = \frac{1}{|P|} \cdot \sum_{p \in P} \Delta\text{ELO}(p)$$

  where the value is defined for R. To adapt the scale for the calculation of the *score*, min-max scaling is applied between 0 and 1. The normalized value is then defined for $R \in [0, 1]$, where a higher value indicates a greater ELO gain during the last month for players P whose most common opening was o.

- *winRate*(o) is the ratio between the number of times opening o was played in a game where the white pieces achieved victory and the total number of times opening o was played, regardless of the game's outcome.
  Its value is defined for $R \in [0, 1]$, where a higher value indicates a greater win percentage during the last month for opening o.

- *popularity*(o) is simply the **number of times opening** o **was played** during the last month. Its value is defined for R. However, as with *averageDeltaELO*(o), min-max scaling is applied between 0 and 1 to adjust the scale for the calculation of the *score*. The normalized value is then defined for $R \in [0, 1]$, where a higher value indicates greater popularity for opening o during the last month.
- *engineEval*(o) is a **score that quantifies how advantageous a position is** for the user based on an engine's evaluation and their ELO. This metric reflects the fact that more skilled players (higher ELO) can convert an advantageous position more often compared to less skilled players. To calculate its value, the position is first evaluated using an engine. Then, a bonus is calculated based on the player's ELO, determined by the formula:

$$\text{eloBonus} = 3 - \frac{2 \cdot \text{ELO}}{3000}$$

  For this value, an upper limit of ELO 3000 is set. This value is then used alongside the engine's evaluation to compute the final score using the following equation:

$$\text{finalScore} = \max\left(\min\left(\frac{\text{engineScore} + \text{eloBonus}}{2 \cdot \text{eloBonus}}, 1\right), 0\right)$$

  Its value is defined for $R \in [0, 1]$, where a higher value indicates a position more advantageous for the player.

Then, *score*(p, o) is a linear combination whose value is defined for $R \in [0, 1]$, where a **higher value indicates a better recommendation** (based on criteria such as similarity, expected ELO gain, win probability, popularity, and estimated pawn advantage after the opening is played).

Finally, this *score* is calculated for the five openings most similar to the ten most common first moves of each player, and the opening with the highest *score* is recommended. It is worth noting that the calculation of recommendations is unique for each time control.

To **evaluate the effectiveness** of the recommendations made by *ChessRec*, metrics were developed based on state-of-the-art chess engines, such as Stockfish, Dragon, and Komodo.

To **assess the personalization** of *ChessRec*'s recommendations, the cosine similarity metric between the most recommended opening and the user's most common moves is used. Lastly, the historical win rate associated with the opening is also included.

6 Parameter Analysis

When training the model to generate recommendations, several parameters influence the final recommendation. The most important ones are summarized below:

- **Number of User Moves for Embedding Generation:** Within the previously described methodology, the first ten moves of the user were considered for comparison with the openings. This number was proposed because most chess openings (excluding very extensive variations) involve a similar number of moves. Thus, comparing the embeddings of openings with the user's most common moves ensures that the general board structure's similarity can be captured. For instance, if three moves were used instead of ten, the overlap between many openings could prevent the familiarity of the position with the user from being properly captured.
- **Embedding Model Used:** To transform the FEN of positions, the `'all-mpnet-base-v2'` model from the Sentence Transformers library was used, as previously mentioned. This choice is based on the claim by the library's creators on their website [10] that this model performs best compared to other pre-trained models available in the library. While it is neither the fastest nor the most lightweight model, in this case, the priority is ensuring the correct similarity relationship between the opening and the user's moves. However, FEN chess positions are not natural language sentences, which are the primary focus of these models. For this reason, exploring other embedding methods more specialized for chess positions could be an interesting future direction.
- **Weights in the Scoring Function:** When an opening is scored for recommendation, the score equation is used. Five weights are evident in this formula, detailed as follows:
  a) **Cosine Similarity Weight:** This weight was considered one of the most relevant in the formula, with a value of

0.3. This is primarily because ChessRec aims to recommend highly personalized openings, and therefore greater influence was given to the cosine similarity between the opening and the user's most common moves.

b) **Average ELO Delta Weight:** Similar to the previous weight, this weight is assigned a value of **0.3**. This value accompanies *averageELODelta*, a factor that reflects an opening's ability to improve a user's skill level. ChessRec seeks to ensure that a recommended opening can help a player improve their abilities over time, reflected in their ELO. For this reason, a higher weight is assigned to this metric in the scoring function.

c) **Win Rate Weight:** The weight related to the win rate is set to **0.15**. This value was chosen because it is important for an opening to have been effective for previous users. The primary goal of an opening is to provide future benefits to the player.

d) **Popularity Weight:** Associated with the number of times an opening has been played, this weight is assigned a value of **0.1**. This is the lowest value within the function since high popularity does not necessarily ensure good player performance when the opening is used. However, the frequency of an opening validates that other players have used it and that it is a viable option in the game context. For instance, an opening might be highly advantageous but historically rarely reached, indicating it may not be the best recommendation.

e) **Engine Evaluation Weight:** Finally, the weight for the engine evaluation is set to **0.15**. This value closely relates to point c, as ChessRec seeks to ensure that the opening provides benefits for the user. The engine evaluation offers an objective perspective on the advantages of an opening and therefore should hold medium relevance in the scoring function.

7 Obtained Results

One of the main challenges in recommending chess openings was defining metrics that could reliably determine whether a recommendation was good. Conventional recommender system metrics could not be directly applied in this context, as many of these rely on user feedback to assess whether the recommendation was actually used. In the context of *ChessRec*, an opening can be recommended, but it does not depend entirely on the user whether the recommended position is reached, as chess is a two-player game, and the opponent is not obligated to follow the recommendation.

For this reason, four metrics were designed to evaluate the recommendations made by *ChessRec*:

- **Stockfish Evaluation:** Using the engine evaluation formula, this metric assesses how advantageous the recommended position is according to the Stockfish 17 chess engine. [9]
- **Dragon Evaluation:** The same procedure as with Stockfish is applied, but the evaluation is performed using the Dragon 1 engine, which is known for evaluating chess positions in a manner closer to human reasoning. Chess engines often identify advantages in positions that are challenging for humans to interpret, and Dragon takes this into account when evaluating a position. [3]
- **Komodo Evaluation:** To provide as many objective evaluations as possible, the Komodo 14 engine is also included. [4]
- **Win-rate:** From the original dataset, information about the win rate of white pieces in the recommended position is extracted. This metric measures whether the recommended opening has historically been effective for other users.

Based on these defined metrics, the recommendations made by *ChessRec* were evaluated. The results are presented in the following table:

Metric	Bullet	Blitz	Classical
Stockfish	0.77	0.80	0.73
Dragon	0.81	0.86	0.78
Komodo	0.81	0.84	0.73
Win-Rate	33.3%	34.375%	37.54%

Table 1: Performance metrics for *ChessRec* across different time controls

As evidenced, *ChessRec* performs well according to the defined metrics, as the recommendations are evaluated favorably by the selected engines. However, concerning the win-rate, a value of approximately 35% is surprisingly low and will undoubtedly become a focus for improvement in future iterations of *ChessRec*. One possible explanation for this low metric value could be the relatively low weight assigned to it in the scoring function, as other factors were prioritized within the formula.

On the other hand, it can be inferred that, for faster-paced game modes (such as Bullet and Blitz), *ChessRec* recommends openings that provide an early advantage to the player. In contrast, for traditional game modes (such as Classical), *ChessRec* suggests openings that do not necessarily grant an early advantage but instead prioritize strategic positioning. This leads to a better win rate, as can be observed when comparing the win rates of Classical games with those of Bullet and Blitz.

8 Conclusions

In this study, *ChessRec* was developed as a recommendation system for chess openings based on personalized embeddings. The results demonstrate that *ChessRec* is capable of suggesting openings similar to those used by the player, while also incorporating evaluations from various chess engines, enhancing the relevance and effectiveness of the recommendations. This work contributes to strategic learning in chess by providing personalized recommendations tailored to players of different skill levels and various time formats.

Among the main limitations of *ChessRec* are the metrics used to evaluate its recommendations. Although multiple metrics were employed to assess its performance, the most compelling evidence of its effectiveness would involve analyzing the direct impact on a

user's win rate after following a recommendation—an aspect that could be explored in future studies.

As another potential avenue for future research, the integration of real-time analysis is proposed, allowing recommendations to be adjusted as the game progresses. This would enable users to apply openings or strategies suggested by *ChessRec* in dynamic contexts, further expanding the system's utility and applicability.

Acknowledgments

To Daniel Sebastián, for supporting us throughout the project, for offering his valuable perspective, and for his approachability.

References

[1] Juan José Alonso. 2024. *ChessRec.* https://github.com/jalonsoluc/chessrec
[2] David Hooper and Kenneth Whyld. 1992. *The Oxford Companion to Chess* (2 ed.). Oxford University Press. 461–480 pages.
[3] Komodo Chess. 2020. *Dragon 1.* https://komodochess.com/
[4] Komodo Chess. 2020. *Komodo 14.* https://komodochess.com/
[5] Lichess. 2014. Lichess Database: Standard Rated Games for July 2014. https://database.lichess.org/ Accessed: 2024-12-08.
[6] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing* (2019). https://www.aclweb.org/anthology/D19-1410/ Retrieved from https://sbert.net/docs/sentence$_t ransformer/pretrained_m odels.html.$
[7] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. 2018. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science* 362, 6419 (2018), 1140–1144. https://doi.org/10.1126/science.aar6404
[8] Colin Stapczynski. 2023. *History of Chess — From Early Stages to Magnus.* Technical Report. Chess.com.
[9] The Stockfish developers. 2024. *Stockfish.* https://stockfishchess.org/
[10] Sentence Transformers. [n. d.]. Pretrained Models. https://www.sbert.net/docs/sentence_transformer/pretrained_models.html. Accessed: 2024-12-08.
[11] Alexander Walton. 2024. *How Many People Play Chess? A Guide to the Numbers.* Technical Report. The House of Staunton.

Received 9 December 2024