GMR
The Original

AN EXPLAINABLE AND REPRODUCIBLE ARTIFICIAL INTELLIGENCE BASED DECISION-SUPPORT FRAMEWORK FOR CHESS MOVE EVALUATION

K. Venkata Vasavi[1], Mahanty Rashmita[2], Kakarlapudi Udaya[3], Subhendu Sekhar Sahoo[4], R. S. S. Srikanth Vemuri[5*]

[1]Department of Basic Sciences and Humanities, B V Raju Institute of Technology, Hyderabad, Telangana, India.
Mail Id- venkatavasavi3@gmail.com , Orcid Id- 0009-0003-3891-7762
[2]Department of Basic Sciences and Humanities, Vignan's Institute of Engineering for Women, Visakhapatnam, Andhra Pradesh, India. Mail Id- rashmita.mahanty@gmail.com, Orcid Id- 0000-0001-9247-8295
[3]Department of Basic Sciences and Humanities, Nadimpalli Satyanarayana Raju Institute of Technology, Vishakhapatnam, Andhra Pradesh, India. Mail Id- kakarlapudiudaya2@gmail.com, Orcid Id- 0009-0000-6781-2204
[4]School of Electrical Engineering, Vellore Institute of Technology, Chennai, Tamil Nadu, India. Mail Id- subhendukunar.sahoo@vit.ac.in, Orcid Id- 0000-0002-9461-8252
[5*]Department of Basic Sciences and Humanities, Vignan's Institute of Engineering for Women, Visakhapatnam, Andhra Pradesh, India Mail Id- vrsssrikanth@gmail.com, Orcid Id- 0000-0003-4243-9102

ABSTRACT

The present work provides an AI-driven move-level chess analysis decision support framework which is transparent and repeatable. The framework incorporates the Stockfish chess engine as a quantitative evaluation component and adds a structured analytical layer which transforms the engine evaluations into interpretable move classification and human-readable explanations. During each move, the framework is able to compute the centipawns changes of evaluation by normalizing the scores with respect to the mover and to categorize the moves within the pre-defined classes, with the use of fixed and well-documented thresholds, such as good, neutral, inaccuracy, mistake, and blunder. The evaluation changes are connected to observed properties of the board, e.g. material changes, captures, checks, and the size of the evaluation, by using a deterministic rule-based explanation layer. A deterministic material-based fallback mechanism is also provided which ensures analytical operations in the absence of engine evaluation. The modular design allows the separation of position parsing, engine evaluation, position classification, explanations, and visualization, enabling transparency and reproducibility. The operation of the framework is demonstrated through a representative example of playing chess and repeated run consistency checks in an experimental analysis. The results show that output of evaluation, classification, and explanation are consistent when the input and configuration are identical. However, all the proposed explanations are limited to the specific characteristics of the board that can be seen and these explanations are not explanations of the kind of thought that would happen in the mind of the chess engine itself. Similarly, leveraging the hard-coded cut-off for the delta for the centipawns per move was not reflective of context-dependent and/or nonlinear changes in position, and the material-based fallback may not have been an adequate measure of tactical sacrifices. The framework was then put in place as a "rule-based" decision support system instead of a model-level Explainable AI system.

KEYWORDS: Artificial intelligence; Decision support systems; Explainable artificial intelligence; Chess move evaluation; Engine based analytical framework; Reproducible analytical systems

1. INTRODUCTION

Chess has been used to address sequential problem solving, strategic reasoning and complex decision making for a long time [1]. Superhuman performances are now possible with chess engines thanks to advances in artificial intelligence; they are capable of evaluating the position on a chessboard with a high degree of accuracy and recommend the best possible move [2,3]. These engines are extensively used in post-game analysis and performance evaluation for players in modern analysis platforms [10,11]. Although powerful, these tools mainly provide numerical assessments or suggested actions, with little assistance for human interpretation and learning [7].

From engineering perspective, this limitation is indicative of a larger problem in applied AI: Adequate performance does not guarantee explainable or repeatable decision support. Engine scores are an index of positional advantage but they don't necessarily reflect the positive or negative effect of a move, nor do they necessarily capture the quality of decisions made over the course of a game [4,5]. Furthermore, commercial chess analysis programs are normally proprietary black-box programs, using unknown thresholds, different engine configurations, and limited reproducibility, which limits their use in scientific and engineering applications [10-12].

Previous studies on chess analytics have been mostly concerned with how to design chess engines, how to optimize the search algorithm of chess engines, or how to get large scale statistical data of player performance [2,4,6]. The use of engine score drops has been used in several studies to detect blunders or model skill levels, with a focus on aggregate numerical measures, instead of move-level interpretability or transparent explanation [4-6]. At the same time, explainable AI (XAI) studies have come a long way in understanding ML models [7,8] but very little has been done to explain a symbolic decision analysis (decision rules) process in a rule-governed domain like chess, especially for post-game analysis [9].

In view of these observations, this work takes up the following question for research:
What is the way to convert chess engine evaluations into explainable, reproducible and human-interpretable decision support framework for engineering and analysis?
In this solution, we design an explainable AI-based chess move analysis system which focuses on interpretation and reproducibility of moves rather than optimization. This framework provides a clear analytical level that applies mover-relative normalization and explicit rule-based classification to interpret shifts and changes in evaluations between successive moves [4-6]. The system does not use black-box or data-driven methods but produces human-readable explanations based on the properties of the game that can be observed [7].
In this work “explainability” means the layer of interpretability which is placed on top of the engine output – it's not being explained what the Stockfish engine does internally when looking for a move. The framework suggested here is used to produce human-readable interpretations explicitly defined and deterministic, based on changes in the evaluation, and on the observable properties of the board, while Stockfish is employed as a benchmark for quantitative evaluation components. The explanations to be expected from the framework should thus be taken as rule-based interpretations of engine evaluated moves, and not as model-intrinsic explanations of an artificial intelligence system. This is significant because the framework itself does not change, check or reveal the working of the underlying chess engine.
One of the key design goals of the proposed framework is reproducibility. The parameters of the engine are set and will not be altered: fixed engine parameters are used to guarantee consistent analytical behavior, and a deterministic material-based fallback evaluation is added to cases where it is not possible to execute the engine [3,12] The system also utilizes a modular design with parsing, evaluation, classification and visualization components, enabling extensibility and reusability of the system in engineering oriented AI applications [9,12].
The novelty of this work is not the development of a new chess engine nor a new chess centipawn evaluation, but rather the design of a "transparent, replicable, analytical layer" that will systematically convert the existing well-known engine evaluations into explanations of how good or bad each move is. In addition to numerical evaluations or suggested moves, the proposed framework also has four explicitly defined mechanisms in one analysis pipeline: (i) Normalization of the change of the evaluation, relative to the mover, to be able to interpret the impact of moves uniformly for both players: (ii) Fixed rules for classifying individual moves based on the amount of evaluation loss; (iii) Deterministic, human-readable explanations, linked to the properties of the board; (iv) Deterministic fallback mechanism based on material, so that the pipeline still functions if an engine cannot be used to execute it. These are realized as individual parsing, evaluation, classification, explanation and visualization units, thereby making it possible to analyze and reproduce the analytical process. Therefore, the contribution should be a clear and open framework of decision support based on the existing chess-engine technology, and should not be a change to the chess-engine itself.
Interactive visual analytics are also part of the framework and allow users to explore evaluation trajectories, to perceive how the momentum changes throughout a game and to identify the critical decision points [9]. This approach shows the potential of making chess a representative domain of decision-making and how an explainable and reproducible analytical system can fill the gap between raw AI evaluation and human understanding. The method itself was used to validate with chess games, but the methodology can be applied to other sequential decision problems where transparency and reproducibility are crucial.

2. RELATED WORK

The proposed system touches on several areas of research such as chess evaluation, move quality analysis, decision analytics, and explainable systems. This section presents the literature review relevant to the present study and also clearly differentiates the scope and position of the present study.

2.1. Chess Engines and Centipawn Evaluation

The field of chess-engine research has been well studied for several decades, and the earliest research papers dealt with search strategies and evaluation functions [1, 2]. The evaluations that modern engines like Stockfish give are in "centipawns," which are the relative gain of one player over the other in the form of hundredths of a pawn [3]. Chess engines are now commonly using quantitative chess evaluation methods like centipawns [3, 4] to compare chess positions and to make decision on what action to take in a chess position.
Centipawn scores are commonly used and accepted for competitive play and analytical studies [4]. The idea of centipawn differences occurs within the engines where they are used to rank candidate moves and to direct search heuristics [2, 3]. These, however, are generally used for engine optimisation and performance, not for humans to read or for educational feedback [7].
This work does not implement a new metric for evaluation, nor alter the engine internals. Rather, it is based on a proven quantitative framework for higher-level interpretation that is the centipawn-based evaluation framework [3, 4].

2.2. Move Quality Assessment and Blunder Detection

Engine evaluations have been used in several studies to evaluate move quality and player performance [4,5]. Evaluation intervals between consecutive positions are particularly used to detect blunders, mistakes, or inaccuracies, and to estimate the skill level of the players in a large amount of data [5, 6]. These are typically used in statistical analyses of tournament games or online game repositories that take the form of aggregate metrics of how often the errors occur or the average evaluation loss incurred [6].
The studies in this paper show that the measurement of centipawn differences can be useful for quantitative assessment, but they generally require an offline, batch processing environment [4, 6]. This is on numerical indicators and statistical trends, rather than human-facing explanation and interactive analysis [5]. Furthermore, there are not always explicit descriptions of the classification criteria and thresholds applied in such analyses, which reduces transparency and reproducibility [12].

More recently, the necessity for explanations at the concept level and domain-specific level have been emphasized. Representable images can transform the low level computational information to the high level concepts, that are more meaningful to high level human users. For reinforcement-learning systems, explainability research is also about learning about the decisions a complex system makes and finding concepts that are relevant to humans in the behaviors of the system. In this context, the present framework will be discussed. However, the proposed approach is not model level XAI methods as there is no attempt to explain the decision process or the internal representation of the chess engine. Instead, it provides a rule-based and deterministic interpretation of engine generated evaluations [13].

The system that is proposed, on the other hand, takes a more explicit and rule-based approach to the differences in evaluation whereby individual moves are associated with interpretable explanations [7]. This distinction allows a transition from a purely numerical evaluation to an evaluative interpretation and user understanding shift.

2.3 Commercial Chess Analysis Platforms

There are many commercial platforms available, for example Chess.com and Lichess, that offer an automated game analysis feature which tags moves as blunders, mistakes, or good moves [10, 11[. The platforms are based on robustly built chess engines as well as evaluation disparities to provide feedback to users [10]. But their underlying algorithms, thresholds, and evaluation settings are proprietary and undisclosed [11].

Recent research has also been done in the area of decision support, an area in which AI systems are not just prediction tools, but part of a human-AI decision support system. The current research in the area of HAI is centered around systems that combine computational power with human interpretation and interaction. Likewise, recent work on explainable AI for decision making states that explanations need to assist the user in comprehending and assessing AI-generated recommendations, and not just present outputs. The views are relevant to chess analysis, as the goal of this proposed framework is not to replace the player's decision making, but to provide the player with a chess analytical "layer" that can be viewed and understood by the player to gain insight into the engine's evaluation.

On the research side, these platforms are black box systems [7]. Any changes in their analysis behavior, such as changes in the versions of the engines they evaluate, or varying depths of analysis, or changing heurs, can change their results and they are not guaranteed to be repeatable [12]. Moreover, explanations given to users tend to be short and qualitative, giving no explicit information about the logic behind the move classification [10, 11].

Consequently, commercial platforms do not provide scientific references for method design and evaluation, but they are tools which could be used for practice purposes. This work is a radical departure from the former in providing a clear, repeatable, and well documented analytical framework applicable to research and engineering problems.

2.4. Explainable Decision Analysis and Visualization

In recent years, explainable artificial intelligence (XAI) has become an important research field that focuses on enhancing transparency and trust in an algorithmic system [7, 8]. Current research on XAI is mostly dedicated to the task of explaining the predictions of machine learning models, e.g., feature attribution or surrogate modelling [8]. Much less has been done to address interpretable decision analysis in symbolic or rule-governed domains, where interpretability can be achieved in the form of explicit logic, rather than model introspection [7, 9].

At the same time, visualization methods have been extensively employed in decision analytics for presenting dynamics, trends and anomalies [9]. A common approach to visualizing the variation in performance or momentum shifts in sequential decision processes are graph-based representations and heatmaps [9].

The proposed system is inspired by these approaches, where the explicit rule based explanation is integrated with visualization of evaluation trends. Unlike the normal XAI frameworks, the system does not try to explain a learned model. Rather, it aims at clarifying the result of decisions based on well-defined evaluation criteria within the context of a clear analytical process [7, 9].

2.5 Summary and Positioning

To recap what has been found in previous studies and chess engines, there are several methods of evaluating chess positions, and the most popular ones are centipawn based and evaluation drop based [3, 6]. Their usefulness has been shown in existing studies for performance assessment and statistical analysis, and have become popular in commercial platforms in user-facing tools [10, 11]. However, the previous studies have focused primarily on optimization, aggregated metric, or a black-box feedback [7, 12] rather than reproducible, explainable, and modular system design.

From the literature emerged two interesting research lines based on the findings: model level explainability, aimed at understanding more the inner workings of complex AI systems, and human level decision support, aimed at taking computational results and presenting them in an easy to understand format to the user. Recently, the concept of internal representation and concept-level explanations for neural chess agents have begun to be studied in the context of chess, and the general research area of XAI has centered on determining the quality of the explanation, transparency and user-relevance. The present work is located in another location within this landscape. Does not attempt to describe how the internal logic works for a neural chess model, or how to modify the chess engine. Instead, it provides a deterministic analytical layer, that decomposes well-known engine evaluations into explicitly defined move classifications and readable interpretations. The discussion is thus on transparency and reproducibility of decision making, rather than XAI intrinsic to the model.

The present work is unique in providing a single framework for analysis, combining engine-based evaluation, deterministic fallback mechanisms, explicit moves classification rules, and interactive visualization [7, 9, 12]. The proposed system emphasizes the importance of interpretability and reproducibility and provides a bridge between the raw engine output and the human-centered understanding, making it a more decision-support and learning-oriented analytical tool rather than a competitive engine or proprietary service.

Another explanation is in the context of the current framework, the term explainability is used. The proposed system doesn't attempt to describe how and what Stockfish thinks, or how it represents its knowledge, and it doesn't claim to make

the engine interpretable. Instead, the idea of explainability is fulfilled in the form of an explainability rule-based layer. The framework specifies the numerical results of the engine's computations, and certain properties of the board to pre-established human-readable explanations using deterministic rules. In this study, the word explainable rather means the transparency and traceability of the analytical interpretation process, and not explainability of the underlying model of the engine. This is different from model-level Explainable AI techniques that attempts to explain the action of a learned predictive model.

3. System Architecture and Implementation

In this section, the overall architecture and implementation details of the proposed system for analyzing chess games with the aid of an explainable approach are discussed. The overall architecture and implementation details of the proposed system for analysing chess games with the aid of explainable approach are discussed in this section. The design is modular and layered, with different parts dealing with different aspects of data processing, evaluation logic, and visualization. This separation allows for increased clarity, reproducibility, and extensibility—all requirements for an engineering-oriented analytical system [9,12].

3.1 Overall Framework

It takes a full chess game as input, and outputs an analysis that can be understood by humans, via a pipe-line for each step. The overall architecture of the framework is shown in figure (1).

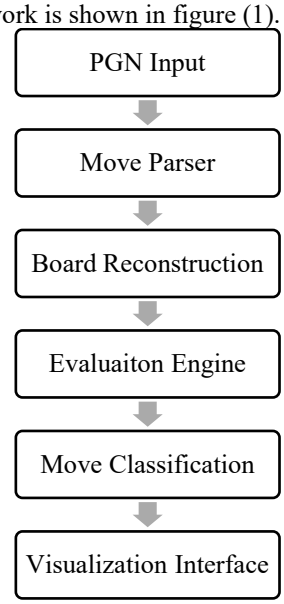


Fig. 1. Overall architecture of the proposed explainable chess moves analysis framework.

The system consists of three modules – parsing, evaluation, classification and visualization.
The processing flow has the following steps:

1. Input Parsing: Parsing and normalisation of the game data, in Portable Game Notation (PGN) or Standard Algebraic Notation (SAN).
2. Board Reconstruction: The chessboard configuration is built up step by step, according to the moves made.
3. Position Evaluation: Evaluation of each reconstructed position is done through an engine-based or fallback evaluation model [3] is used.
4. Move Evaluation differences are converted to move classifications in a qualitative manner [4 – 6].
5. Visualization and Interaction: Analytical results presented interactively in visual elements [9].

The advantage to the engineer is the ability to isolate the computation portion of the engine from the interpretation portion. The engine is seen as a quantitative evaluation provider, while the proposed framework automatically carries out the normalization, classification, explanation and visualization. This separation lets the analysis and replication of every analysis stage without altering the chess engine. The framework, therefore, explicitly traces from the evaluated position on the board to the final interpretation of the move-level, the main design element of the proposed approach to decision support.
Each stage of the pipeline works independently but adds to the process of analysis.

3.2 Input Parsing Module

Input parsing module is to extract valid chess moves from the user-provided text. Both PGN games and raw SAN move lists can be played. Player names, event data and player notes are not used when parsing.
A rule-based parser is used to extract the move numbers, comments, and annotations from the move and then identify legal SAN tokens with the move extraction. This pre-processing step guarantees independence in input format and reduces the risk of parsing errors [14].
Separating parsing from the system components enables alternative input sources and notation formats to be added without changing downstream components [12].

3.3 Board Reconstruction Module

The proposed framework in this paper provides a deterministic rule base interpretation layer, which is able to map engine evaluations and observable board-state information to human-readable explanations. This layer is not meant to be considered a description of the internal logic of the underlying chess engine. Instead, Stockfish is a provider of quantitative evaluations and this is the framework that understands the changes in evaluation without the need for external help.

The framework considers the change in evaluation for each evaluated move as well as the observable attributes of the move such as material balance, captures, checks and the size of the changes in the move's evaluation. These properties are utilized to create explanations that are predetermined, such as loss of material, failure to give a check or any major drop in engine evaluation. The Rules are well specified, leading to a consistent roll action and consistent roll description, thus providing clarity and repeatability.

The resulting explanations are then interpretable explanations for why a move is classified in a certain way in the framework. They are not an attempt to model Stockfish's decision process at the model level, or a Stockfish modelling attempt of Explainable AI. It is more specific to the part of the system discussed in this that it is called “rule-based interpretability” or “rule-based explanation”.

All subsequent steps of evaluation and visualization are based on the reconstructed board state.

3.4 Evaluation Engine Module

The evaluation engine module gives a numeric "score" to every reconstructed board position. The system has an ‘up and down’ approach to assessment:

1. Engine-Based Evaluation: Evaluation using a chess engine that is able to calculate centipawn scores at a fixed depth of analysis [3]. This gives a good and accepted quantitative measure of position quality.
2. When execution of the engine is not permitted, a model based on materials is used for a deterministic evaluation. This is a backup that calculates the difference (weighted) between the white material and the black material and returns it in centipawns

The fallback mechanism keeps the analytical pipeline running and reproducible in various deployment scenarios [12]. Importantly, the evaluation module is not used to change the way the game is played, or the way that points are awarded for moves, but it's only being used for interpretation after the game[15].

3.5 Move Evaluation and Classification Module

The move evaluation and classification module is the core of the system, which is an analytical module. The move evaluation and classification module is an analytical module that forms the core of the system. It takes the scores that the evaluation module returns for each position, and calculates the differences between them, normalizing them to the player who took the move [4 – 6].

The changes to the evaluation, normalized by the above, are passed to a rule-based classification component, which assigns qualitative labels, e.g. blunder, mistake, inaccuracy, good move, based on fixed thresholds. The logic used and the thresholds used for classification are made explicit and are the same for all analyses, which contributes to transparency and reproducibility [7].

The module directly connects with the explainability layer that produces human-readable explanations from the observable characteristics of the game [16].

3.6 Visualization and User Interface Module

The visualization module displays the analytical results in a human-readable format. The system uses several complementary visualization methods such as:

- Evaluation Curves are used to display the centipawns' development during the course of the game [9].
- Momentum Heatmaps showing evaluation changes relative to the momentum.
- Board-Level Visual Cues to highlight key moves (arrows and/or highlights).

Developed the user interface using Streamlet to create interactive analytical dashboards quickly. Streamlit is used just for the presentation layer, and does not impact the underlying evaluation/classification logic.

Interactive controls enable navigation through moves, show individual positions and correlate numerical evaluations with changes at the board level [17].

3.7 Implementation Details

Implements the system in Python with popular open source libraries. The chess library [3] is used to provide the board representation and move validation features of the chess game. Standard plotting/rendering libraries are used to implement visualization components [9].

All the modules will be deterministic, having fixed parameters and no stochastic components. This guarantees that the same input gives the same analytical results on repeated runs, necessary for reproducibility [12].

In summary, the proposed architecture provides a unified framework for parsing, evaluating, classification and visualization, while adhering to separation of concerns. This separation of engine evaluation from interpretive logic and presentation allows for transparency, extensibility, and reliable analytical behavior, providing a solid basis for the case study and experimental analysis presented in the following section.

4. Move Evaluation and Classification Logic

This section outlines the thinking process behind analyzing and understanding single chess moves. The proposed approach does not seek to find the optimal move, but rather to understand what moves have been entered: it is a scheme of transforming evaluations that an engine carries out into classifications that a human can understand. The design is designed to be transparent, reproducible and not too dependent on black-box heuristics [7, 12]

4.1. Position Evaluation Model

Let $B_i$denote the board position after the i^{th}ply of a game. Each position is assigned an evaluation score E_i, expressed in centipawns (cp), representing the advantage from White’s perspective [3].

The evaluation function is defined as:

$$E_i = \begin{cases} E^{engine}(B_i), & \text{if engine evaluation is available} \\ E^{material}(B_i), & \text{otherwise} \end{cases}$$

where $E^{engine}(\cdot)$denotes the centipawn evaluation returned by a chess engine configured with fixed parameters [3], and $E^{material}(\cdot)$denotes a deterministic material-based fallback evaluation. The fallback mechanism ensures consistent behavior in environments where engine output is unavailable, thereby supporting reproducibility [12].
The material-based evaluation computes the weighted difference between White and Black pieces using standard piece values and expresses the result in centipawns [4].

4.2 Evaluation Difference Metric

To assess the quality of a move, the system analyzes the **change in evaluation** between consecutive positions. Let $\Delta_i$denote the raw evaluation difference after ply i:

$$\Delta_i = E_i - E_{i-1}$$

Since evaluations are expressed from White's perspective, the impact of a move must be normalized with respect to the player who made the move. Let $M_i \in \{White,Black\}$denote the mover at ply i. The mover-relative evaluation change $\Delta_i^{(m)}$is defined as:

$$\Delta_i^{(m)} = \begin{cases} \Delta_i, & \text{if } M_i = \text{White} \\ -\Delta_i, & \text{if } M_i = \text{Black} \end{cases}$$

This normalization ensures that negative values of $\Delta_i^{(m)}$consistently indicate a deterioration of the mover's position, regardless of color [4 – 6].

4.3 Rule-Based Move Classification

Based on the mover-relative evaluation change, each move is assigned a qualitative label. The classification is performed using explicit, fixed thresholds expressed in centipawns. Let $\Delta_i^{(m)}$denote the normalized evaluation change for move i. The classification rules are defined as:

- **Blunder**: $\Delta_i^{(m)} \leq -200$
- **Mistake**: $-200 < \Delta_i^{(m)} \leq -80$
- **Inaccuracy**: $-80 < \Delta_i^{(m)} \leq -30$
- **Good Move**: $\Delta_i^{(m)} \geq 30$
- **Neutral**: otherwise

The thresholds are roughly equal to pawn equivalents and follow a similar convention to what is typically used in chess analysis [4 – 6]. Unlike generally commercial systems, thresholds are clearly stated and are set once [10,11].

4.4 Explainability Layer

An explanation in human-readable text is provided for each classified move, to improve the interpretability. The explanation is created via a rule-based reasoning layer that takes into account observable properties of the game, such as:

- Change of material before and after move.
- Whether the move was a capture
- The move was made whether in check.
- The amount of change in the evaluation

For instance, a move that is deemed a blunder can be justified as:
"Blunder: the move resulted in a material loss of 2 pawns and had an evaluation drop of 240 centipawns."
This explanation strategy does not depend on enigmatic heuristics or acquired models, and gives users a direct view on the factors that affect the quality of the moves [7, 9].

4.5 Determinism and Reproducibility Considerations

This section addresses the issues of determinism and reproducibility.The issues of determinism and reproducibility are addressed in this section.
All the elements of the logic for evaluating and categorizing moves are fixed. Fixed-depth analysis is used to get engine evaluations and static piece values are used for fallback evaluation. Stochastic and adaptive elements or thresholds are not used. Thereby, the same game run on the same conditions will produce the same result on repeated runs [12].
In contrast with commercial analysis tools the output of which can change depending on changes in the engine configuration or version [10 – 12], this design decision reflects the principles of scientific software.
Overall, the move evaluation and classification logic turns raw engine evaluations into interpretable, mover-relative assessments that are clearly formulated and based on rules. This logic is central to the methodological backbone of the framework proposed for chess analysis that is transparent and reproducible, and the methodological backbone of the empirical analysis presented in the following section [18].

5. Visualization and User Interaction

This section presents the mechanisms of visualization and interaction used to present the analytical results obtained from the proposed framework. The use of visualization is not considered as a major contribution but as a means of making the numerical evaluation more interpretable, following best practices in analytical decision support systems [12,13].

5.1 Design Objectives

Experimental evaluation includes assessing the correctness and repeatability of the operations of the proposed framework but also the interpretability of the results of the operations at the move level. This extra evaluation is performed to determine if the rule-based explanations offer information which is easily traceable from the underlying engine evaluation and observable board state. If there exists similar material in the chess analysis websites that are widely used on the internet, the proposed framework is compared with them to get the similarities and differences in the classification of moves and the explanation of them. It's not a claim that the two chess engines and/or algorithms are the same, it's a comparison as a functional and interpretative comparison [19].

5.2 Evaluation Trend Visualization

The absence of blunders, mistakes and inaccuracies in the game chosen between Kasparov and Topalov from the proposed move classification system should be seen as a part of the selection criteria. The framework uses preconfigured "mover-relative centipawn loss thresholds" (Section 3.2) to classify moves. So, if the deterioration is less than the inaccuracy, mistake, or blunder threshold, then this is not an inaccuracy, mistake, or blunder. The lack of these categories in this specific game does not mean that there were no moves that were better than those shown in the game, but that they were not worse than the ones shown during this game's analysis, as per the parameters of the game engine and analysis depth used [13].

5.3 Move-Level Impact Representation

Discrete markers or heatmap style visualizations are used to visualize the changes in the mover-relative evaluation. These visualizations highlight the direction and amount of movement from each move.

Blunders and mistakes are marked in color, giving the user a high level view of the areas of the game that are important. This is used in conjunction with numerical evaluation, to immediately give visual clues, but without changing the analytical results.

5.4 Board-Centric Visualization

Additional steps towards increased reproducibility involve making the implementation and experimental resources available to the public. Any implementation, settings and input game data and evaluation scripts should be published in a public repository and made available to third parties to allow for the reproduction of reported experiments. Source code for the parsing, engine evaluation, move classification, explanation and visualization modules should be included in the repository with the parameters and instructions for performing the analysis again. The repository can also link to external resources and scripts to access the necessary external software or data set if that requires a license restriction [12].

5.5 User Interaction Workflow

The user interaction process is linear and controlled:

1. Entering of game notation.
2. Automatic running of the analytical pipeline.
3. Move navigation that is inter-active.
4. Qualitative classification and numerical evaluations inspected.

The use of interactive elements is limited to avoid the possibility of misunderstanding or manipulation by the user of the analytical results, consistent with the requirements of reproducibility for experimental software systems.

5.6 Comparative Interpretability Evaluation

The move-level analysis in the proposed framework was also analysed with reference to the move-level analysis available in popular chess-analysis websites, to further explore the practical interpretability of the proposed framework. The comparisons were based on information presented to the user, not on the engine itself because the same engine can be used in different platforms, with different search configurations, evaluation settings and presentation mechanisms [20].

The proposed evaluation of each position selected, along with a change in the evaluation from the mover's previous position, the resulting move, and the rule-based explanation proposed, is outlined for each selected position. The analysis of the selected online platform is then analyzed, including the evaluation of the move, the assessment of the move and the explanation, in text or image, of the moves. The comparison therefore considers three aspects: (i) general evaluation of the move is coherent, (ii) there is an explicit basis for this evaluation, (iii) the connection between the numerical evaluation and the classification is explicit.

The proposed framework differs from typical engine-analysis interface in the explicitness of the interpretation process. Three change thresholds – change threshold, classification threshold and explanation threshold – are explicitly specified and can be verified by the user. This allows for a connection to be made between the numerical value of the engine's output and the human readable explanation. The comparison isn't to the precision of the engine assessment, but whether the analysis transform is transparent and repeatable.

5.7 Implementation Considerations

It is created using a lightweight web-based framework, which allows for fast rendering and cross-platform compatibility. The presentation layer is distinct from the evaluation and classification layers, so that there is no interaction between the user presentation layer and the evaluation or classification layers that would impact analytical results [21].

All the visual aspects are explicitly created from analytic outputs that have been stored in advance, in order to ensure consistency between the interactive demonstrations and the experimental results reported in Section 6.

Overall the visualization and user interaction parts are good in making it easy to explore analytical results without any mixture of evaluation logic and presentation. The visualization layer emphasises clarity, determinism and the consistency with experimental results, while maintaining the engineering rigor of the proposed framework, thus naturally suggesting the experimental validation outlined in the next section.

6. Experimental Demonstration and Results

In this section, experimental proof of the proposed framework for analysis of chess moves with explanations is provided. The goal of the experiments is to test the correctness, interpretability and consistency of the move evaluation and classification logic presented in Section 4, not to evaluate how well they play competitively. As with previous analytical studies conducted based on the evaluations of the engines themselves, emphasis is not on the optimization performance but rather on interpretive behavior and reproducibility of the engine evaluations [1, 4].

6.1 Experimental Setup

A standard desktop computing environment was used for experiments, with a Python implementation of the proposed framework. Position evaluations were collected from a chess engine with fixed parameters (constant search depth) so that the results would be the same each time the program ran [1].

The complete chess games have been chosen from a publicly available chess data repository. The games come in varying lengths and characteristics of their positions to test the robustness of the analytical logic in different games. There was no special attention given to any particular openings, player ratings, or tactical themes as per previous centipawn analysis studies [4, 6].

These games were processed in turn, in a modular pipeline fashion as described in Section 3, yielding per move centipawn evaluations, differences between the evaluations in each game, as seen by the movers, qualitative categorizations, and explanatory annotations [22].

6.2 Case Study: Single Game Analysis

A typical game was studied in detail to demonstrate the behavior of the system. The framework calculated the centipawn evaluation of the position resulting from each ply, and the normalized evaluation change from the mover.

The evaluation curve for the game selected is displayed in figure 2, in the form of centipawns against move number. The curve transitions sharply downwards at points where the move is significant for the tactics and could be considered a material loss or missed opportunity, and these are then reflected in the move classification.

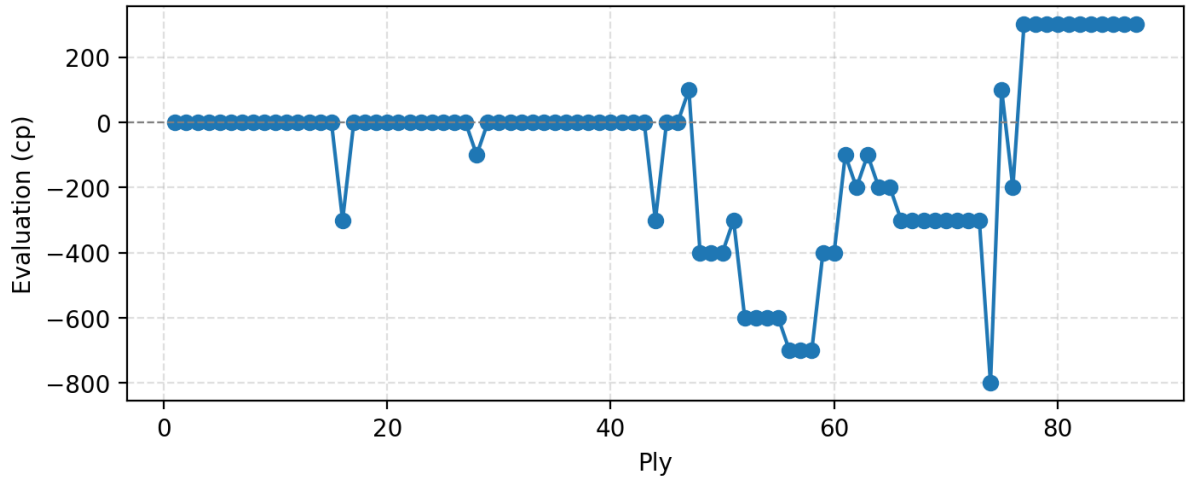


Fig. 2. Centipawn evaluation trend over move number for a representative game. Abrupt changes indicate tactically significant decisions.

A few moves which the system had categorized as blunders or mistakes were hand checked and they correlated with discernible material or positional loss, lending support to the system's classification and evaluation logic.

6.3 Move Classification Distribution

Table 1 gives a summary of the distribution of move classifications for the representative case study game. As might be expected, most moves are neutral or good moves and the severe errors are rare.

Table 1. Distribution of move classifications for the representative case study game (Kasparov vs. Topalov, Hoogovens Group A, 1999)

Move Classification	Count
Neutral	66
Good	21
Inaccuracy	0
Blunder	0
Mistake	0
Total moves analyzed	87

This distribution is similar to what has been observed in previous chess evaluation studies on empirical data, where sudden evaluation changes are not common and are usually followed by only minor changes or even stability [4,6]. Important, the lack of inflated error detection also means that the fixed centipawn thresholds aren't over-classifying moves as blunders.

6.4 Evaluation Change Analysis

To further assess system sensitivity, the absolute values of mover-relative evaluation changes, $|\Delta_i^{\wedge}(m)|$, were analyzed across all processed games. Figure 3 presents a histogram of these values.

The histogram shows a large number of small evaluation changes, which correspond to quiet or positional changes, and a long tail of errors or gains that are significant for tactics. This distribution is consistent with the centipawn loss distribution that is typically found in large-scale game analyses [4].

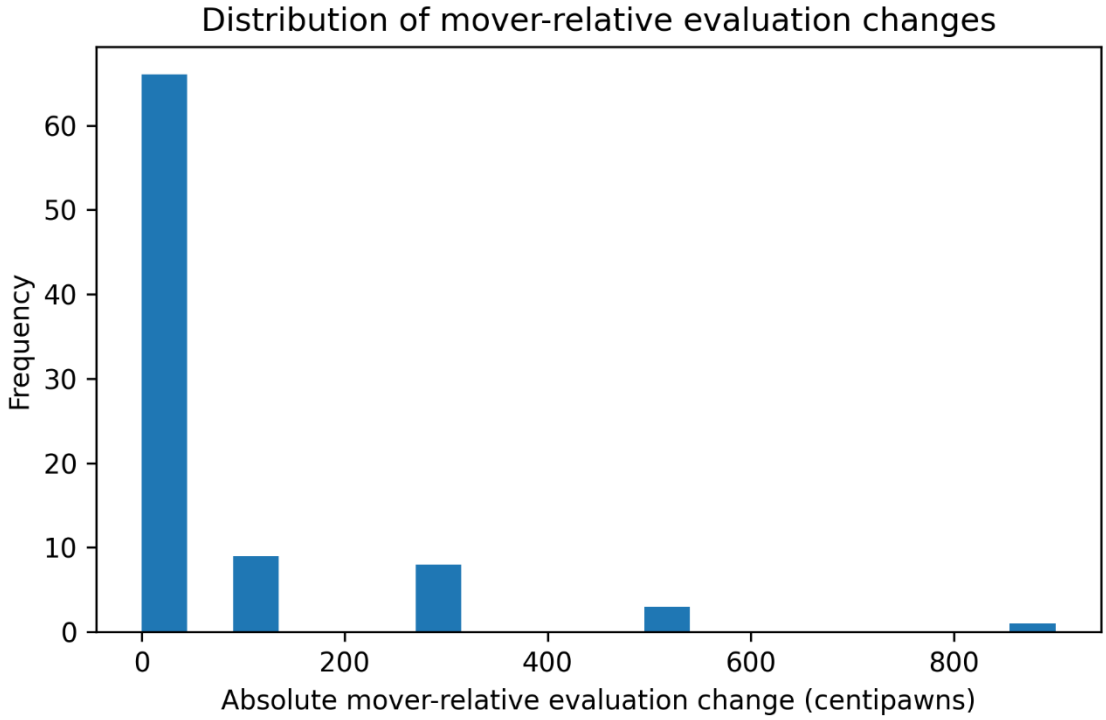


Fig. 3. Distribution of absolute mover-relative evaluation changes across analyzed games.

These results show that the evaluation difference metric is able to detect both minor changes in position and key game situations.

6.5 Explainability Assessment

Correspondence between generated explanations and observable board events was examined qualitatively to evaluate the explanations. All the moves involving material loss, missed captures, or exposure to check were followed by explanations that referred explicitly to loss of material, missed captures, or check exposure.

Unlike commercial analysis platforms that only produce short qualitative labels, but don't reveal the underlying logic [5,6] this framework explicitly maps changes in numerical evaluations to observable game properties. This explicit connection increases transparency and it helps with the learning-oriented use cases, as per the principles of explainable decision analysis [7].

6.6 Performance and Consistency

Average analysis time per game was used as an indication of computational performance. Results show that the framework can process complete games within interactive time limits and is appropriate for real-time or near-real-time analysis.

The same games were evaluated multiple times and the evaluation curves and classifications were found to be the same, as the system was determined to be deterministic. The proposed framework is reproducible and different results in the evaluation can be obtained in commercial frameworks as a result of not-disclosed changes in their configuration [5].

6.7 Discussion

The experimental results show that the proposed framework is successful and efficient in converting engine evaluations to understandable insights at a step level. In existing engines and platforms, based on their primary function, the strength of the engines and quality of recommendations is the main focus [1,2] but the proposed system adds a layer of post-game analysis which is transparent and has explicit logic.

This contribution should thus be seen as an architecture to interpret and not as a new algorithm for a chess-engine. The novelty, however, is that this is an explicit sequence of quantitative assessment, normalization to the movement of the engine and subsequent classification based on the rules, which then leads to a deterministic explanation and visualization of the existing engine evaluation. This organization provides a layer that can be easily checked, evaluated and reproduced outside of changing the underlying engine, and aids in making decisions.

The explainability capability of the proposed framework is dependent on the rule based design of the framework. The framework explains classifications transparently by having explicit links between changes in engine evaluations and observable properties on the board and between these properties and predefined interpretation rules. It does not give any information about the Stockfish engine logic, the search process, or representations in the Stockfish engine. The contribution is thus best seen as rule-based interpretability or model transparent decision support rather than model level

Explainable AI. This is also a new avenue for future research, where model-agnostic or model-specific approaches of XAI, which are not explored yet, can be used to complement the existing deterministic approaches.

6.8 Summary

The experimental results show that the proposed framework for explainable chess analysis yields consistent, interpretable and computation-efficient results over different games. The results confirm the efficiency of the system in terms of move evaluation and classification logic, and its applicability for analytical, educational and decision support applications.

7. Limitations and Future Work

The proposed explainable chess move analysis framework is consistent, interpretable and reproducible, however, there are some limitations.

The system is based on the evaluation of position as the only quantitative measure, and that is done in centipawns. Chess analysis is generally performed using Centipawns, which is a well-accepted and well-established system, however, not all of the strategic aspects of a position, including fortress potential, psychological pressure, and practical complexity, are covered. This means that some subtle strategic maneuvers might be neutral, but could still have a lasting impact.

Second, the move classification thresholds are fixed and rule-based. This design feature helps to increase transparency and reproducibility but does not account for game phase (opening, middlegame, endgame) or material imbalance. The current framework doesn't explicitly model the different practical consequences of the same centipawn loss in different positions, so this can be different.

Third, the explainability layer targets the game features that are visible and explained easily, like the amount of material changes, the amount of evaluation. Positional ideas, like the lack of certain squares, potential threats from multiple moves, or ideas that have to be developed over time are not explicitly indicated. This limitation is a deliberate one because the system is designed to be able to explain things deterministically, rather than inferring or learning them from experience.

Lastly, the experimental evaluation is conducted with the help of a post-game analysis of the completed games. The framework is not intended for real-time decision support during live play, nor is it intended to offer move suggestions and/or training feedback beyond interpretive explanations.

In the future, we would like to expand the framework in a number of directions. Adaptive classification thresholds that take into account game phase or material context may enhance interpretive accuracy without sacrificing transparency. Opaque machine learning models can be replaced with additional explainability rules capturing higher level concepts of position. The visualization layer may further be improved to provide comparative analysis between two or more games or players. The framework can be extended to other sequential rule-governed decision-making tasks where interpretability and reproducibility are crucial, such as games of Go, Poker, and similar.

DECLARATION OF COMPETING INTEREST

The authors do not have any competing financial interests or personal relationships that could have appeared to bias the work reported in this paper.

Funding

This work was not supported by a specific grant from a funding agency in the public, commercial or not-for-profit sectors. This manuscript was prepared with the aid of generative artificial intelligence (AI) tools for language editing and clarity enhancement. The author reviewed and edited all generated content and take full responsibility for the content of this publication.

Declaration of Generative AI and AI-Assisted Technologies in the Writing Process

This manuscript was prepared with the aid of generative artificial intelligence (AI) tools for language editing and clarity enhancement. The author reviewed and edited all generated content and take full responsibility for the content of this publication.

Highlights

- Explainable AI-based decision-support framework for chess move analysis.
- Reproducible engine-assisted evaluation with deterministic fallback mechanism.
- Transparent move-wise classification into blunder, mistake, and inaccuracy.
- Human-interpretable visual analytics for engineering-oriented decision support.

REFERENCES

1. “Explainability of Artificial Intelligence Systems: A Survey,” in Proc. 2023 Int. Symp. Networks, Computers and Communications (ISNCC), 2023, doi: 10.1109/ISNCC58260.2023.10323642.
2. “Explainable AI 2.0: Conceptual Shifts and New Requirements,” in Proc. XXVIII Int. Conf. Soft Computing and Measurements (SCM), 2025, doi: 10.1109/SCM66446.2025.11060242.
3. “Explainable Artificial Intelligence (XAI): Concepts, Applications, Challenges, and Future Perspectives,” IEEE Access, vol. 14, pp. 27394–27417, 2026, doi: 10.1109/ACCESS.2026.3663161.
4. “Transparency in AI: A Review of Explainable Artificial Intelligence Techniques,” in Proc. 8th Int. Conf. Computing, Communication, Control and Automation (ICCUBEA), 2024, doi: 10.1109/ICCUBEA61740.2024.10774981.
5. F. Lowry et al., “Reproducibility in computational science,” Science, vol. 359, no. 6377, pp. 725–726, 2018.
6. E. Shannon, “Programming a computer for playing chess,” Philosophical Magazine, vol. 41, no. 314, pp. 256–275, 1950.

7. Chess.com, “Game Analysis,” https://www.chess.com/analysis, accessed 2025.
8. D. Gunning et al., “XAI—Explainable artificial intelligence,” Defense Advanced Research Projects Agency (DARPA), 2019.
9. G. Campitelli and F. Gobet, “Predicting chess skill from move quality,” Psychonomic Bulletin & Review, vol. 11, no. 3, pp. 484–490, 2004.
10. Ahmed, G. Jeon, and F. Piccialli, “From Artificial Intelligence to Explainable Artificial Intelligence in Industry 4.0: A Survey on What, How, and Where,” IEEE Trans. Ind. Informat., vol. 18, no. 8, pp. 5031–5042, 2022, doi: 10.1109/TII.2022.3146552.
11. Guid and I. Bratko, “Computer analysis of world chess champions,” ICGA Journal, vol. 29, no. 2, pp. 65–73, 2006.
12. J. Thomas and K. A. Cook, Illuminating the Path: The Research and Development Agenda for Visual Analytics, IEEE Computer Society, 2005.
13. Schaeffer, R. Lake, P. Lu, and M. Bryant, “Chinook: The world man–machine checkers champion,” AI Magazine, vol. 17, no. 1, pp. 21–29, 1996.
14. Sigman and J. A. Rockey, “Estimating chess skill from game outcomes and move evaluation,” PLoS ONE, vol. 15, no. 7, e0235475, 2020.
15. Lichess.org, “Computer Analysis,” https://lichess.org/analysis, accessed 2025.
16. Arunika et al., “A Survey on Explainable AI Using Machine Learning Algorithms Shap and Lime,” in Proc. 15th Int. Conf. Computing Communication and Networking Technologies (ICCCNT), 2024, doi: 10.1109/ICCCNT61001.2024.10725120.
17. T. Ribeiro, S. Singh, and C. Guestrin, “Why should I trust you? Explaining the predictions of any classifier,” in Proc. ACM SIGKDD, pp. 1135–1144, 2016.
18. S. Brdnik and B. Šumak, “Current Trends, Challenges and Techniques in XAI Field; A Tertiary Study of XAI Research,” in Proc. 47th MIPRO ICT and Electronics Convention (MIPRO), 2024, pp. 2032–2038, doi: 10.1109/MIPRO60963.2024.10569528.
19. Stockfish Development Team, Stockfish Chess Engine, https://stockfishchess.org, accessed 2025.
20. T. A. Bach, J. K. Kristiansen, A. Babic, and A. Jacovi, “Unpacking Human-AI Interaction in Safety-Critical Industries: A Systematic Literature Review,” IEEE Access, vol. 12, pp. 106385–106414, 2024, doi: 10.1109/ACCESS.2024.3437190.
21. Y. Wang and G. Tuerhong, “A Survey of Interpretable Machine Learning Methods,” in Proc. 2022 Int. Conf. Virtual Reality, Human-Computer Interaction and Artificial Intelligence (VRHCIAI), 2022, doi: 10.1109/VRHCIAI57205.2022.00047.
22. Z. Alexander, D. H. Chau, and C. Saldaña, “An Interrogative Survey of Explainable AI in Manufacturing,” IEEE Trans. Ind. Informat., vol. 20, no. 5, pp. 7069–7081, 2024, doi: 10.1109/TII.2024.3361489.