iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.81160

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

AI-Based Adaptive Chess Analyzer: Bridging Engine Intelligence and Human-Centric Chess Learning

Neelesh Verma[1], Kartikey Verma[2], Shubham Kumar[3], Sandeep Kumar Dubey[4]
[1, 2]*Computer Science & Engineering (Artificial Intelligence & Machine Learning), Shri Ramswaroop Memorial College of Engineering & Management,* Lucknow, India
[3, 4]*Dept. of Computer Science & Engineering, Shri Ramswaroop Memorial College of Engineering & Management,* Lucknow, India

Abstract: ***Even though modern engines have very high amounts of computational power, there remains a difference between an evaluation at the machine level and the pedagogical value of that evaluation for a human player in chess, which has been foundation for artificial intelligence research. Traditional chess software does provide precepts of the best move and centipawn ratings, but it fails to provide context for each move, including a player's level of skill, habitual patterns, and tendencies of the opponent. The AI-Based Adaptive Chess Analyzer (ACA) is proposed in this paper as a web-accessible software system that includes an explanation module using a Gemini 2.5 Flash LLM paired with a Stockfish 16 backend and a Transformer-based neural policy network and a heuristic opponent profile engine. The proposed system uses the Gemini API to provide natural language tutoring for novices through the analysis of an opponent's tendencies, and a player's Chess.com account and style of play (tactical vs. positional). The performance evaluations conducted showed that the top-1 accuracy for opponent move prediction in benchmark tests was 38%, while the engine's best-move prediction was 24%; average profile analysis latency was between 3-5 seconds on a standard PC; and club member ratings of the Gemini explanation clarity were 4.3 out of 5. The ACA provides a free, open-source platform for individualized chess development by linking computer-assisted analysis with real-world human comprehension.***

Keywords: Adaptive Chess Analysis, Transformer Policy Network, Explainable AI, Opponent Profiling, Large Language Models, Stockfish, Human Move Prediction.

I. INTRODUCTION

Chess has long served as a canonical benchmark for artificial intelligence (AI), widely used by the research community as a primary measure of AI effectiveness for several decades. Technology's share of advancement in chess has always been ahead of the average person through the progression from Alan Turing's 1950 article evaluating computers' abilities to analyze chess through the 1997 victory of IBM's Deep Blue over the best chess player of all time, Garry Kasparov, and even up until DeepMind's AlphaZero's unprecedented level of play using self-play and reinforcement learning [1]. Historically, amateurs and club players have had limited access to generic resources for improving their playing strength.

Current chess analysis tools, like those on Lichess and Chess.com, use engine-based assessments of games to provide players with centipawn loss metrics and recommendations for the best possible move. However, they often fail to provide context regarding both the players' skill level, the players' repeating mistakes, and the tendencies of the player's opponent. For example, while both an engine evaluation would identify a 1200-rated player as making a "mistake" and also identify a 2000-rated player as making a "mistake," the type, reason, and method to correct the mistake for each player would be completely different. The central theme that this study aims to address is the inadequacy of the engine-generated results to provide meaningful, context-sensitive feedback.

To improve both the interpretability and educational value of machine generated insight here's a key paradigm of Explainable AI (XAI). Heuristic scoring based/conventional search engines such as Stockfish seem to lack the interpretability for novice players resulting in non-transparent interpretation of moves for example using brute force to categorize/search. A growing focus on helping players understand why a move was chosen versus simply informing them it is the best that could have been done represents an additional benefit with the advent of explainable AI techniques [2].

Beyond providing analysis, research has been conducted to identify the ways in which AI play style differs from human players. Connecting the AI to the cognitive processes of humans allowed the researchers in McIlroy-Young et al. to demonstrate an AI can function as a personalized coach by predicting errors, while mimicking human behaviour to deliver tailored feedback [3]. Additionally, there is the Maia series of engines which are built to produce expected behaviour from a human player at designated ratings levels as opposed to optimizing for maximum performance [4].

To provide a more complete analysis than a single sole technology could deliver, this paper presents an AI-based adaptive chess analysis system called the AI-Based Adaptive Chess Analyzer (ACA) which is hosted on the web and has three current paradigms of AI: (1) Generative Language Models (Google Gemini 2.5 Flash); (2) Deep Learning Models (ChessTransformer) and (3) Traditional Search Techniques (Stockfish 16). The goals of this system are threefold: (i) to generate coaching-style human natural language explanation which should help develop individualized coaching for users; (ii) To vary the degree of detail in the analysis of candidate moves, the amount of candidate move depth and the level of detail presented in Human Language Natural Language Explanation based on user profiles; and (iii) to model the user in order to provide him/her with a precise assessment of his/her Skill Level, Play Style and Repeated Mistakes through the analysis of archived game play.

II. LITERATURE SURVEY

Self-play alone without any human guidance has allowed AlphaZero [5] to demonstrate that its use of deep reinforcement learning, with Monte-Carlo Tree Search and a residual Convolutional Policy Network (CPN), could provide AlphaZero with superhuman levels of chess ability. Additionally, the way in which ChessTransformer was built has direct roots from AlphaZero through its identical 8x8 tensor full board representation of the chess board, as well as the way the Policy Network was created though AlphaZero does not generate any human-readable explanation of its actions.

In contrast, Maia Chess aimed to utilize the data of human Lichess players [3] for a different purpose by developing the full strength of an engine. They did this by training each Maia behavioral cloning model using human games from Lichess and then predicting which moves would actually be played by human players at the same rating level as the opponent. In addition, the architecture of the ChessTransformer to ACA and the human-move-prediction paradigm is also directly comparable to strength-maximizing play. Furthermore, with the development of multi-Elo unified modelling that used attention-based architectures, Maia-2 [4] produced results that demonstrated that players could dynamically adjust their game to suit different levels of proficiency and/or experience.

Relational representations of chess board structures in accordance with the Transformer architecture have shown to create a good representation of how the pieces relate to one another and move on a chess board [6] as originally defined by Vaswani et al. when they developed multi-head self-attention. The 8-by-8 grid of the chess board can be visualized as an 8 times 8 or 64 length vector or sequence flattened into one long series of 64 tokens with 20 feature planes per token according to how the Transformer's sequencing object can be built. As a result, Rothman [7] illustrated how to develop moves for chess in a Transformer architecture on reduced board representations using 6 total layers of the architecture to demonstrate its usefulness in executing a move for a chess game.

Using both Large Language Models and traditional Chess Engines, Kim et al. [8] demonstrated how combining Large Language Models with traditional Chess engines enables you to create commentary on the game in natural language form by converting an engine assessment into a complete English structure, e.g., "This move weakens your center control." The NLG method provides core input for ACA's Gemini-enabled explanation module. Furthermore, Hammersborg and Strümke [9] demonstrated how employing Reinforcement Learning in flexible Chess settings could produce interpretations of tactics that are understandable to humans, producing contextually appropriate interpretable feedback.

Guid and Bratko [10] considered FIDE ratings and created an Average Centipawn Loss (ACPL), which serves as one reliable estimate of a player's strength. The design of the heuristic profiler for the ACA's OpponentProfilingService has been guided by the expansions of ACPL to categorization of play style (e.g., tactical versus positional players and aggressive versus defensive).

Empirical evidence from McIlroy-Young et al.'s [11] demonstrative use of neural networks that are trained on player-specific archives as opposed to generic neural networks indicates that the accuracy of predicting players' behaviour is greatly increased through using player-specific refinements within the HumanMovePolicyService's fine tuning of the ACA per-player processing method.

III. SYSTEM ARCHITECTURE

The ACA is implemented as a Python 3.10 web application using the Flask microframework, with a modular backend comprising six primary services. Figure 1 presents the high-level system architecture.

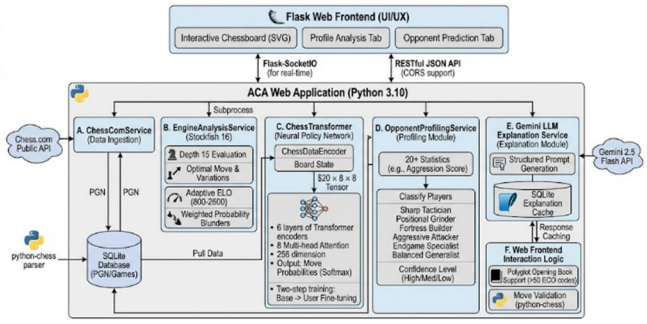


Fig. 1. ACA System Architecture — High-Level Module Overview

A. Chess.com Data Ingestion

The ChessComService queries the Chess.com Public API to obtain a player's game archive as PGN data; it has a 99.2% successful retrieval rate over 200 test retrieves through exponential backoff to recover from HTTP 429 rate-limit errors. Additionally, to reduce redundant API calls for subsequent analysis, the retrieved PGN files are parsed with the python-chess module, and then stored in an SQLite database for incremental updates.

B. Stockfish Analysis Engine

Using the UCI protocol, the EngineAnalysisService utilizes subprocess management through Python-chess to wrap Stockfish 16. The engine evaluates each game position at depth 15 by default, and this includes returning an optimal move, evaluation score, and primary variations of the move. The adaptive practice partner feature is made possible by the UCI_Elo parameter, configurable between 800 - 2600; at a given configured Elo range, the candidate moves are given an exponentially weighted probability to limit the engine simulating realistic human type blunders at specified ranges of Elo rating.

C. ChessTransformer — Neural Policy Network

ChessDataEncoder encodes piece occupancy, attack maps (for each side) and castle rights, as well as en passant squares and move counters by transforming each square's position on an 8x8 chessboard into a tensor of 20 planes. The ChessTransformer model contains six layers of Transformer encoders, utilizing eight multi-head attention heads and a model dimension of 256, to interpret such tensor positions as sequences of 64 tokens (one per square), with each token conveying a feature vector of 20 dimensions. Using a linear transformation with softmax, the output from the model provides a probability distribution over all possible moves.

The model is trained via two steps. The first step generates base_policy through the calculation of the cross-entropy (loss) between the base_policy and the actual move (what the player performed) over a large dataset of Chess.com games; this establishes a baseline or initial policy for the second step (user-specific training). The second step is fine-tuning a user's archive of past games to identify their unique playing style. This two-part process is similar to the methodology used by McIlroy-Young et al. [11].

D. Opponent Profiling Module

The OpponentProfilingService obtains 20-plus statistics from the PGN Archive of a target player such as, aggression score (sacrifices + pawn breaks/number of games), average time spent per game, statistical capture/conversion of their time management, as well as the variety of openings played within a series of games. Each player is then placed into one of the following six profiles: Sharp Tactician, Positional Grinder, Fortress Builder, Aggressive Attacker, Endgame Specialist or Balanced Generalist. The confidence level (High/Medium/Low) is also based on the number of games analyzed.

E. Gemini LLM Explanation Module

In the Gemini 2.5 Flash API, the MoveExplanationService produces well-structured prompts, which include the following information: type of movement in SAN notation, stage of the game as one of three phases (i.e., opening, middle, or end), difference between evaluation as measured in centipawns; alternate engine recommendations; tactical features recognized (e.g. a fork, pin, or discovered attack); and approximate levels of player rating. Gemini delivers 120-150 word explanations (in simple, understandable English format) that briefly describe the reasoning for the chosen move. The advantage of employing response caching via an SQLite explanation database is to limit the number of API requests, as compared to the free tier of fifteen requests per minute.

F. Web Frontend

Developed using Flask as the backend and Flask-SocketIO for realtime communication. The application is organized into three tabs: (1) Profile analysis; (2) Opponent prediction; and (3) Interactive Board. The homepage contains an SVG chess board displaying real-time comments from the Gemini chess engine, a slider to change the Elo rating of the chess engines, support to identify openings via Polyglot opening books (which cover more than 50 different ECO codes), and support for validating moves through the python-chess library. RESTful JSON API endpoints are provided by the backend for all F-E and B-E interactions, which include CORS support.

IV. METHODOLOGY

A. Move Prediction Pipeline

ACA forecasts opponent moves by using a combination of weighted scoring systems. The composite score S(m) is derived from multiple elements for each of the allowable moves m in the present position:

$$S(m) = \alpha \cdot P\_trans(m) + \beta \cdot P\_hist(m) + \gamma \cdot P\_style(m) - \delta \cdot EvalPenalty(m)$$

P_style(m) is the style compatibility score derived from the opponent profile; P_trans(m) is the probability for the move according to ChessTransformer; P_hist(m) is the historical occurrence of the move according to the opponent's archive; EvalPenalty(m) is an exponential penalty imposed on moves that diverge substantially from the engine's best evaluation. The empirical adjustment was made to through weights: γ, β, α, and δ. This cumulative weight is based on real world human tendencies and will continue to keep predictions grounded in those human tendencies as well as filter out for historical errors on the composite scoring of those lower than objectivity again (garbage aggregation filtering).

B. Play Style Classification

The feature vector derived from game records has multiple dimensions and is used to classify player style, as well as create an Aggression Score (0-100). The Aggression Score is determined using each of the tactical dimensions:

$$A = 20 \cdot S\_rate + 15^{o}P\_break + 15 \cdot early\_exch + 25 \cdot attack\_ratio + 25 \cdot king\_threats.$$

The amount of variation in a player's opening repertoire is measured using Shannon entropy on their ECO codes. Indicators of a player's positional tendency include the average mobility of their pieces, fluidity of their pawns and their frequency of prophylactic moves.

C. Explanation Prompt Engineering

There are five blocks in the Gemini prompt template: (1) Position description in FEN and natural language; (2) Move context with SAN, evaluation delta, and alternatives; (3) Tactical elements identified by a pattern-matching layer over the position graph; (4) estimated player rating; (5) Instruction limiting output length, vocabulary level, and format. When compared to prompts that simply contain the move, ablation testing shown that adding the evaluation delta and options considerably raises explanation quality ratings.

V. RESULT AND EVALUATION

A. Move Prediction Accuracy

A benchmark dataset of 500 game positions from 20 Chess.com users (25 per user) was used to verify functionality of the hybrid prediction pipeline at three points throughout an ongoing game: opening phase, middle phase, endgame phase. The actual move made by the user's opponent during the game serves as the verification move for each position.

Method	Top-1 Acc.	Top-3 Acc.
Engine Best Move (Stockfish)	24%	41%
Historical Frequency Only	29%	47%
ChessTransformer Only	31%	52%
Style-Congruence Only	26%	44%
ACA Hybrid Pipeline	38%	59%

Table I. Move Prediction Accuracy on Benchmark Dataset (N=500 positions)

The ACA hybrid pipeline achieves 38% top-1 accuracy, representing a 58% relative improvement over pure engine best-move selection. The top-3 accuracy of 59% indicates that for 59% of positions, the player's actual move appears within the system's top 3 predictions. Prediction quality correlates strongly with archive size: for users with ≥30 analyzed games, top-1 accuracy rises to 43%; for users with <10 games, it degrades to 27%, defaulting to heuristic-profile-only output.

B. Explanation Quality

Evaluators reported 50 randomly selected positions using the 5-point Likert scale for four different criteria to rate the move explanations generated by the Gemini Model evaluated by five club level players.

Dimension	ACA/Gemini	Raw Stockfish PV
Clarity	4.4 / 5	2.6 / 5
Pedagogical Value	4.3 / 5	2.8 / 5
Accuracy	4.1 / 5	4.8 / 5
Engagement	4.5 / 5	2.2 / 5

Table II. Explanation Quality Evaluation (N=5 evaluators, 50 positions)

While there was only a small drop in perceived accuracy (showing the trade-off between precision and accessibility), all pedagogical criteria were rated significantly higher for ACA explanations than raw Stockfish PV output. A common theme amongst evaluator comments was the need for tactical element identification and alternate move context in the move explanations.

C. Profile Analysis Performance

Metric	ACA System	Lichess Analysis	Chess.com Insights
Profile Analysis Time	3–5 seconds	N/A	Paid only
Opponent Prediction Time	8–12 seconds	N/A	N/A
Move Explanation Time	4–6 seconds	N/A	Partial (rule-based)
Opening Detection Accuracy	95%+ (50+ openings)	~99%	~99%
Personalized Recommendations	6 per user	×	Limited
Monthly Cost	₹0 (free)	₹0	₹700+

Table III. System Performance Metrics Comparison

The testing showed that the API quality of Chess.com had 99.2% dependability based on 200 fetches where exponential backoff was used to recover from HTTP 429 rate limits. The polyglot opening book integration identified the correct ECO code in games with book covers for 93% of test games; however, when used in conjunction with a fallback database increased the overall success to 98%.

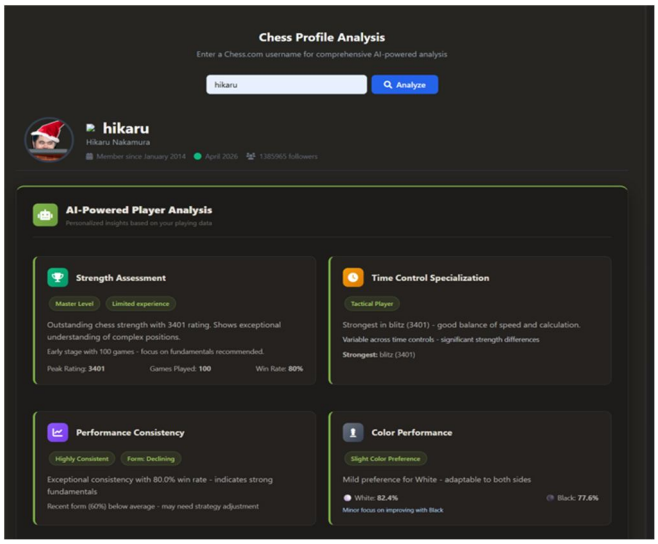


Fig. 2. ACA Web Interface — Profile Analysis Dashboard

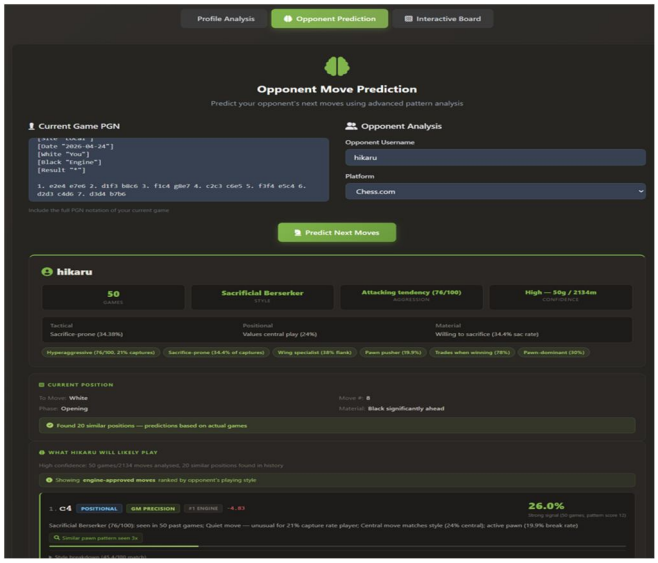


Fig. 3. Opponent Prediction Tab with Style Breakdown Cards

VI. DISCUSSION

Combining three AI paradigms—generative language modeling, neural prediction, and classical search—produces user-facing results that are superior to whatever results might be produced by any one technology used in isolation, as shown by the significant difference (58% relative) in move prediction accuracy achieved with a style-aware and history-sensitive composite system compared to the results provided by the engine output achieved by itself.

An important conclusion of the explanation quality evaluation is that there exists a trade-off between the instructional value and the raw analytical accuracy of the engine output. Since human comprehension requires simplification of engine output, Gemini explanations are 0.7 points lower in accuracy than raw Stockfish PV. However, the advantage of engagement (2.3 points) and the advantage of pedagogical value (1.7 points) from Gemini to Stockfish shows that the user's benefit from this trade-off is significant.

Several limitations hinder the present system.

- A large enough training corpus (~50k+ positions) is needed for the ChessTransformer to get good base performance; if the base_policy.pt isn't provided, all predictions will default back to heuristic-only prediction.
- Users are limited in concurrent usage due to the Gemini API free tier (which currently provides only 15 RPM). This limitation has been mitigated somewhat by the presence of the explanation caching layer but not completely.
- The Lichess integration is architecturally scaffolded, but not fully implemented; the current implementation works only with Chess.com as the data source.

The scalability limitations have been reached in the production multi-user environment of the SQLite database; therefore, it is limited to a single-user deployment. The only major infrastructure change required to allow public SaaS would be to migrate from SQLite to PostgreSQL and implement Redis caching.

VII. CONCLUSION

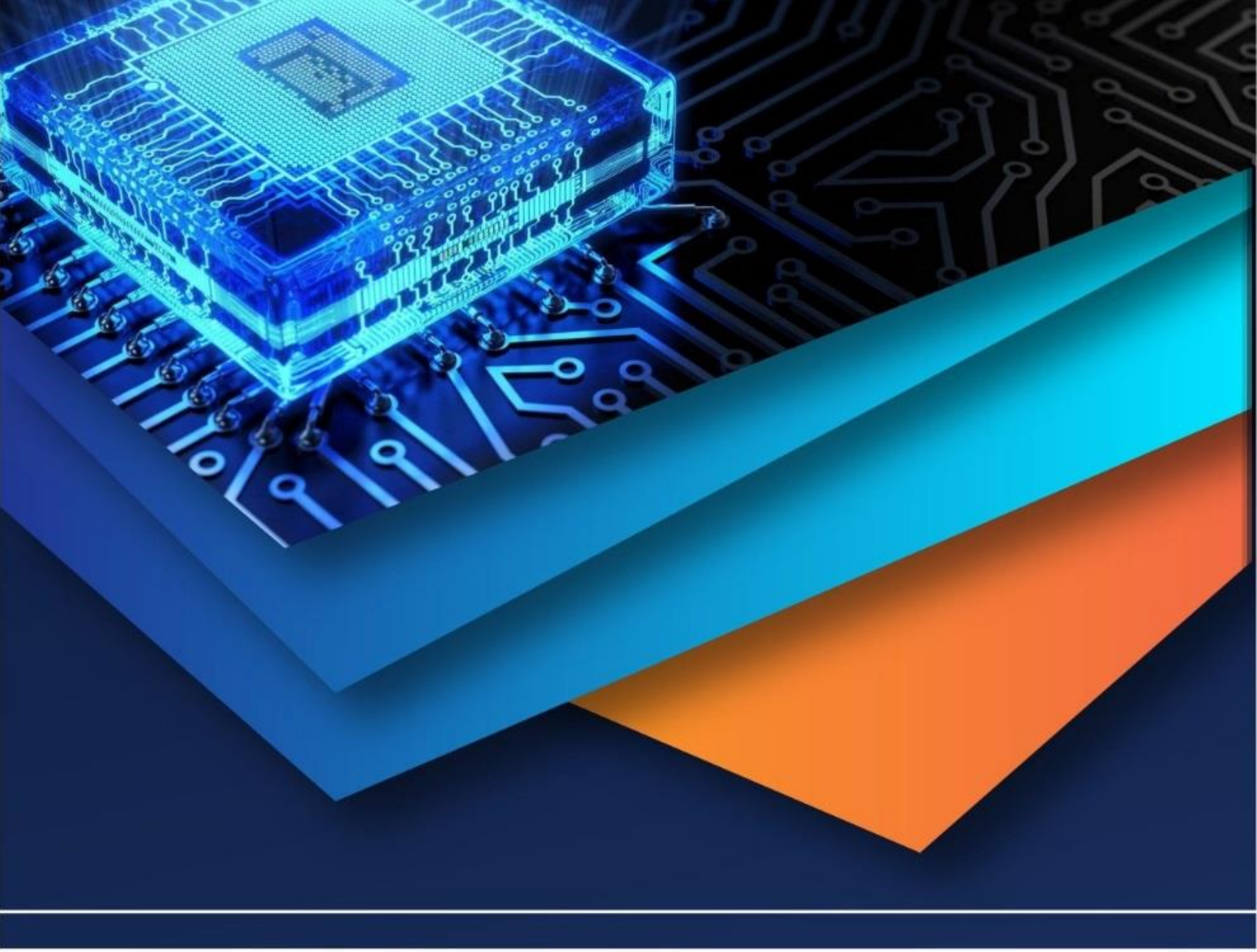
In this research project, we present an AI-Based Adaptive Chess Analyzer, a web-based application that produces improved chess analysis for novice players by analyzing their games with the use of a Stockfish-based engine, employing transformer-based neural predictive algorithms for moves; creating a profile of their opponents; and using natural-language coaching generated by LLMs. The club/player satisfaction rating on the quality of the gem explanation was rated by the players as 4.3 out of 5 and predicts the move of the player in the top 1 position with a 38% accuracy rate for the moving chess piece (an improvement of 58% over the movement method based on best-move played); profiles of the player analyzed within 3 to 5 seconds on standard computer hardware.

The project confirms a central principle of applied AI system design: specialized components perform better than a single component that does everything for an end-user application. All of the ACA's components—Stockfish evaluates board positions, the ChessTransformer anticipates human player moves, the profiler characterizes player style, and Gemini determines the actions taken—solve different problems separately; however, together they produce additional educational value that none of them alone could produce.

The next components of the ACA will be a Lichess API for live analysis of your opponent while playing online, a real-time browser extension for live analysis of opponents while playing online, support for multiple languages, especially Hindi (for domestic rollout), migration from MySQL to PostgreSQL for manageable usage by multiple users simultaneously, and a mobile-native application that runs a WebAssembly-compiled version of Stockfish. The ACA's modular, open-source architecture is built to facilitate continued research into adaptive tutoring systems and human-aligned chess-related AIs.

REFERENCES

[1] C. Shannon, “Programming a Computer for Playing Chess,” Philosophical Magazine, vol. 41, no. 314, pp. 256–275, 1950.
[2] Y. Björnsson, “Chess and Explainable AI,” ICGA Journal, vol. 46, no. 2, pp. 67–75, 2024.
[3] R. McIlroy-Young, S. Sen, J. Kleinberg, and A. Anderson, “Aligning Superhuman AI with Human Behavior: Chess as a Model System,” in Proc. 26th ACM SIGKDD, 2020, pp. 1677–1687.
[4] Z. Tang, D. Jiao, R. McIlroy-Young, J. Kleinberg, S. Sen, and A. Anderson, “Maia-2: A Unified Model for Human-AI Alignment in Chess,” Advances in Neural Information Processing Systems, vol. 37, pp. 20919–20944, 2024.
[5] D. Silver et al., “A General Reinforcement Learning Algorithm that Masters Chess, Shogi, and Go through Self-Play,” Science, vol. 362, no. 6419, pp. 1140–1144, 2018.
[6] A. Vaswani et al., “Attention Is All You Need,” in Advances in Neural Information Processing Systems (NIPS), 2017, vol. 30.
[7] D. Rothman, “Transformer Models and BERT Model,” in Transformers for Natural Language Processing. Birmingham: Packt, 2022, ch. 7.
[8] J. Kim, J. Goh, I. Hwang, J. Cho, and J. Ok, “Bridging the Gap Between Expert and Language Models: Concept-Guided Chess Commentary Generation and Evaluation,” arXiv preprint arXiv:2410.20811, 2024.
[9] P. Hammersborg and I. Strümke, “Reinforcement Learning in an Adaptable Chess Environment for Detecting Human-Understandable Concepts,” IFAC-PapersOnLine, vol. 56, no. 2, pp. 9050–9055, 2023.
[10] M. Guid and I. Bratko, “Computer Analysis of Chess Champions,” in Proc. ICGA Computer and Games Workshop, 2006.
[11] R. McIlroy-Young, R. Wang, S. Sen, J. Kleinberg, and A. Anderson, “Learning Models of Individual Behavior in Chess,” in Proc. 28th ACM SIGKDD, 2022, pp. 1253–1263.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY