# Rylee: Creating a Deployable Human-Like Chess Engine to Enhance the Learning Experience

1st Ethan Gee
*Computer Science*
*Utah State University*
Logan, UT, USA
workethangee@gmail.com

2nd Nate Stott
*Computer Science*
*Utah State University*
Logan, UT, USA
nate.stott@pm.me

***Abstract*—Artificial Intelligence (AI) has surpassed human performance in many domains; since 2005, chess engines have demonstrated consistent superhuman play. However, fields such as law, medicine, and human resources cannot directly deploy superhuman AI due to ethical constraints and the underlying differences in human evaluation. These limitations do not preclude the use of AI systems designed to complement rather than exceed human performance. Prior work shows that aligning AI with human-observable behavior, rather than focusing solely on explaining its decisions, can yield desirable results. Chess provides a strong test-bed for developing such systems. We introduce Rylee, a neural-network-based chess bot trained to mimic human board evaluation, aiming for a lightweight and reasonably accurate model. Traditional chess engines, designed for optimal play, offer limited value as training tools; attempts to limit them to capture human-like behavior resulting in confusing player experience. Maia improves on the training experience by replicating human play. However, Maia requires substantial machine resources to train and run. Our work builds on the Maia project by developing a more computationally efficient, human-aligned model, similar in size to Stockfish, capable of running on limited hardware. Trained on the Lichess dataset across multiple rating levels, Rylee learns to predict human moves with modest success. Key results show that Rylee achieves 25% Top-1 accuracy and 51% Top-5 accuracy in predicting human moves using a model 30 times smaller than Maia (800,000 parameters vs. 25 million). Experiments demonstrate that the model generalizes well across training and validation sets with minimal overfitting and can be trained on edge devices in 2-3 days, compared to Maia's 3-4 weeks on high-end GPUs. Furthermore, Rylee expands capabilities beyond Maia by including opening move predictions (first 10 moves), supporting a broader ELO range (700-2500 vs. 1100-1900), unifying into one combined model, and incorporating diverse game types rather than filtering to only blitz games. The broader significance of this research lies in advancing human-aligned AI and enabling more accessible, engaging AI-based training systems across diverse domains.***

***Index Terms*—Chess, Neural Networks, Machine Learning, Artificial Intelligence, AI, Human-like AI, Deep Learning**

## I. Introduction

Since the 1940s, chess-playing programs, or "chessbots," have evolved to surpass the capabilities of the most skilled human players. These bots are now integral to modern chess training, with players globally using them to refine their skills. However, a significant disparity exists between the strategies employed by traditional chessbots and human players. Human chess is characterized by the use of heuristics, intuition, and pattern recognition. In contrast, conventional chessbots rely on brute-force computation, analyzing vast numbers of potential future board states to select the move that maximizes their probability of winning.

This computational superiority creates an imbalance in human-computer matches, where the bot's expansive memory and processing power provide a decisive advantage. To mitigate this, bots are often programmed to introduce deliberate errors, resulting in an unnatural and often confusing experience for the human player. This presents a paradox: while chessbots dominate the game, their non-human-like mode of play limits their effectiveness as a training tool for humans.

This project addresses the challenge of creating a more "human-like" chessbot. Our work builds upon the research of McIlroy-Young et al. on the Maia models, which were trained using supervised learning to emulate human decision-making, a departure from the reinforcement learning approach common to engines like AlphaZero. While Maia demonstrated strong performance in predicting human moves, its reliance on the large and resource-intensive AlphaZero architecture limits its practical application. Conversely, Stockfish, a significantly smaller engine, achieves comparable performance to AlphaZero, indicating that model size does not scale linearly with performance.

The primary objective of this project is to develop a chessbot that not only plays like a human but is also computationally efficient enough to run on standard consumer hardware. We aim to achieve a level of human-like play comparable to the Maia models but with a model deployable to edge devices. To this end, we will utilize the Lichess Dataset, a comprehensive collection of human chess games and player ratings, which is the same dataset employed in the training of Maia. This work contributes to the growing field of human-aligned AI by exploring the trade-offs between model size, performance, and human-like behavior in the domain of chess.

## II. Related Work

### *A. Chess Engines*

Chess has been a benchmark of the computing world for over a century. Early chess engines, such as Alan Turing's TuroChamp, relied on traditional search algorithms. Recently, the chess world has been dominated by deep learning based models such as DeepMind's AlphaZero. Instead of evaluating the strength of board states through human defined rule systems, they learn through rigorous self-play, leading to superhuman performance.

### B. Human-Like Chess AI

This superhuman level of play results in models that approach games in very different ways than human players. This, in addition to models being artificially weakened by reducing their search space and purposely choosing non-optimal moves, leads to confusing experiences when paired with human players. The Maia Project, by McIlroy-Young et al., represents a significant step in creating human-like chess engines. Maia is designed to be aligned with human play. This is done by transfer learning on a large dataset of human games, with the goal of predicting the move a human player would make in a given position. This results in a series of models that can effectively mimic human play at various skill levels.

### C. Dataset and Training Data

The Lichess database is a publicly available collection of human played games from the Lichess.org platform, containing billions of games spanning a wide range of player skill levels from beginner to expert. Alongside chess moves, Lichess also provides metadata for each game, such as player Elo ratings (rated skill level), outcomes, and time controls (the time limit for each game).

The Maia Project utilized 169 million Lichess games filtered to specific rating bands (1100-1900 ELO) and game types (blitz time controls only), training separate models for each 100-point rating interval. This approach enabled rating-specific modeling but required substantial computational resources for data processing and model training.

## III. Proposed Method

### A. System Architecture

We created Rylee, a chess engine, which can process Lichess data, train with tunable parameters, and deploy into a playable environment. First, automating Lichess game file downloads, then processing to a consumption ready format stored in the database. Second, model training draws directly from this preprocessed dataset via a custom PyTorch Dataset class. Lastly, models can be manually evaluated using a custom chess game implementation; a human, Stockfish, or Leela player can be used as the opposing player.

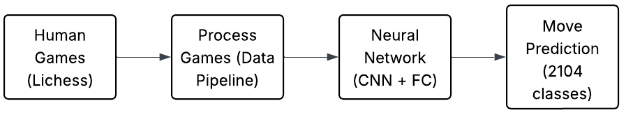
Human Games (Lichess)
Process Games (Data Pipeline)
Neural Network (CNN + FC)
Move Prediction (2104 classes)

Fig. 1. A High level system architecture diagram showcasing the full data flow from Lichess and processing to training and move prediction.

### B. Data Collection and Processing

*1) PGN File Processing:* Lichess data is stored in Portable Game Notation (PGN) files. PGN is made up of two parts the tag pairs, which stores game metadata (e.g., game duration, white player elo, winner), and the move sequence (e.g., e2e4, e1g1) stored in algebraic chess notation.

For each game, useful tags are extracted including the players elos. Next, we go through the moves section creating "snapshots" of the board at every step; we combine the snapshots with the move the player made on their turn and the Elo of both players (we call this metadata). Finally, each board snapshot–metadata pair is represented using three-dimensional tensors that are 8-by-8 (due to the size of the board) with 12 channels: six channels for the white pieces and six for the black pieces.

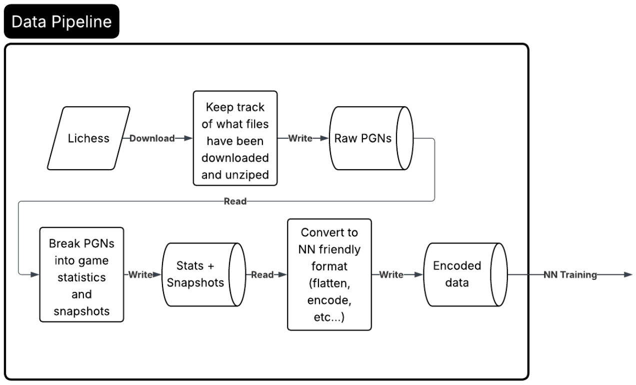
Data Pipeline
Lichess
Download
Keep track of what files have been downloaded and unziped
Write
Raw PGNs
Read
Break PGNs into game statistics and snapshots
Write
Stats + Snapshots
Read
Convert to NN friendly format (flatten, encode, etc...)
Write
Encoded data
NN Training

Fig. 2. A detailed data processing pipeline diagram showcasing the process starting from Lichess PGN files to neural network training data. Steps include file parsing, snapshot and metadata extraction, and encoding.

### C. Neural Network Architecture

The neural network architecture consists of three main sections: convolutional layers for processing board piece positions (pieces are spatially related), a fully connected network for feature learning and dimensionality reduction, and two output heads for human move and auxiliary predictions (predicting the legal moves available to the player).

The convolutional section processes the 8×8×12 encoded board through six convolutional layers. Each layer uses 64 filters with 8×8 kernels utilizing "same" padding. After each convolution, ReLU activation functions are applied to ensure non-linearity.

The fully connected section starts with a 4100-dimensional input (flattened board features + metadata) and progressively reduces dimensionality through six layers ($512 \rightarrow 32 \rightarrow 32 \rightarrow 32 \rightarrow 32 \rightarrow 32$) with ReLU activations. This creates a 32-dimensional embedding of the board state and metadata.

This architecture balances efficiency and complexity to capture human chess playing patterns. The dual head design enables joint learning of move prediction and the rules of chess.

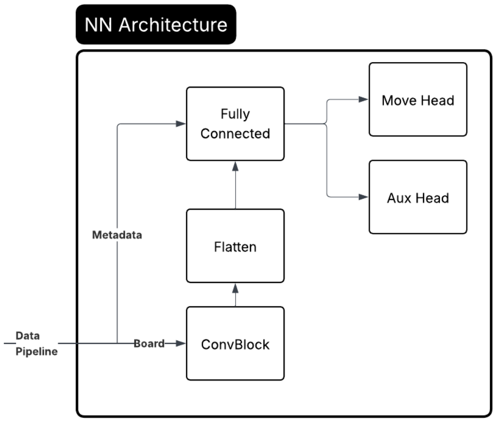

Fig. 3. A neural network architecture diagram showcasing the convolutional, flatten, fully connected layers, with the dual output heads for player move and legal move predictions.

### D. Training Process

*1) Training Parameters:* We began training the model with the following configurations:

- Batch size: 256
- Initial learning rate: 0.001
- Weight decay: 1e-4
- Beta parameters (Adam): (0.9, 0.999)
- Number of epochs: 50
- Data split: 80% training, 10% validation, 10% test

*2) Optimization Strategy:* The following combinations were explored during random search for hyperparameter optimization:

- Learning rates: [0.1, 0.01, 0.001, 0.0001]
- Weight decay rates: [1e-3, 1e-4, 1e-5]
- Beta values: [0.9, 0.95, 0.99]
- Momentum values: [0.9, 0.95, 0.99]

The best configuration was determined by the lowest validation loss.

*3) Hardware Configuration:* Training was done on embedded hardware. Note: a Jetson Orin Nano is roughly comparable to a Raspberry Pi 5 with a TOPS unit.

- Platform: NVIDIA Jetson Orin Nano 8GB Developer Kit
- GPU: 1024-core NVIDIA Ampere architecture GPU
- CPU: 6-core Arm Cortex-A78AE v8.2 64-bit CPU
- RAM: 8GB 128-bit LPDDR5
- Storage: 256 GB micro SD flash storage

## IV. Experiments

### A. Dataset Description

We trained our model on 15,167 games from January of 2013. These games were played by a wide breadth of players in a variety of states. The original Maia model was split into nine separate models by Elo ranges of 100 (e.g., 1100–1200); this limited the model's generalizability, so we trained our model on one complete dataset to increase generalizability. Expanding from 1100–1900 to 700–2500 to increase coverage.

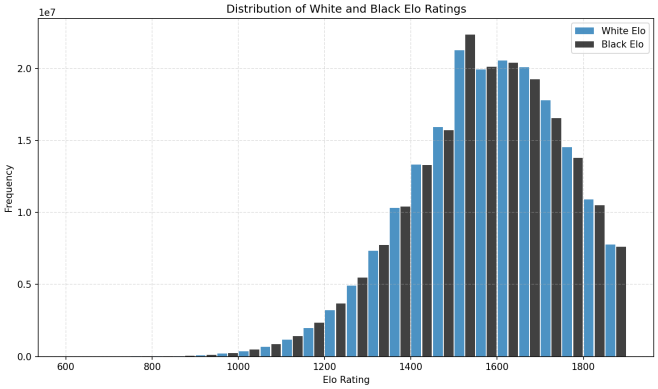

Fig. 4. Spread of the elos present in the training set.

Both iterations of the Maia paper were unable to handle openings because they do not model the first ten moves. This is a profound limitation, as the opening is one of the key areas where players struggle within a game. Knowing this, we chose to include the first ten moves to increase understanding of openings.

We also didn't limit the model to blitz games only, as we wanted to increase the overall variability in the games the model was capable of representing. We introduced this noise as it gives a more holistic view of human players, since blitz games comprise a small portion of the data and we wanted to represent a broader sample of human play.

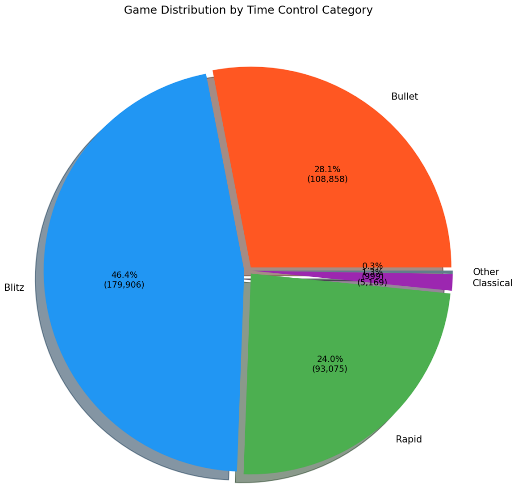

Fig. 5. Time Control Splits of the games from January of 2013.

For training, we generated approximately 1 million board snapshots. The dataset was split into 80% training (800,000 snapshots), 10% validation (100,000 snapshots), and 10% test (100,000 snapshots). Games were validated to remove incomplete or corrupted entries.

### B. Baseline Description

To evaluate Rylee's performance in predicting human-like moves, we compared against five baseline models representing different approaches to chess move prediction:

**Random Move Selection:** As a lower bound, we implemented a baseline that randomly selects from all legal moves in a given position.

**Random Forest Classifier:** We trained a Random Forest model on the same dataset to establish a classical machine learning baseline.

**Stockfish 15:** A traditional minimax-based chess engine with alpha-beta pruning and hand-crafted evaluation functions.

**Leela Chess Zero (LC0) 4200:** A neural network-based engine trained through self-play reinforcement learning, similar to AlphaZero.

**Maia-1 1500:** The primary comparison point for our work. Maia-1 is a supervised learning model specifically trained to predict human moves at the 1500 ELO level.

**Evaluation Metrics:** We use Top-1 accuracy (percentage of positions where the predicted move exactly matches the human move) and Top-5 accuracy (percentage where the human move appears in the top 5 predictions) as our primary metrics. These directly measure alignment with human decision-making. We also report filtered accuracy, which excludes noisy board states (blitz games only, exclude first 10 moves, filter games between narrow ELO ranges).

All of the source code for this project is available at https://github.com/EthanDGee/ryleeeeeeeeeeeee

### C. Experimental Evaluation

Our initial approach focused on training a model that is similar in scale and architecture to Stockfish. Stockfish is extremely small, and currently is the second strongest chessbot on the market. Furthermore, it can be deployed easily to edge devices due to its relatively small size, as it only consists of six fully connected layers. The deployability of its neural engine is due to its more limited role of simply evaluating board states and its NNUE (Efficiently Updatable Neural Network) architecture. Thus, its strength is due to its ability to effectively operate in a minimax search system. Accordingly, we used that architecture as our starting point. After training we achieved a meager score of around  3.5% accuracy.

Seeking to improve, we added convolutional layers to board state evaluation. Adding these layers is in line with our humanistic approach, as humans also use spatial features of the chessboard to evaluate next moves (e.g., knights are valued higher near the center of the board). This led to a large improvement, increasing accuracy to about 23.5%.

One further problem a machine learning-based approach faces is learning the rules of chess. To address this, Maia added an auxiliary head to improve the model's performance by having it predict the legal moves possible from a given board state. This was shown to have a profound impact. Implementing the auxiliary head in Rylee led accuracy to increase to about 25%.

TABLE I
COMPARISON OF MODEL ACCURACY

| Name | Accuracy |
|---|---|
| Random Moves | 6% |
| Random Forest | 13% |
| Stockfish | 40% |
| Leela 4200 | 44% |
| Maia1 1500 | 51% |
| Rylee Fully Connected | 3.5% |
| Rylee Conv | 23.5% |
| Rylee Conv with Aux | 25% |
| Rylee Conv with Aux Filtered | 35% |

*1) Training and Validation Performance:* Our final model with convolutional layers and auxiliary head demonstrated strong performance and generalization. Table II shows the comprehensive metrics across training and validation sets.

TABLE II
TRAINING AND VALIDATION METRICS FOR RYLEE CONV WITH AUX, WHERE 'FILTERED' INDICATES THE SAME FILTERING RULES USED IN THE MAIA PAPER.

| Metric | Training | Validation |
|---|---|---|
| Loss | 0.0152 | 0.0164 |
| Top-1 Accuracy | 27% | 25% |
| Top-5 Accuracy | 53% | 51% |
| Top-1 Accuracy (Filtered) | 36% | 35% |
| Top-5 Accuracy (Filtered) | 54% | 53% |

The close alignment between training and validation metrics indicates minimal overfitting, suggesting that Rylee has successfully captured generalizable patterns in human chess decision-making rather than memorizing specific positions. The filtered accuracy metrics, which exclude noisy board states, show that Rylee achieves 35% Top-1 accuracy on non-noisy board states.

*2) Computational Efficiency:* A key contribution of this work is demonstrating that human-aligned chess AI can be trained and deployed on edge devices:

- **Training Time:** 2-3 days on NVIDIA Jetson Orin Nano (vs. 3-4 weeks for Maia on dual A100 GPUs)
- **Preprocessing Time:** 1.5 hours (vs. 8 days for Maia)
- **Model Size:** 800,000 parameters (30× smaller than Maia)
- **Inference Speed:** 10,000 positions/second on edge hardware
- **Memory Footprint:** 14MB for model weights (deployable on Chromebooks and Raspberry Pis)

*3) Comparison to State-of-the-Art:* While Rylee's Top-1 accuracy is lower than Maia's, this comparison must be contextualized by the substantial differences in model size, training data scale, and computational requirements detailed above. The performance gap is expected given these constraints, but Rylee's Top-5 accuracy demonstrates that it captures human-like thinking in over half of positions when considering the top 5 candidate moves.

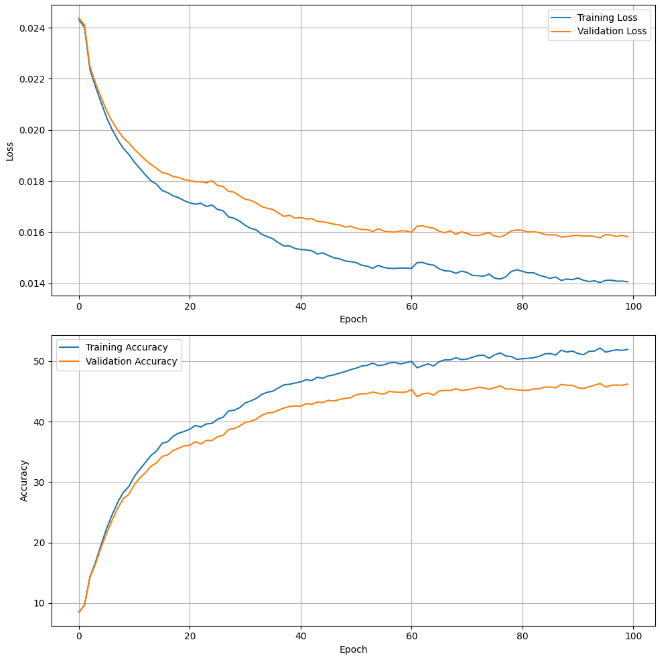

Fig. 6. Training and validation curves showing loss and accuracy over 50 epochs. The close alignment between training and validation metrics demonstrates strong generalization with minimal overfitting.

## V. Conclusion and Future Work

### A. Summary

We successfully developed Rylee, a compact neural network-based chess engine for human-aligned chess training. The key achievement is demonstrating that meaningful human-aligned behavior can be captured with far fewer resources than existing approaches, opening the door to accessible AI-based chess training tools. Limitations include the accuracy gap compared to larger models, and being trained on data from a limited time range (January 2013).

This work set out to create a human-like chess engine that is small enough to be deployed to edge devices such as mobile phones, Chromebooks and single board computers. Traditional chess engines provide limited educational value due to their focus on optimal play, while existing human-aligned models like Maia require substantial computational resources that restrict their accessibility.

We successfully developed Rylee, a compact neural network-based chess engine that achieves 25% Top-1 accuracy and 51% Top-5 accuracy in predicting human moves. Using only 800,000 parameters—30 times smaller than Maia's 25 million—Rylee can be trained in 2-3 days on edge devices like the NVIDIA Jetson Orin Nano and deployed on standard consumer hardware including Chromebooks and Raspberry Pi devices.

Rylee further improves upon Maia's capabilities by including opening move predictions, supporting a broader player base (700-2500 ELO vs. 1100-1900), and incorporating diverse game types. This allows Rylee to be a more practical tool as it covers a broader skill range.

### B. Future Work

*1) Training Enhancements:* **Larger Datasets:** Expanding beyond our current dataset of 15,000 games would likely lead to significant performance improvements.

**Time-Conditioned Training:** Including remaining time and time control as input features would allow the model to capture how time affects the way that humans make decisions, further improving its ability to play under different time controls.

**Cross-Validation:** Implementing k-fold cross-validation would provide more robust performance estimates. At our current speeds, it would take around 15 days to complete a 5-fold cross-validation.

*2) Feature Additions:* **ELO Prediction Model:** Adding an RNN-based model would allow Rylee to dynamically adjust in real time to player skill and improve the learning process.

**Blunder Detection:** Adding an auxiliary head to determine whether a player is likely to blunder (an incorrect move) would allow Rylee to identify areas where players at certain skill levels struggle. This would also allow for greater post-game analysis to further augment the learning process.

*3) Evaluation Extensions:* **User Studies:** Conducting controlled experiments where chess students practice against Rylee vs. traditional engines would quantitatively measure educational effectiveness and user engagement.

**Position Difficulty Analysis:** Analyzing model accuracy across position types and sections of the game would allow for more targeted improvements.

## References

[1] 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 Int. Conf. on Knowledge Discovery and Data Mining (KDD), 2020.

[2] D. Silver et al., "Mastering the game of Go with deep neural networks and tree search," Nature, vol. 529, pp. 484–489, 2016.

[3] 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.

[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," in Proc. 38th Conf. on Neural Information Processing Systems (NeurIPS), 2024.

[5] T. McGrath et al., "Acquiring human-like concepts from a superhuman AI," arXiv:2210.13432, 2022.

[6] R. McIlroy-Young, R. Wang, A. Anderson, and J. Kleinberg, "Detecting Individual Decision-Making Style: Exploring Behavioral Stylometry in Chess," in Proc. 35th Conf. on Neural Information Processing Systems (NeurIPS), 2021.

[7] A. Paszke et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in Advances in Neural Information Processing Systems 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035.

[8] N. Moskopp, "python-chess: a chess library for Python," GitHub repository, 2014. [Online]. Available: https://github.com/niklasf/python-chess

[9] Doshi-Velez, F., & Kim, B. (2017). A Roadmap for a Rigorous Science of Interpretability. ArXiv, abs/1702.08608.

[10] Stadie, B. C., Abbeel, P., & Sutskever, I. (2017). Third-person imitation learning. arXiv preprint arXiv:1703.01703.

[11] Charness, N. The impact of chess research on cognitive science. Psychol. Res 54, 4–9 (1992). https://doi.org/10.1007/BF01359217